EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

Instituto Politécnico Nacional
"La Técnica al Servicio de la Patria"

Research in Computing Science

Vol. 150 No. 6
June 2021

Research in Computing Science

RESEARCH IN COMPUTING SCIENCE, *Año 21, Volumen 150, No. 6, 04 de Junio de 2021, es una publicación mensual, editada por el Instituto Politécnico Nacional, a través del Centro de Investigación en Computación. Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, Ciudad de México, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor responsable: Dr. Grigori Sidorov. Reserva de Derechos al Uso Exclusivo del Título No. 04-2019-082310242100-203 ISSN: en trámite, ambos otorgados por el Instituto Nacional del Derecho de Autor. Responsable de la última actualización de este número: el Centro de Investigación en Computación, Dr.Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Fecha de última modificación 04 de Junio de 2021.*

RESEARCH IN COMPUTING SCIENCE, *Year 21, Volume 150, No. 6, June 04, 2021, is a monthly publication edited by the National Polytechnic Institute through the Center for Computing Research. Av. Juan de Dios Bátiz S/N, Esq. Miguel Othón de Mendizábal, Nueva Industrial Vallejo, C.P. 07738, Mexico City, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor in charge: Dr. Grigori Sidorov. Reservation of Exclusive Use Rights of Title No. 04-2019-082310242100-203. ISSN: pending, both granted by the National Copyright Institute. Responsible for the latest update of this issue: the Computer Research Center, Dr. Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othón de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Last modified on June 04, 2021.*

Advances in Artificial Intelligence

Félix Agustín Castro Espinoza (ed.)

Instituto Politécnico Nacional,
Centro de Investigación en Computación, México 2021

ISSN: en trámite

Formerly ISSNs: 1870-4069, 1665-9899

Instituto Politécnico Nacional (IPN)
Centro de Investigación en Computación (CIC)
Av. Juan de Dios Bátiz s/n esq. M. Othón de Mendizábal
Unidad Profesional "Adolfo López Mateos", Zacatenco
07738, México D.F., México

http://www.rcs.cic.ipn.mx
http://www.ipn.mx
http://www.cic.ipn.mx

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

ISSN 1870-4069

Table of Contents

Page

ISSN 1870-4069

Semáforo inteligente utilizando procesamiento digital de imágenes controlado por medio del perceptrón de Rosenblatt

Mario Rosa Otero, Natalia Sánchez Patiño, David Tinoco Varela

Universidad Nacional Autónoma de México,
Facultad de Estudios Superiores Cuautitlán,
México

{malttz15, natyangelessp}@gmail.com,
dativa19@hotmail.com

Resumen. En este artículo se presenta el diseño e implementación de un semáforo inteligente. Para su funcionamiento, este semáforo capta imágenes en tiempo real, pre procesa tales imágenes y las convierte en vectores de información binaria que se ingresan a un perceptrón dentro de un microcontrolador que se encargará de definir el cambio de colores en un semáforo para vehículos y para peatones. La propuesta de semáforo inteligente busca que sea un sistema capaz de distinguir entre personas y vehículos para garantizar una interacción adecuada entre estos dos entes y afianzar la seguridad de los peatones, sin embargo, en los experimentos realizados, se han utilizado figuras humanas de plástico, así como autos de juguete a escala, pero detallados lo suficientemente bien para poder representar la forma real de estas entidades. Originalmente se planteó la posibilidad de sólo utilizar la binarización de una imagen para su clasificación dentro del perceptrón, sin embargo, esta situación identificaba las posiciones de los objetos a clasificar en vez de la distinción entre vehículo y peatón, por tal motivo se generó un sistema de identificación diferente, en el cual se extraen características de la imagen correspondientes a su área dentro de un entorno y estas características eran clasificadas por el perceptrón, dando una convergencia de gran robustez. Es posible obtener el código total del proyecto en[1].

Palabras clave: Perceptrón, inteligencia artificial, procesamiento de imágenes, redes neuronales.

Intelligent Traffic Light Using Digital Image Processing Controlled by Rosenblatt Perceptron

Abstract. This article presents the design and implementation of an intelligent traffic light. For its operation, this traffic light captures images in real time, pre-processes such images and converts them into vectors of binary information that are entered into a perceptron within a microcontroller that will be in charge of defining the change of colors in a traffic light for vehicles and for pedestrians.

The intelligent traffic light proposal seeks to be a system capable of distinguishing between people and vehicles to guarantee an adequate interaction between these two entities and strengthen the safety of pedestrians, however, in the experiments carried out, plastic human figures have been used, as well as toy cars to scale, but detailed well enough to be able to represent the real form of these entities. Originally, the possibility of only using the binarization of an image for its classification within the perceptron was raised, however, this situation identified the positions of the objects to be classified instead of the distinction between vehicle and pedestrian, for this reason a different identification system was generated, in which characteristics of the image corresponding to its area within an environment are extracted and these characteristics were classified by the perceptron, giving a convergence of great robustness. It is possible to obtain the total code of the project[1].

Keywords: Perceptron, artificial intelligence, image processing, neural networks.

1. Introducción

Con el crecimiento de áreas como Inteligencia Artificial (IA), Big Data (BD), Internet de las Cosas (IoT), Industria 4.0, entre otras; se ha popularizado el uso de sistemas de ingeniería para control automático e inteligente de espacios comunes. Un sistema dentro de una ciudad, así como dentro de algunos edificios industriales y comerciales, que requiere de forma intrínseca un control (inteligente), es el sistema que componen los semáforos de paso.

Estos elementos están centrados en espacios donde es necesario tomar el control de un flujo, principalmente, vehicular y de esta manera eficientar la movilidad de un entorno, además de esto, mediante estos sistemas de control también se busca lograr una interacción segura entre dos entes que comparten las zonas de acción, como pueden ser, los peatones y los vehículos. Por los motivos mencionados, se vuelve importante e interesante, la generación de semáforos que reaccionen de forma inteligente a las circunstancias del entorno en los que se ven inmersos.

Por lo que, el esquema planteado en este artículo implica construir un semáforo independiente, que sea capaz de ceder el paso a humanos o vehículos motorizados, reaccionando de forma autónoma de acuerdo a las imágenes que el sistema adquiera en tiempo real, estas imágenes serán pre procesadas, evaluadas por el perceptrón y esto finalmente producirá una respuesta que se reflejara en el cambio de luz del semáforo.

Si se consigue trasladar el esquema planteado en este proyecto, a un sistema utilizable en calles reales, donde será necesario tomar en cuenta más parámetros de variación dados en las condiciones de dicha situación, podríamos optimizar los tiempos de paso de acuerdo a la afluencia entre carros y personas, mejorando así, la eficiencia de movilidad en una ciudad. Adicionalmente, con este semáforo se busca mejorar la seguridad de los peatones.

[1] https://github.com/NM-Labs/Semaforo-Inteligente-Con-Perceptron

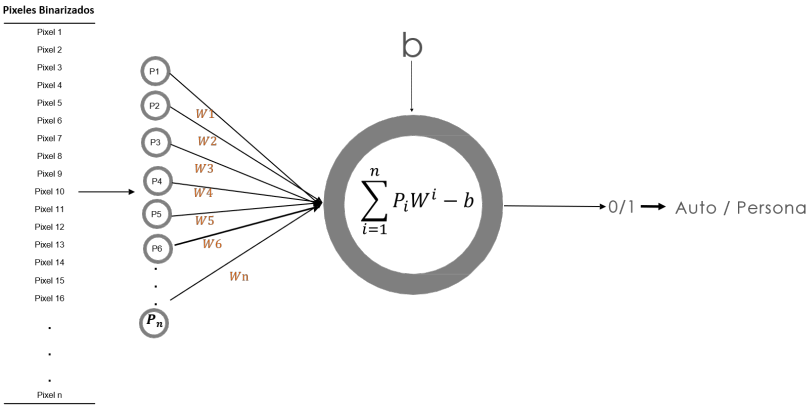


Fig. 1. Representación de un perceptrón para clasificación de imágenes.

2. Estado del arte

Normalmente los semáforos cambian de estado en sus luces de acuerdo a tiempos de activación y desactivación, al menos en México, sin embargo, estos esquemas no son de ayuda en eficientar el control vehicular y el control de paso peatonal ya que no consideran las condiciones de flujo. En una búsqueda de lograr un semáforo eficiente y que responda a las necesidades del entorno en donde se encuentra, se han ideado los semáforos autónomos, estos tratan de responder dinámicamente a las situaciones vehiculares y ciudadanas que se presentan en su entorno.

Existen diferentes, y muy variadas, propuestas para la generación de semáforos de control autónomos, algunas de ellas trabajando sólo mediante sensores y otras de ellas realizando análisis inteligente de los datos de entrada. En esta sección vamos a considerar las diferentes propuestas existentes para el diseño de los semáforos vehiculares, de forma general, y los semáforos peatonales, en forma particular.

Algunas de las propuestas de semáforos existentes están enfocadas al análisis de la densidad vehicular por medio de sensores IR y así lograr ranuras de tiempo dinámico [5]; otras propuestas consideran una conexión al IoT [9] o una conexión a internet por medio de redes de sensores inalámbricos [12].

Obviamente el uso de la IA ha jugado un papel importante en el desarrollo de estos nuevos esquemas de control, utilizando muy variados algoritmos y técnicas para lograr el objetivo, por ejemplo el uso de algoritmos de optimización adaptativa basado en el aprendizaje por refuerzo [16], el diseño de sistemas expertos para control de semáforo dinámico y automático [15], técnicas de aprendizaje profundo para lograr un ajuste dinámico de los tiempos de trafico real [14], utilización de controladores difusos para el control de semáforos en situaciones de emergencia [11] (hay que mencionar que existen una gran cantidad de propuestas basadas en lógica difusa, entre las que podemos observar [6, 8, 1]), y por supuesto, el uso de redes neuronales, como lo son el control basado en la teoría de la red neuronal de extensión (ENN) para encrucijadas [3] o el uso de redes neuronales para predecir y ajustar los tiempos de las señales en ambos lados de la carretera al mismo tiempo [10].

Fig. 2. Representación de coches y personas.

Hasta este punto, se ha hablado del diseño de diferentes tipos de semáforos vehiculares, sin embargo, un aspecto de principal interés dentro de este proyecto es el diseño de los semáforos que permiten el paso de peatones en un entorno vehicular.

Al igual que los semáforos vehiculares, este otro tipo también cuenta con una gran cantidad de propuestas dentro de la literatura científica, por ejemplo: el uso de algoritmos genéticos para mejorar el rendimiento del control de semáforos con paso de peatones [13], el diseño y entrenamiento de una red neuronal de memoria a corto plazo para predecir las intenciones de los peatones al cruzar la luz roja y evitar accidentes [17], la utilización del aprendizaje Q multi agente distribuido [7], así como el uso de redes neuronales convolucionales para el análisis de los datos del entorno [2].

Como se ha descrito en esta sección, existen diferentes técnicas para el control de semáforos, tanto vehiculares como peatonales, algunas de ellas con gran potencia de procesamiento y conexión a internet, sin embargo, la propuesta realizada en este artículo se enfoca al desarrollo de un semáforo que pueda ser implementado en entornos donde no se cuente con grandes cantidades de potencia de procesamiento y de conexión a internet, es decir, que pueda reaccionar de forma adecuada sin importar si existe conexión o no a internet.

3. Marco teórico: Perceptrón de Rosenblatt

Un perceptrón es la unidad básica de una red neuronal y consiste en una neurona con pesos sinápticos y un bias que son ajustables y puede ser usado para la clasificación en problemas linealmente separables. El algoritmo usado para ajustar los parámetros libres de esta red neuronal apareció por primera vez en un procedimiento de aprendizaje desarrollado por Rosenblatt (1958) para su modelo de perceptrón del cerebro.

Dicho perceptrón está limitado a realizar clasificación de patrones con sólo dos clases. Expandiendo la capa de salida del perceptrón para incluir más que una neurona, podemos realizar dicha clasificación con más de dos clases. Sin embargo, las clases tendrían que ser linealmente separables para que el perceptrón trabaje correctamente.

En la figura 1, podemos observar la representación básica del perceptrón. En la figura 1 los pesos sinápticos se denotan por $W_1, W_2, \ldots, W_n$ y la información que entra al perceptrón son los datos que representan al problema a clasificar, para fines de este proyecto, estos datos serán los píxeles de una imagen, representados como $P_1, P_2, \ldots, P_n$, y donde b es el sesgo o umbral que se añade al proceso.

Después de realizar la operación de suma de la multiplicación entre pesos sinápticos y entradas, para posteriormente sumar el sesgo y obtener una salida, dicha salida será

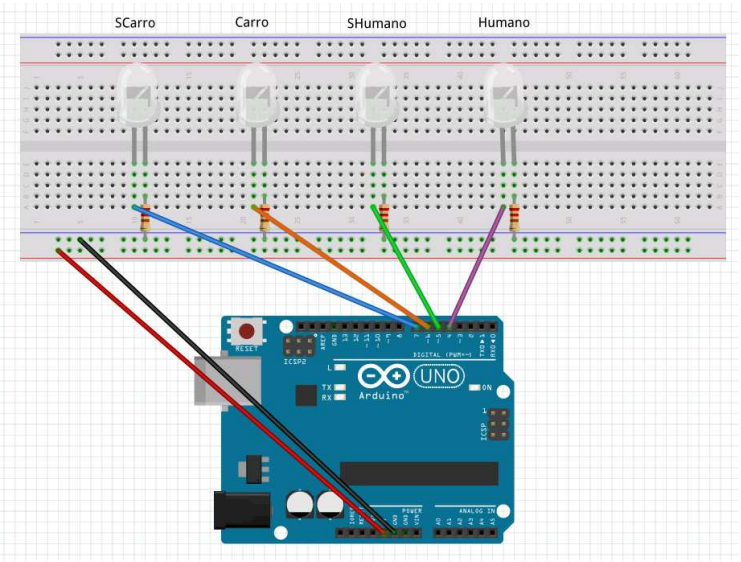


Fig. 3. Esquema de conexión del arduino.

evaluada por una función de activación, en este caso una función escalón de Heavisise, el cual establece un límite para determinar si la imagen contiene una determinada clase.

El aprendizaje de este algoritmo se basa en el cálculo del error obtenido de la diferencia entre la salida esperada y_e con respecto a la salida obtenida y_i, entonces el error es calculado como: $e = y_e - y_i$.

A partir de este se van a actualizar los pesos sinápticos y el sesgo, intentando adaptarse a las entradas ingresadas e intentando buscar una configuración de los mismo con los que se puede obtener una salida esperada para todos los elementos, dentro de las operaciones de actualización interviene también un valor conocido como factor de aprendizaje, o tasa de aprendizaje denotado como λ, este factor regula la cantidad de corrección de error para no divergir en la respuesta al intentar corregir. [4] A continuación se muestran las ecuaciones utilizadas para actualización de sesgos y pesos:

$$W_n[t+1] = W_n[t] + \lambda(y_e - y_i)P_n, \tag{1}$$

$$b[t+1] = b[t] + \lambda(y_e - y_i)x. \tag{2}$$

4. Descripción del sistema

De forma general se da la descripción del sistema, para posteriormente ir detallando cada paso: Una cámara web enviará información de las imágenes tomadas hacía Matlab, en donde se realiza el procesamiento de las mismas, realizando esta vinculación por medio de la URL dada por IP webcam, la cual contendrá la imagen en tiempo real. La imagen pre procesada y vectorizada se guarda en archivos .cvs, los cuales se enviaran al perceptrón, y a su vez, el perceptrón regresará el resultado de la clasificación hacia Matlab, quién realizará el control de la tarjeta Arduino.

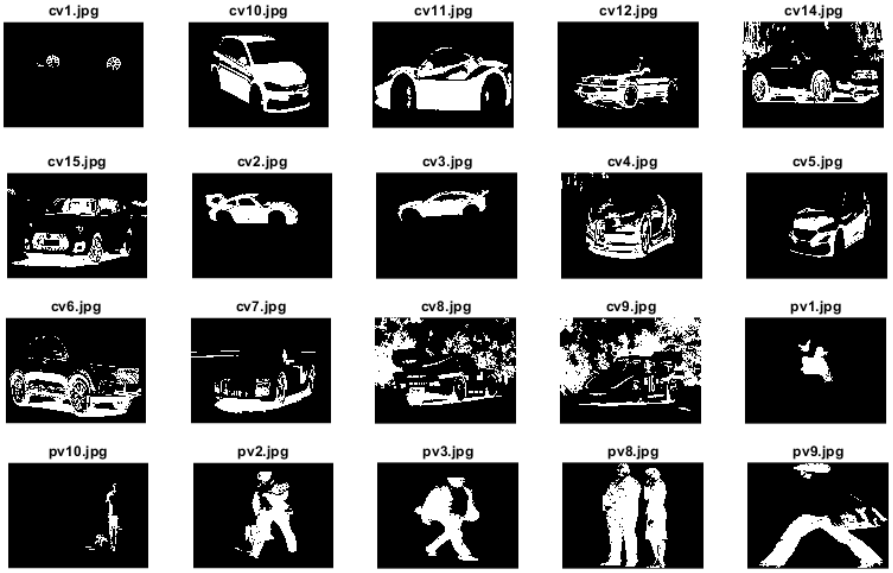


Fig. 4. Ejemplos de imágenes binarizadas que presentan ruido en estructura.

4.1. Adquisición y almacenamiento de datos

Para la obtención de un conjunto de datos representativo del problema, se procedió a colocar un fondo blanco y los elementos a utilizar en él, en esta forma, se tomaron fotografías a los objetos a clasificar, cuidando que no existan tantas variaciones de ángulo, distancia e iluminación entre las fotografías tomadas, para mantener controladas las condiciones del problema.

Siendo además importante tomar la misma cantidad de imágenes para el total de categorías a clasificar, en este caso misma cantidad de imágenes de autos y personas, estas imágenes se pueden ver en la figura 2. Del total de imágenes captadas de los elementos, se tomó un porcentaje entre el $15\,\% - 30\,\%$ que serán las muestras de validación, y el resto serán imágenes para entrenamiento, verificando que haya variedad de las mismas respecto a la variación de posición entre los elementos utilizados para ambas clases.

4.2. Pre-procesamiento de datos

Con el fin de simplificar la información contenida en una imagen y que pueda ser clasificable por un perceptrón se realizan métodos de pre-procesamiento digital de imágenes, en este procedimiento lo primero que se realiza es cambiar las imágenes a otro espacio de color, en este caso la imágenes cuando se ingresan están dadas en el espacio de color RGB, pero estas como primer paso sufren una transformación hacia el espacio CMYK, que tiene cuatro capas. Después de realizar la transformación se toma solo la capa *Y*, que resalta un rango de tonalidades en específico.

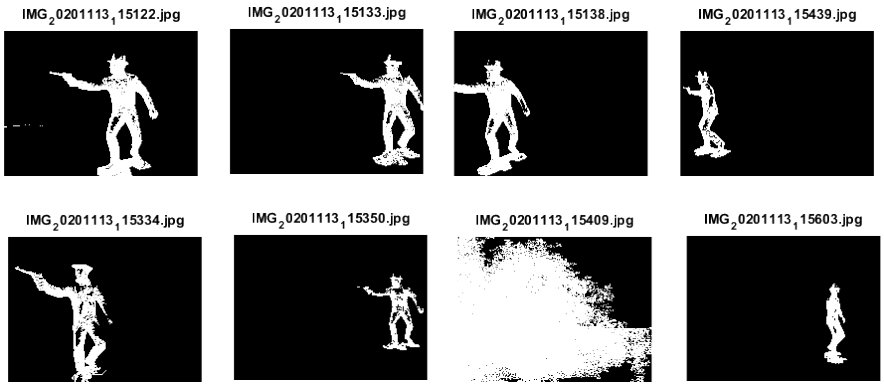


Fig. 5. Ejemplos de humanos binarizados.

Posteriormente dicha imagen en la capa *Y*, se pasa por una función de contraste que resalta aún más ciertos rangos de valor en la imagen, en esta caso se utiliza una función cúbica para realizar la mejora de contraste. Los píxeles $P_1, P_2, ..., P_n$ son utilizados para la operación:

$$\frac{P_n^3}{256^2}. \tag{3}$$

Esto dado que el rango de valores de píxeles están en una escala de 0 a 255. Posterior a esto, se pasa los pixeles de la imagen, ya con mejoras de contraste, por el algoritmo de Otsu, el cual busca en los valores de píxeles el valor de umbral que minimice la varianza σ^2 entre los mismos, el cual va a dar un umbral para realizar la binarización de la imagen.

El proceso del algoritmo requiere de obtener el histograma de la imagen, y obtener las probabilidades de cada nivel de intensidad de los píxeles, se establecen pesos iniciales $W_n(0)$ o probabilidades, y se obtiene así mismo las medias entre clases $\mu_i(0)$, con estos datos para cada nivel de intensidad se itera para obtener las varianzas entre clase, y después de tener todas las varianzas, toma la mayor varianza entre dos clases para establecer el umbral de binarización.

Posterior a este procedimiento, se utilizan algunos Momentos de Hu, que describen la manera en que se distribuyen los píxeles de un objeto sobre el plano de una imagen, estos momentos son invariantes a las transformaciones geométricas como traslación, rotación y escalamiento. Estos momentos se calculan para eliminar ruido en las imágenes binarizadas, dado que se discriminan objetos que hayan superado el umbral establecido por el algoritmo de Otsu, discriminándolos por algunas propiedades como el área.

Es necesario mencionar que se binarizan las imágenes para posteriormente convertirlas en vectores de entrada, vectores que contendrán únicamente dos valores, simplificando así, el proceso de clasificación del perceptrón. Durante el proceso de clasificación, se generó una respuesta errónea que será explicada más adelante, y debido a esta situación, al proceso ya mencionado, posteriormente se le agrego la extracción de características a las imágenes binarizadas, por medio de cajas de identificación, a estas cajas se le extrajeron datos como las distancias en los ejes X y Y, y el área de tales cajas.

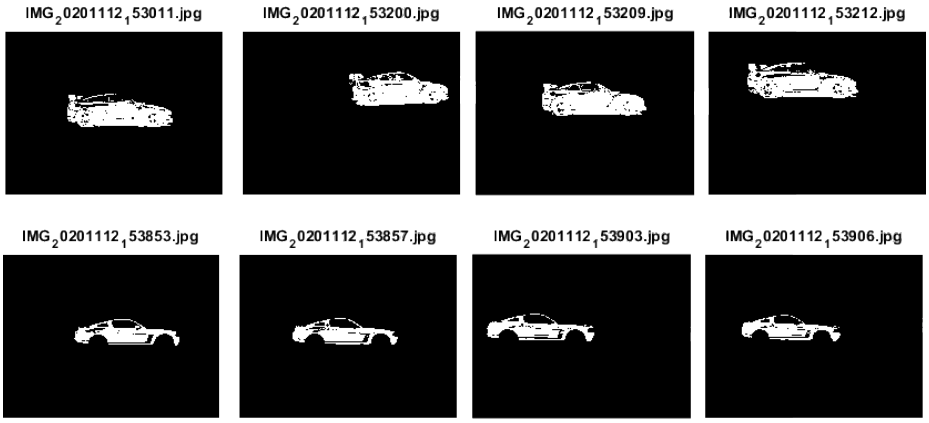


Fig. 6. Ejemplos de coches binarizados.

4.3. Entrenamiento de la red

Para poder ingresar las imágenes al perceptrón, estas tienen que ser re-dimensionadas, ya que se tienen matrices, que son convertidas a vectores. Realizado el pre-procesamiento a todas las imágenes de entrenamiento, se forma una matriz de características, es decir, se forma una matriz con todas las imágenes de entrenamiento ya re-dimensionadas como vectores, cada vector es una imagen binarizada, y se construye el vector de salidas esperadas conforme se acumularon las imágenes en la matriz de características.

Con estos cambios en los detalles de las imágenes, se puede realizar el entrenamiento. La construcción del perceptrón y su algoritmo de entrenamiento se divide en dos partes principales: La primera parte es con la que se realiza el entrenamiento, donde se lee la matriz de características de un archivo externo, y se realiza el mismo procedimiento para la lectura del vector de valores esperados, así como se realiza la inicialización de pesos y sesgo.

Cada una de las imágenes es pasada por el perceptrón, logrando así el entrenamiento de la neurona, cuando el entrenamiento es alcanzado, se guarda el vector de pesos en un archivo así como el sesgo, para que sean posteriormente utilizados al momento de realizar simulación o simplemente implantar el sistema; La segunda parte del código es una función para realizar la simulación del sistema en tiempo real.

4.4. Simulación del sistema

Una vez que se tuvieron todos los elementos integrados y ejecutándose de forma correcta, se inicia con la simulación del sistema, armando un circuito como el que se muestra en la figura 3, donde el LED con el nombre ”SHumano” representa el alto para peatones mientras que ”Humano” representa el siga. De manera similar pasa con los nombres de ”SCarro” y ”Carro”.

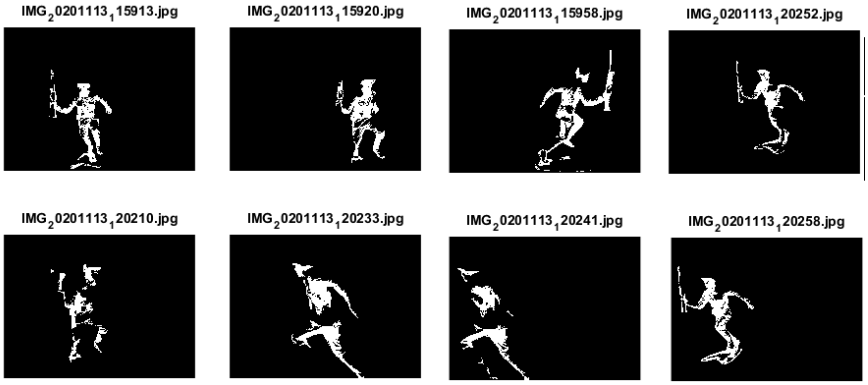


Fig. 7. Ejemplos de humanos binarizados con imagen limpia.

5. Resultados obtenidos: Binarización de imágenes

Al inicio se realizaron pruebas con imágenes y fotografías tomadas de internet donde se aplicó el binarizado a dichas imágenes, con esto se verificó que tan bien se realizaba la segmentación de los objetos de interés.

Como podemos ver en la figura 4 las imágenes al tener una gran cantidad de elementos pueden resultar muy ruidosas. Las mejores binarizaciones de imágenes fueron las personas que tenían un fondo blanco o transparente, así como autos en un entorno claro y sin demasiadas cosas alrededor. Debido al ruido que presentan las imágenes de la figura 4 , se decidió utilizar imágenes de un entorno controlado, donde las condiciones fueran óptimas para la binarización y detección de dichas imágenes.

5.1. Experimentos de clasificación

De manera formal, y para la clasificación de imágenes de autos y personas, se realizaron dos modelos experimentales. Las diferencias principales entre ambos, están dadas por el número de imágenes utilizadas para entrenamiento y validación, así como diferencias en la forma de clasificación.

Modelo experimental 1. Para la realización de este proyecto, se realizaron diferentes modelos experimentales hasta lograr una adecuada interpretación de la información obtenida, es decir, que la convergencia del perceptrón fuera adecuada al problema a resolver. Para el primer experimento de este modelo se utilizaron 48 imágenes en total de entrenamiento, contemplando 24 por clase, y 12 para validación. Las imágenes fueron tomadas mediante las cámaras de dos celulares distintos, ambos en un escenario blanco, como se puede ver en la figura 2.

Posteriormente se almacenaron, organizaron y binarizaron dichas imágenes, el resultado de binarización de algunas de ellas se puede ver en las figuras 5 y 6. En un segundo experimento, se utilizó un conjunto nuevo de imágenes más grande que el utilizado en el experimento 1 (100 imágenes de entrenamiento, 50 por clase, y 15 para validación), donde se cuidaron más las condiciones del entorno al momento de tomar las imágenes buscando un escenario más uniforme en texturas e iluminación.

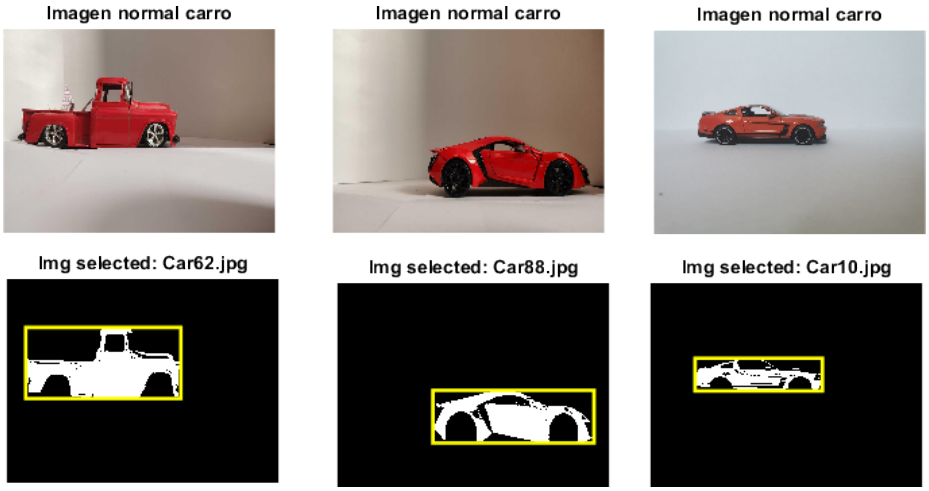


Fig. 8. Ejemplos de carros encerrados en una caja a la cual se le extraen carcterísticas tales como dimensiones y área.

Esto ayudó a que sobre todo las imágenes de humanos salieran más limpias al momento de realizar la binarización. Algunos ejemplos se pueden observar en la figura 7. Para este primer par de experimentos, observamos que en ciertas imágenes no se alcanza a segmentar el objeto de interés de forma correcta, esto debido a falta de control en condiciones del entorno como la iluminación y sombras creadas por terceros objetos, como se ve en la figura 5.

A pesar de esto, el perceptrón logró hacer una adaptación exitosa a los datos de entrenamiento, pero una clasificación menor en los datos de validación, presentando errores al momento de clasificar en tiempo real. Se realizaron 20 pruebas con la iluminación parecida a la que se tenía en los datos de entrenamiento, 10 imágenes tomadas en tiempo real por cada una de las cámaras utilizadas, cada una en su respectivo escenario.

Conforme se fueron analizando los resultados de las pruebas en tiempo real, tanto para el modelo obtenido con en el experimento uno, como en el experimento dos, con el sistema completamente armado se pudo notar que, al cambiar las condiciones de luz, los resultados se veían afectados. Con una buena iluminación, es decir midiendo de día y sin crear muchas sombras en el escenario, 100 % de las predicciones eran correctas, siempre y cuando se mantuvieran en una posición determinada para cada una de las figuras, izquierda para carros y derecha para personas.

Pero al hacer experimentos con variación de iluminación el rendimiento decaía a un: 60 %, cometiendo errores más frecuentemente cuando se trataba de coches. A pesar de los resultado de la clasificación, se observó un detalle sumamente importante en estos experimentos, y es que, lo que estaba clasificando el perceptrón eran las zonas donde había activaciones en lugar de los objetos que se buscaban detectar, es decir, no identificaba al objeto en sí mismo, identificaba la posición del objeto y en base a ella, determinaba la clase. Así, se observó que, si había objetos blancos en la parte derecha de la imagen binarizada, el resultado de la clasificación era ‘persona’.

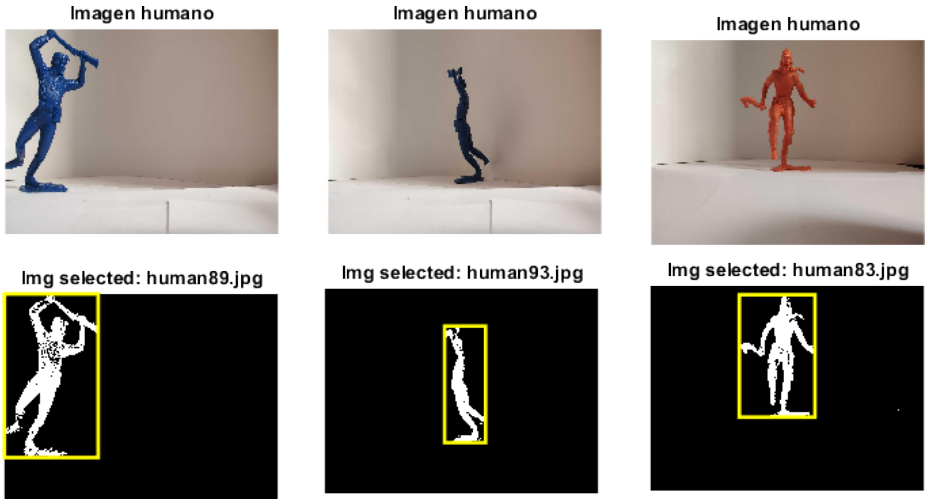


Fig. 9. Ejemplos de humanos encerrados en una caja a la cual se le extraen carcterísticas tales como dimensiones y área.

Si había objetos blancos a la izquierda de la imagen binarizada el resultado de la predicción era 'carro'. Lo cual hacía el sistema débil, y que clasificase mal si los objetos no se encontraban en la zona correcta.

Modelo experimental 2. Para solucionar el problema encontrado en los experimentos mencionados anteriormente, se buscó la forma en la que el perceptrón identificara características particulares de los objetos a clasificar, tratando de evitar que se centrara en la posición de tales objetos, que fue el caso anterior.

Para este fin, el siguiente paso fue hacer un entrenamiento, donde las imágenes tuvieran más elementos diferenciadores entre sí, es decir, hubiese más variabilidad respecto a las condiciones de iluminación, utilizando más modelos de objetos, variación en el ángulo en profundidad con que se colocaba el objeto respecto a la cámara, y también para poder retirar el sesgo de posición existente se tomaron imágenes con los objetos a diferentes distancias y ubicando el objeto en distintas zonas de la imagen, obteniendo así, variabilidad en la posición, y el tamaño.

Para ello, se tomaron de las imágenes previamente existentes, y se quitaron de ese conjunto las que tenían sombras muy marcadas y por ende, los resultados de la binarización eran muy inadecuados, con datos para entrenamiento. Adicionalmente, se agregaron las nuevas imágenes con mayor variabilidad. Dejando así, un total de 260 imágenes de entrenamiento, 130 de autos, y 130 de personas. Para validación se utilizaron un total de 90 imágenes, 45 de autos, y 45 de personas.

Con estas imágenes se relizó pre procesamiento mayor al ya realizado con la binarización, y en vez de pasarle la imagen aplanada al perceptrón, lo que se envió a la neurona fue el resultado de la extracción de características de cada una de las imágenes tomadas.

Como una primera parte en la identificación de objetos, se tomaron en cuenta 18 características, las cuales se enviaron al perceptrón: circularidad, área, solidez, longitud

Tabla 1. Cantidad de imágenes utilizadas para el entrenamiento y validación de los modelos experimentales.

Modelo experimental 1						
	Experimento 1			**Experimento 2**		
	Entrenamiento	**Validación**	**Pruebas en tiempo real**	**Entrenamiento**	**Validación**	**Pruebas en tiempo real**
Autos	24	12	10	50	15	10
Personas	24	12	10	50	15	10
Modelo experimental 2						
	Experimento 1					
	Entrenamiento		**Validación**	**Pruebas en tiempo real**		**Fuera de distribución**
Autos	130		45	30		7
Personas	130		45	30		8

del eje mayor, longitud del eje menor, perímetro, excentricidad, diámetro equivalente, tasa de pixeles de la figura dentro de la caja entre los pixeles totales de la caja,longitud sobre el eje x de la caja que encierra al área mayor, longitud sobre el eje y de la caja que encierra al área mayor, primer momento de Hu, segundo momento de Hu, tercero momento de Hu, cuarto momento de Hu, quinto momento de Hu, sexto momento de Hu y séptimo momento de Hu. Al entrenar, el perceptrón no logró converger para dichas características en el número de épocas límite (100,000).

Debido a ello, se bajó la dificultad y nos quedamos únicamente con 3 características: Área y longitud del área en el eje x y en el eje y, haciendo mención que estas características se extraen directamente por medio del procesamiento de imágenes, encapsulando los objetos y obteniendo las áreas de las cajas en las cuales se encapsulaban, estas cajas pueden verse en las figuras 8 y 9 (Es importante mencionar que al tener imágenes binarizadas, el encapsulamiento de objetos es óptimo, ya que identifica los espacios blancos del conjunto negro).

Con estas características logró converger en 18,162 épocas, en 1 hora 1 minuto y 54 segundos. El modelo logró ajustarse a todas las imágenes de entrenamiento, y todas las de validación, obteniendo un 100 % de rendimiento para ambos conjuntos. En el cuadro 1, es posible resumir la cantidad de imágenes utilizadas para el entrenamiento y ejecución de los dos modelos experimentales descritos.

6. Experimentación en tiempo real

Al realizar las pruebas en tiempo real con los pesos obtenidos para el 2do. modelo, se observó una exactitud del 100 %, cuando el total de pruebas realizadas fue de 30 para cada objeto, es decir 60 pruebas en tiempo real. Aun después de ello, se le pasaron al sistema imágenes que salían completamente de la distribución de las imágenes de entrenamiento, con cámaras muy diferentes, bajo condiciones de iluminación distinta, incluso iluminación artificial, y el modelo, para 15/15 de estas imágenes dio solución correcta.

En la figura 10, se puede apreciar el tipo de condiciones en donde se realizó la obtención de imágenes de entrenamiento, validación y prueba en tiempo real. Con lo que se deja en claro que el experimento tuvo condiciones controladas con la intención de que se pudiera hacer un buen binarizado y fuera clasificable para un perceptrón.

Fig. 10. Representación de coches y personas en escenario controlado.

7. Conlusiones

En este proyecto se realizó la implementación de un sistema controlador de un semáforo inteligente, capaz de detectar a una persona, o a un auto, y ceder el paso, conforme al objeto presentado frente a la cámara de detección, para esto se utilizaron dos lenguajes de programación distintos, los cuales brindan bondades, en uno de los casos, para manejo de datos, y pre-procesamiento de imágenes, y en el segundo caso, rapidez de cálculo para la toma de una decisión mediante el procesamiento de la información. Durante el desarrollo de este proceso, surgió una situación interesante al momento de realizar el entrenamiento de la neurona.

Al binarizar la imagen en el primer modelo experimental, el perceptrón logró identificar un patrón, sin embargo, este patrón no era el esperado ya que identificaba las posiciones con respecto a la línea peatonal, es decir, si el objeto estaba delante de la línea peatonal, lo identificaba como una persona; pero si el objeto estaba detrás de la línea peatonal, lo identificaba como un vehículo. Esta situación dio como resultado una falsa interpretación de las circunstancias a evaluar por la neurona.

Sin embargo, este error permitió la generación de una idea diferente para la solución del problema, en vez de considerar las posiciones de los objetos, se consideró la extracción de las características de los objetos, como el tamaño y forma de cada uno de ellos, lo que desemboco en que el perceptrón identificara correctamente una persona o un vehículo, es importante mencionar, que para la obtención de estas características no fueron necesario un cálculo o procesamiento complejo de las imágenes, ya que, según los experimentos, basta con distinguir las distancias que ocupan en el espacio, así como el área de cada objeto.

Las características relacionadas a los tamaños y formas de los objetos, fueron un punto primordial en la caracterización realizada por la neurona, estas características fueron extraídas gracias a las cajas en las que se encapsularon los objetos.

Bajo el problema planteado, un procesamiento sencillo por medio de una sola neurona, es adecuado, debido a que una persona siempre tendrá un área dentro del espacio más reducida que un vehículo, por lo que, para este proyecto, no fue necesario generar redes de extracción de características más complejas y de mayor tiempo de ejecución. En este artículo se pudo realizar experimentación con el sistema creado fuera del ámbito virtual, y comprobando su respectivo funcionamiento, con pruebas en tiempo real a través de la tarjeta de desarrollo Arduino.

8. Trabajo futuro

Uno de los fines de este proyecto, es la búsqueda de sistemas inteligentes, en este caso un semafóro, que puedan ser utilizados en entornos en los que no se tenga una capacidad alta de procesamiento computacional, por lo que se buscará que el siguiente paso sea la implementación de todo el proceso en una tarjeta de desarrollo de bajo costo, que pueda ejecutar la propuesta de forma autónoma e independiente, tal como una tarjeta Raspberry Pi.

Agradecimientos. Este proyecto fue apoyado por los proyectos PAPIIT IN105219, PAPIME PE100221 y PIAPI 2053 de la UNAM y de la FES-C.

Referencias

1. Arora, M., Banga, V. K.: Real time traffic light control system using morphological edge detection and fuzzy logic. In: Proceedings of the 2nd International Conference on Electrical, Electronics and Civil Engineering, pp. 28–29 (2012)
2. Ash, R., Ofri, D., Brokman, J., Friedman, I., Moshe, Y.: Real-time pedestrian traffic light detection. In: Proceedings of the IEEE International Conference on the Science of Electrical Engineering in Israel, pp. 1–5 (2018) doi: 10.1109/ ICSEE.2018.8646287
3. Chao, K. H., Lee, R. H., Wang, M. H.: An intelligent traffic light control based on extension neural network. In: Proceedings of the International Conference on Knowledge-Based and Intelligent Information and Engineering Systems, pp. 17–24 (2008) doi: 10.1007/978-3-540-85563-7_8
4. Gallant, S. I.: Perceptron-based learning algorithms. In: Proceedings of the IEEE Transactions on Neural Networks, vol. 50, no. 2, pp. 179–191 (1990) doi: 10.1109/72.80230
5. Ghazal, B., ElKhatib, K., Chahine, K., Kherfan, M.: Smart traffic light control system. In: Proceedings of the Third International Conference on Electrical, Electronics, Computer Engineering and their Applications, pp. 140–145 (2016) doi: 10.1109/EECEA.2016.7470780
6. Kulkarni, G. H., Waingankar, P. G.: Fuzzy logic based traffic light controller. In: Proceedings of the International Conference on Industrial and Information Systems, pp. 107–110 (2007)
7. Liu, Y., Liu, L., Chen, W. P.: Intelligent traffic light control using distributed multi-agent Q learning. In: Proceedings of the 20th International Conference on Intelligent Transportation Systems, pp. 1–8 (2017) doi: 10.1109/ITSC.2017.8317730

8. Mehan, S.: Introduction of traffic light controller with fuzzy control system. International Journal of Electronics & Communication Technology, vol. 2, no. 3, pp. 119–122 (2011)
9. Miz, V., Hahanov, V.: Smart traffic light in terms of the cognitive road traffic management system based on the internet of things. In: Proceedings of IEEE East-West Design and Test Symposium, pp. 1–5 (2014) doi: 10.1109/EWDTS. 2014.7027102
10. Rizwan, J. M., Krishnan, P. N., Karthikeyan, R., Kumar, S. R.: Multi layer perception type artificial neural network based traffic control. Indian Journal of Science and Technology, vol. 9, no. 5, pp. 1–6 (2016) doi: 10.17485/ijst/ 2016/v9i5/87267
11. Salehi, M., Sepahvand, I., Yarahmadi, M.: TLCSBFL: A traffic lights control system based on fuzzy logic. International Journal of u-and e-Service, Science and Technology, vol. 7, no. 3, pp. 27–34 (2014) doi: 10.14257/ijunesst.2014.7.3.03
12. Tubaishat, M., Shang, Y., Shi, H.: Adaptive traffic light control with wireless sensor networks. In: Proceedings of the 4th IEEE Consumer Communications and Networking Conference, pp. 187–191 (2007) doi: 10.1109/CCNC.2007.44
13. Turky, A. M., Ahmad, M. S., Yusoff, M. Z., Hammad, B. T.: Using genetic algorithm for traffic light control system with a pedestrian crossing. In: Proceedings of the International Conference on Rough Sets and Knowledge Technology, vol. 5589, pp. 512–519 (2009) doi: 10.1007/978-3-642-02962-2_65
14. Wei, H., Zheng, G., Yao, H., Li, Z.: IntelliLight: A reinforcement learning approach for intelligent traffic light control. In: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2496–2505 (2018) doi: 10.1145/3219819.3220096
15. Wen, W.: A dynamic and automatic traffic light control expert system for solving the road congestion problem. Expert Systems with Applications, vol. 34, no. 4, pp. 2370–2381 (2008) doi: 10.1016/j.eswa.2007.03.007
16. Wiering, M. A., Veenen, J., Vreeken, J., Koopman, A: Intelligent traffic light control. Utrecht University: Information and Computing Sciences (2004)
17. Zhang, S., Abdel-Aty, M., Yuan, J., Li, P.: Prediction of pedestrian crossing intentions at intersections based on long short-term memory recurrent neural network. Transportation research record, vol. 2674, no. 4, pp. 57–65 (2020) doi: 10.1177/0361198120912422

ISSN 1870-4069

Modelo de espacio vectorial como método de selección de expresiones gestuales de un agente virtual creíble

Melissa Castillo-Pérez, María Lucila Morales-Rodríguez,
Claudia Gómez-Santillán, Laura Cruz-Reyes, Nelson Rangel-Valdez

Instituto Tecnológico de Ciudad Madero,
México

{G19073014, lucila.mr}@cdmadero.tecnm.mx, {claudia.gomez,
lauracruzreyes, nelson.rangel}@itcm.edu.mx

Resumen. En el desarrollo de agentes virtuales se busca que estos sean creíbles y que el usuario se sienta inmerso en el mundo virtual. En la literatura se describen las características de un agente virtual creíble, por ejemplo, la capacidad de interactuar por medio de la comunicación verbal o no verbal (CNV). La expresión gestual es uno de los componentes de la CNV que permite reforzar el diálogo y su ejecución es influida por la personalidad y el estado emocional. Simular la CNV es complicado, ya que existen diferentes factores que hacen que la interpretación y la presentación del gesto sean variadas. Este artículo presenta una arquitectura para la selección de expresiones gestuales creíbles que ilustren el diálogo de tal forma que se puedan personalizar con el estado emocional y personalidad del agente. Para esto se propone utilizar el modelo de espacio vectorial, el cual se adaptó para permitir el ordenamiento de un conjunto de gestos en función de las palabras clave del diálogo.

Palabras clave: Agente virtual creíble, selección, diálogo, expresión gestual, modelo de espacio vectorial.

Vector Space Model as a Method of Selection of Gestural Expressions of a Believable Agent

Abstract. The aim of developing Believable Agents is to make the user feel immersed in the virtual world. The characteristics of them are described in the literature, for example, the ability to interact through verbal or non-verbal communication (NVC). Gesture expression is a component of NVC that reinforces dialogue and its execution is influenced by personality and emotional state. Simulating NVC is complicated because there are different factors that make interpretation and presentation of the gesture varied. This paper presents an architecture for credible gesture selection that illustrate the dialogue and therefore can be personalized with the agent's emotional state and personality. For this, it is proposed to use the Vector Space Model which was adapted to allow the ordering a set of gestures based on the key words in the dialogue.

Keywords: Believable agent, selection, dialogue, gesture expression, vector space model.

1. Introducción

Un agente virtual es un software que cuenta con cierto grado de autonomía en sus acciones, también es capaz de comunicarse con otros agentes y trabaja en beneficio de un usuario en particular [1]. La aplicación de los agentes virtuales se puede dar en diferentes áreas como los videojuegos, la educación, los servicios en línea, la asistencia médica, etc. lo que permitirá la creación de entornos de simulación que puedan apoyar el aprendizaje. Para ello se busca que el desarrollo de estos agentes sea creíble y así el usuario interactúe con ellos y se abstraiga de la realidad.

Para que un agente logre ser creíble y provoque una sensación de inmersión este debe ser capaz de reproducir el comportamiento del ser humano, debe poder comunicarse de forma verbal y/o no verbal [2]. De acuerdo con [3] es posible crear agentes virtuales creíbles imitando la comunicación no verbal (CNV) considerando la expresión de la emoción y la personalidad. Lo complicado de imitar la CNV del ser humano es que existen diferentes factores que hacen que la interpretación de la CNV sea variada, por ejemplo, algunos de estos factores son el estado de ánimo, la personalidad o el contexto, estos se encuentran en constante cambio por lo que pueden afectar tanto a la interpretación como a la forma en la que los gestos son realizados, esto hace que la tarea de simular la CNV aumenta su dificultad.

Debido a que esta tarea resulta muy extensa, este artículo sólo estará enfocado a la arquitectura de selección de expresiones gestuales en función del diálogo del agente. Esta arquitectura cuenta con una base de gestos caracterizados por palabras clave a fin de utilizar el Modelo de Espacio Vectorial. Éste modelo forma parte de los modelos de Recuperación de Información y son mayormente utilizados para la Búsqueda Web.

Con la aplicación de este modelo se busca ordenar los gestos en función de la relevancia que tienen con relación a las palabras clave del diálogo, una vez ordenados se selecciona el gesto con mayor relevancia. Debido a la manera en la que se caracterizan los gestos es posible adaptar el modelo de espacio vectorial para la selección de expresiones gestuales en función del diálogo.

2. Trabajos relacionados

2.1. Comunicación no verbal en agentes virtuales

En [4] se menciona que a los agentes virtuales que cuentan con apariencia humana se les asocia con capacidades comunicativas, emocionales y sociales, ya que son capaces de utilizar expresiones faciales y gestuales que acompañan el diálogo. Estas características permiten que el uso de agentes virtuales sea de provecho para la interacción hombre máquina, por ejemplo, la CNV en los agentes virtuales puede influir en como son percibidos, apoyando a la creación de entornos de simulación que puedan apoyar al aprendizaje.

En [5] se presenta un estudio donde se buscaba evaluar como la apariencia y el comportamiento no verbal de un agente virtual afectan a la forma en la que el usuario percibe la amabilidad y la competencia del agente, en dicho estudio se determinó que el usuario se ve influenciado por el uso de la CNV para determinar si el agente virtual

es competente, mientras que la apariencia del agente parece ser un factor para determinar la amabilidad del mismo.

Por otra parte, en [6] se estudió el efecto que tiene el comportamiento no verbal dominante y el no dominante de un agente virtual, a este estudio se sometieron participantes jóvenes y participantes adultos, en donde se presentó que los adultos fueron persuadidos por el agente que hacía uso del comportamiento no verbal dominante y generalmente lo evaluaron de manera positiva a comparación de los participantes jóvenes. En ambos casos se puede observar que el uso de la CNV en agentes virtuales ayuda a reflejar la intención del agente e incluso parte de la personalidad del mismo.

2.2 Selección de expresiones corporales en agentes virtuales

Cada agente virtual cuenta con una arquitectura, la cual establece los procesos por los cuales el agente debe pasar para generar una respuesta acorde a la información que obtiene de la interacción con su medio ambiente.

Existen diferentes tipos de arquitecturas pero en todas existen módulos que se encargan de tareas específicas, estos módulos trabajan en conjunto para cumplir el objetivo del agente [7]. Para la arquitectura de un agente virtual creíble es importante contar con un módulo encargado del comportamiento no verbal, por lo general en este módulo se seleccionan los gestos, las posturas y/o las expresiones faciales en función de ciertos factores.

En [8] se presenta una arquitectura que fue desarrollada como un sistema de exposición para un museo donde dos personajes virtuales conversacionales asisten a los visitantes con información sobre los temas referentes al museo. Esta arquitectura permite la construcción de personajes capaces de generar expresiones corporales sincronizadas con el diálogo. La selección de expresiones no verbales se logró a través de reglas simples IF-THEN, también se consideró la posibilidad de que el personaje aprendiera nuevas reglas por medio de clustering.

La arquitectura recibía información del visitante por medio de cámaras y etiquetas, lo que después se interpretan como eventos y dentro del proceso cognitivo del agente se seleccionaba el diálogo según el evento dado, posteriormente una versión del diálogo se enviaba al generador de acciones no verbales para la aplicación de las reglas del gesto y así seleccionar un gesto apropiado para el diálogo.

En [9] también se presenta una arquitectura basada en reglas para la selección de expresiones no verbales, en este caso se analizaron videoclips de personas conversando para la extracción de reglas. La arquitectura cuenta con un módulo para la generación del comportamiento no verbal (NVB Generator), el módulo recibe el mensaje XML de entrada y registra la información afectiva del agente y extrae el texto del diálogo.

Después el diálogo es analizado para obtener la estructura semántica del enunciado y dado el resultado junto con las reglas de generación del comportamiento el módulo selecciona el comportamiento no verbal apropiado, una vez hecho esto, el comportamiento se modifica con la información afectiva del agente. De ambos trabajos se puede resaltar que a pesar de que no en ambos se tomó en cuenta el estado afectivo para la presentación del comportamiento no verbal, si se consideró el diálogo para la selección del gesto.

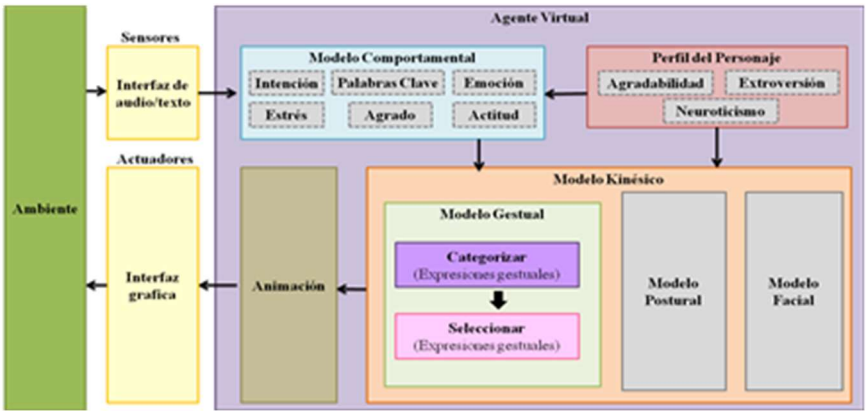


Fig. 1. Arquitectura propuesta para un agente virtual creíble.

En el siguiente punto se propone la utilización del Modelo de Espacio Vectorial para la selección del gesto de un agente virtual en función del diálogo, lo que permitirá que la información necesaria para realizar la selección del gesto se encuentre en la base de conocimientos y no en el código del agente, como sería el caso del uso de reglas.

3. Modelo de espacio vectorial para la selección de expresiones creíbles

3.1. Arquitectura propuesta para un agente virtual creíble

En Fig. 1 se muestra la propuesta de una arquitectura que busca generar agentes virtuales creíbles a través de dotar al agente con la capacidad de poder comunicarse por medio de la expresión no verbal. En esta arquitectura se presenta un módulo encargado de generar el comportamiento no verbal del agente, también cuenta con un módulo encargado de la personalidad y el estado emocional del agente.

El Modelo Comportamental recibirá información del ambiente por medio de la interfaz de audio/texto y después de un proceso cognitivo este generará los atributos mostrados en la Fig. 1; el Perfil del Personaje se encargará de almacenar los rasgos importantes de la personalidad del agente. Estos dos componentes proveerán de información al Modelo Kinésico, el cual se encargará de tomar los atributos antes mencionados para general una respuesta no verbal.

Este trabajo se centra en el Modelo Kinésico, el cual está integrado por 3 modelos que se encargan de seleccionar la comunicación no verbal. El Modelo Gestual seleccionará los movimientos hechos por las manos, el Modelo Postural seleccionará la postura del cuerpo acorde al estado emocional y personalidad del agente y el Modelo Facial seleccionará la expresión facial correspondiente al estado emocional y la personalidad del agente. El Modelo Gestual y el Modelo Postural trabajan con las mismas partes del cuerpo (manos y brazos) lo que podría provocar que la ejecución del gesto se vea afectada por la postura del agente. Por lo que es necesario se considere la

implementación de un modelo de integración. De los procesos que se realizarán en el Modelo Gestual el más importante es la selección del gesto, para esto se debe tener un conjunto de *gestos caracterizados por sus palabras clave*.

El Modelo Gestual recibirá un conjunto de palabras clave extraídas del acto ilocutivo que se quiere comunicar a través del dialogo, el cual será la consulta y se deberá buscar dentro del conjunto de gestos ya conocidos si existe alguno que comparta similitud con la consulta. Para esto se propone utilizar el Modelo de Espacio Vectorial, ya que permitirá ordenar el conjunto de gestos considerando la similitud de cada gesto con el conjunto de palabras clave.

3.2. Modelo de espacio vectorial para la selección del gesto

Los modelos de Recuperación de Información (IR) buscan reducir un conjunto de documentos en función de la relevancia que tengan con la consulta dada. Estos modelos se suelen utilizar en el área de Búsqueda Web, donde el método ayuda al usuario a encontrar la información necesaria dentro de un conjunto grande de documentos de texto [10]. Dentro de los modelos IR se encuentran el Modelo Booleano (Boolean Model), el Modelo de Espacio Vectorial (Vector Space Model) y el Modelo de Lenguaje (Language Model), cada modelo representa de manera diferente los documentos y las consultas, pero siguen una misma estructura, por ejemplo:

- Cada documento o consulta son tratados como un conjunto de palabras o términos.
- Un término es simplemente una palabra cuya semántica sólo ayuda a recordar los temas principales del documento.
- La secuencia de los términos en una oración o en un documento no se toma en cuenta.
- Existe una colección de documentos D, donde cada documento dj se representa con un vector de términos: dj= {w1j, w2j…, w|V|j}.
- A cada término se le asocia un peso. Un peso wij>0 es asociado a cada término ti de un documento dj que pertenezca a D. Para cada término que no aparezca en el documento dj, wij=0.
- En cada modelo el peso (w_{ij}) se calcula de manera diferente.
- Se tiene un vocabulario V que es el conjunto de distintos términos de la colección, $V= \{t_1, t_{2...}, t_{|V|}\}$ donde $|V|$ es el número de términos en V.

El propósito de este artículo es trabajar con el Modelo de Espacio Vectorial para la selección de gestos, por lo que a continuación se describirán los elementos principales para su implementación.

Como primer punto, se debe conocer cómo se interpreta un documento en este modelo. Un documento se representa como un vector de pesos, en donde el peso de cada componente se calcula en función de alguna variación del esquema de Frecuencia de Término (TF) o del esquema de Frecuencia de Término y Frecuencia Inversa del Documento (TF-IDF). Para el primer esquema, el peso de un término t_i en el documento d_j corresponde al número de veces que t_i aparece en d_j y se indica con f_{ij}. Para calcular

Tabla 1. Caracterización de la colección de gestos en base a 6 palabras clave e intención.

Alto	Bajo	Persona	Personas	Derecha	Objeto	Intención	ID
1	0	0	0	0	0	200	G1
0	1	0	0	0	0	200	G2
0	0	1	1	1	0	100	G3
0	0	0	0	1	1	100	G4

Tabla 2. Caracterización del acto del habla que se quiere expresar.

Alto	Bajo	Persona	Personas	Derecha	Objeto	Intención
0	0	1	0	1	0	100

el esquema TF-IDF existen variaciones pero en [10] describen el más básico, el cual está dado por la ecuación 1:

$$w_{ij} = tf_{ij} \times idf_i \tag{1}$$

donde tf_{ij} es la frecuencia del término t_i en el documento d_j y se calcula por la ecuación 2. Sea f_{ij} la frecuencia del término t_i en el documento d_j y el máximo se calcula sobre todos los términos que aparecen en d_j. Si t_i no aparece en d_j, entonces $tf_{ij}=0$.:

$$tf_{ij} = \frac{f_{ij}}{\max\{f_1, f_2, \dots, f_{|V|j}\}} \tag{2}$$

Por otro lado, idf_i es la frecuencia inversa del término t_i está dada por la ecuación

$$idf_i = log\frac{N}{df_i} \tag{3}$$

,

donde N es el número total de documentos de la colección y df_i el número de documentos en los que el termino t_i aparece al menos una vez.

La consulta q se representa de la misma manera que un documento de la colección. Para calcular el peso de cada término t_i en q se usará la ecuación 4 utilizada en [11]. Donde f_{iq} es la frecuencia del término t_i en la consulta, idf_i es la frecuencia inversa del término t_i, la frecuencia máxima en la consulta:

$$w_{iq} = \left(\frac{f_{iq}}{\max(f_{1q}, f_{2q}, \dots, f_{|V|q})}\right) \times idf_i \tag{4}$$

Una de las formas de calcular el grado de relevancia es calculando la similitud de la consulta q con cada documento d_j en la colección de documentos D. Existen varias medidas de similitud, la más conocida es la similitud de coseno, que es el coseno del ángulo entre el vector de la consulta q y el vector del documento d_j. Para calcular el coseno de similitud entre d_j y q se debe seguir la ecuación 5:

$$coseno(d_j, q) = \frac{\sum_{i=1}^{|V|} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{|V|} w_{ij}^2} \times \sqrt{\sum_{i=1}^{|V|} w_{iq}^2}} \tag{5}$$

Una vez presentados los componentes del Modelo de Espacio Vectorial se describe la información que pasará a ser la colección de documentos y la consulta en el modelo. Para representar la colección de documentos se utilizará un conjunto de gestos recopilados de [12] y de [13], donde se seleccionaron 53 gestos en total, para la selección se tomó en cuenta que el gesto fuera un gesto icónico o un gesto deíctico. De estos gestos se consideraron las palabras clave, la intención, las partes del cuerpo que lo configuran.

Para conformar el conjunto de gestos con el que trabajará el modelo se tomaron todas las palabras claves que aparecen en todos los gestos y cada una se consideró como un atributo en una tabla, seguidas de la intención del gesto y un identificador para cada gesto (ID), cada renglón de la tabla representa un gesto y las celdas de las palabras clave se marcan con un 1 si la palabra clave se encuentra en el gesto y un 0 en caso contrario; en la celda de la intención se toman en cuenta dos valores, si el gesto es deíctico su intención es indicar por lo que se le asigna un valor de 100 y en caso de ser un gesto icónico su intención es ilustrar por lo que se le asigna un valor de 200.

En la tabla 1 se puede observar un ejemplo de cómo sería la colección de gestos. Los gestos que se presentan en la tabla son algunos de los que estarán presentes en la base de conocimientos del Modelo Gestual. Para poder seleccionar un gesto de esa base de conocimientos el modelo necesitará una consulta, la cual recibirá del Modelo Comportamental. Esta consulta estará caracterizada por las palabras clave del diálogo del agente y la intención del diálogo.

En la tabla 2 se muestra un ejemplo de lo que recibirá el modelo gestual. Teniendo caracterizado el conjunto de gestos y la consulta el primer paso es filtrar el conjunto de gesto por la intención dada en la consulta, esto con el fin de que al momento de aplicar el modelo de espacio vectorial solo se tomen en cuenta los gestos que coincidan con la intención que se está solicitando, una vez filtrado se puede aplicar el modelo para calcular la similitud que hay entre los gestos y la consulta. Al finalizar, los gestos podrán ordenarse de mayor a menor con relación a la similitud obtenida, de esta manera se podrá seleccionar el gesto que se encuentre en primer lugar. Este proceso se ilustra en la Fig. 2.

3.3. Experimentación y resultados

Se evaluó el desempeño del método descrito con anterioridad para seleccionar un gesto considerando las palabras clave del diálogo del agente. Para la evaluación se generaron 18 consultas diferentes. Cada consulta representa a las palabras clave del diálogo que se desea expresar y considerando el conjunto de gestos que se encuentra en la base de conocimiento del agente, se le asignó a cada consulta un gesto esperado.

La definición del gesto esperado fue dada por las palabras clave de la consulta, dentro del conjunto de gestos se buscó el ID de los gestos que tuvieran asociadas las palabras clave de dicha consulta y él o los gestos encontrados pasarían a ser el gesto o gestos esperados.

Posteriormente a cada consulta se le asignó un gesto utilizando el Modelo de Espacio Vectorial, para después comparar los gestos asignados contra los gestos esperados de las consultas.

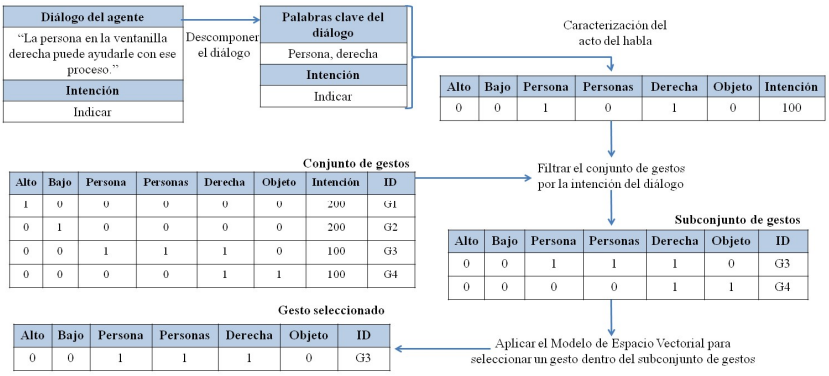


Fig. 2. Proceso de selección gestual.

Tabla 3. Comparación de resultados de la experimentación.

Consulta	Gesto esperado	Gesto Asignado	Consulta	Gesto esperado	Gesto Asignado
1	G6	G6	10	G9	G9
2	G5	G5	11	G40	G40
3	G40	G40	12	G24	G24
4	G12	G12	13	G26	G26
5	G51	G51	14	G49	G49
6	G8	G8	15	G21	G21
7	G25	G25	16	G7	G7
8	G2	G2	17	G6	G30
9	G12	G15	18	G1	G1

En la tabla 3 se muestra dicha comparación, se presenta el número de consulta, el ID del gesto esperado y el ID del gesto asignado. El modelo acertó un total de 16 asignaciones de gestos, obteniendo un 89% de eficacia.

4. Conclusiones y trabajo a futuro

En resumen, el hecho de que un agente virtual sea capaz de expresarse por medio de la CNV contribuye a que se le considere creíble y sea de gran apoyo para mejorar la comunicación hombre-máquina, debido a esto, en este artículo se propone seleccionar el comportamiento no verbal de un agente virtual a través del Modelo de Espacio Vectorial en función del diálogo.

A pesar de que este modelo pertenece a los modelos de Recuperación de Información y es mayormente utilizado para el ámbito de Búsqueda Web, en este artículo se ha

demostrado que también se puede emplear para la selección de expresiones gestuales en agentes virtuales debido a la manera en la que se han caracterizado los gestos.

Los gestos se caracterizaron por medio de las palabras clave que lo representan, las cuales pasarían a ser los términos del documento en el modelo de espacio vectorial, de esta manera fue posible ajustar el modelo a la selección del gesto. El modelo permitió ordenar el conjunto de gestos en función de la similitud que tenían con las palabras clave del diálogo, debido a esto se seleccionó, en su mayoría, el gesto esperado a las consultas de prueba. A pesar de que el modelo presento un buen desempeño, el conjunto de gestos y el conjunto de consultas se pueden considerar como conjuntos pequeños, por esto se planea robustecer ambos conjuntos y evaluar que el rendimiento hasta ahora presentado no decaiga.

Por otra parte, la CNV no sólo está integrada por las intenciones del diálogo, también se necesita que el agente sea capaz de expresar personalidad y emoción. Por lo que en el futuro se planea modificar la forma en la que el gesto es ejecutado o presentado considerando el estado emocional y el perfil de personalidad del agente. Para esta tarea se considera adaptar las 6 dimensiones del movimiento que se estudian en la literatura [14, 15, 16, 17], las cuales buscan describir el movimiento a través de características como la amplitud, la velocidad, etc.

Estas dimensiones del movimiento se caracterizarán por medio de atributos emocionales y los rasgos de personalidad para determinar las dimensiones del movimiento que le corresponden a una cierta emoción y personalidad, lo que ayudaría al proceso de personalizar el gesto seleccionado. Por ejemplo, si a una emoción "x" le corresponde una amplitud del movimiento larga, el gesto seleccionado deberá ser amplio (despegando él o los brazos del trozo). Permitiendo que el comportamiento no verbal generado por el agente sea creíble.

Agradecimientos. El autor, con el CVU 1006593, agradece al CONACyT el apoyo otorgado a través de la Beca para Estudios de Posgrado.

Referencias

1. Nwana, H. S.: Software agents: An overview. Knowl. Eng. Rev., vol. 11, no. 3, pp. 205–244 (1996) doi: 10.1017/S026988890000789X
2. Loyall, A. B.: Believable agents: Building interactive personalities: Doctoral Thesis in the field of Computer Science. School of Computer Science Computer Science Department Carnegie Mellon University, Pittsburgh (1997) https://apps.dtic.mil/dtic/tr/fulltext/u2/a327862.pdf
3. Rulicki, S., Cherny, M.: Comunicación no verbal - CNV: Cómo la inteligencia emocional se expresa a través de los gestos. 1era Edición. Argentina: Ediciones Granica (2007)
4. Andre, E., Pelachaud, C.: Interacting with embodied conversational agents. In Speech Technology: Theory and Applications, In: Chen, F., Jokinen, K. (eds) Speech Technology. Springer, New York, NY. pp. 123–149 (2010) doi: 10.1007/978-0-387-73819-2_8
5. Bergmann, K., Eyssel, F., Kopp, S.: A second chance to make a first impression? How appearance and nonverbal behavior affect perceived warmth and competence of virtual agents over time. In: Intelligent Virtual Agents, Berlin, Heidelberg, pp. 126–138 (2012) doi: 10.1007/978-3-642-33197-8_13
6. Rosenthal-von der Pütten, A. M., Straßmann, C., Yaghoubzadeh, R., Kopp, S., Krämer, N. C.: Dominant and submissive nonverbal behavior of virtual agents and its effects on

evaluation and negotiation outcome in different age groups. Comput. Hum. Behav., vol. 90, pp. 397–409 (2019) doi: 10.1016/j.chb.2018.08.047
7. Corchado-Rodríguez, J.: Modelos y arquitecturas de agente. Agentes Software y Sistemas Multiagentes, Pearson Educación, pp. 29-64 (2005)
8. Kipp, M.: Creativity meets automation: combining nonverbal action authoring with rules and machine learning. Intelligent Virtual Agents, vol. 4133, J. Gratch, M. Young, R. Aylett, D. Ballin, P. Olivier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 230–242 (2006) doi: 10.1007/1 1821830_19
9. Lee, J., Marsella, S.: Nonverbal behavior generator for embodied conversational agents. In: Intelligent Virtual Agents, vol. 4133, J. Gratch, M. Young, R. Aylett, D. Ballin, y P. Olivier, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 243–255 (2006) doi: 10.1007/1 1821830_20
10. Bing, L.: Web data mining: Exploring hyperlinks, contents and usage data. 2da Edición. United States of America: Springer (2011)
11. Hernández-González, L. J.: Implementación de un prototipo de un sistema de recuperación de información que utilice antologías para la expansión de consultas. Instituto Tecnológico de Cd Madero, Cd. Madero, Tamaulipas (2016) http://200.188.131.162:8080/jspui/han dle/1234 56789/360
12. Gaviño-Rodríguez, V.: Español coloquial | Colloquial spanish. Coloquial.es. Español Coloquial (2010) https://www.coloquial.es/es/
13. Pérez, F.: Diccionario de gestos dominicanos. Advanced Reasoning Forum (2018)
14. Pelachaud, C.: Studies on gesture expressivity for a virtual agent. Speech Commun, vol. 51, no 7, pp. 630–639 (2009) doi: 10.1016/j.specom.2008.04.009
15. Mancini, M., Castellano, G.: Real-time analysis and synthesis of emotional gesture expressivity (2007)
16. Wallbott, H. G.: Bodily expression of emotion. Eur. J. Soc. Psychol., vol. 28, no. 6, pp. 879–896 (1998) doi: 10.1002/(SICI)1099-0992(1998110)28:6<879:AID-EJSP901 >3.0.CO;2-W
17. Hartmann, B., Mancini, M., Pelachaud, C.: Implementing expressive gesture synthesis for embodied conversational agents. pp. 199 (2005) doi: 10.1007/11678816_22

ISSN 1870-4069

Recuperación de escenarios naturales por contenido por medio de la causalidad de Wiener-Granger: metodología auto-organizante

Cesar Benavides-Álvarez, Carlos Avilés-Cruz, Arturo Zúñiga-López, Andrés Ferreyra-Ramírez, Eduardo Rodríguez-Martínez

Universidad Autónoma Metropolitana,
Departamento de Electrónica, División de Ciencias Básicas e Ingeniería,
México

cesarbenavides32@gmail.com,
{caviles, azl, fra, erm}@azc.uam.mx

Resumen. La organización, clasificación y recuperación de imágenes digitales son hoy en día una de las áreas más aplicadas dentro de la ciencia de datos, minería de datos o búsqueda de información en internet. Por otro lado, la búsqueda de imágenes por su contenido o una zona de la misma, es el interés actual en la web. En el presente artículo se hace uso de la teoría de causalidad de Wiener-Granger, en un sistema auto-organizante de escenarios naturales. La metodología propuesta comprende una etapa de extracción de características en puntos aleatorios dentro de la imagen, después estas características son organizadas en forma de series temporales posteriormente, se realiza la estimación de causalidad Wiener-Granger. Una vez obtenidas las causalidades, se aplica el algoritmo k-means para lograr la auto-organización de atributos. En cuanto a la clasificación, se hace uso del algoritmo de clasificación de distancia k-NN para encontrar las imágenes más parecida que comparta las relaciones causales entre los elementos de los escenarios. Nuestra metodología se valida con 2 bases de imágenes públicas, obteniendo resultados de 100 % de recuperación.

Palabras clave: Clasificación de imágenes, recuperación de imágenes por contenido, causalidad de Wiener-Granger.

Content-based Natural Scenario Retrieval by Wiener Granger Causality: Self-Organizing Methodology

Abstract. The organization, classification, and retrieval of digital images are nowadays one of the most applied areas within data science, data mining, or information search on the Internet. On the other hand, the search of images by their content or an area of it is the current interest in the web. In this paper, we make use of the Wiener-Granger causality theory in a self-organizing system of natural scenarios. The proposed methodology comprises a stage of feature extraction at random points within the image, then these features are organized in the form of time series, and Wiener-Granger causality estimation is

performed. Once the causalities are obtained, the k-means algorithm is applied to achieve attribute self-organization. As for classification, the k-NN distance classification algorithm is used to find the most similar images that share the causal relationships between the elements of the scenarios. Our methodology is validated with two public image bases, obtaining results of 100%.

Keywords: Classification, content-based image retrieval, image retrieval, image classification, Wiener-Granger causality.

1. Introducción

Con el desarrollo del Internet y el uso masivo de dispositivos digitales, la recuperación de imágenes basada en su contenido (CBIR) ha tenido importante desarrollado y muchas aplicaciones, particularmente, a la visión artificial e inteligencia artificial [20].

Actualmente, se han realizado avances en nuevas teorías y modelos de CBIR y se han establecido muchos algoritmos CBIR eficaces que permiten buscar y recuperar imágenes (por su contenido) a partir de una imagen de entrada; entre los campos de aplicación se tiene: moda, identificación de personas, recuperación de productos de comercio electrónico, recuperación de imágenes de tele-detección, recuperación de imágenes de marcas, recuperación de escenarios naturales, entre otros [15, 12].

En el campo de la visión por computadora o visión artificial, lo que se busca es emular el sistema visual humano, con las capacidades y habilidades como lo hacen nuestros ojos. Así, el objetivo que persigue la visión artificial, es dotar a las computadoras de nuestras capacidades visuales de un mundo tridimensional, partiendo, generalmente, de imágenes bi-dimensionales [2].

Dado que no existe un algoritmo eficaz que pueda reconocer completamente cualquier objeto o escenario, la visión por computadora se considera un problema abierto, en donde se conjugan áreas como el procesamiento digital de imágenes, reconocimiento de patrones, aprendizaje de máquina, etc.

Una de las tareas de los sistemas automatizados de reconocimiento de imágenes es clasificar e identificar con éxito las imágenes de paisajes naturales[1], que con el aumento exponencial en la generación de imágenes de escenarios naturales ha tenido la web. Se estima que, más de la mitad de la información en Internet son imágenes, de las cuales el 85 % fueron tomadas a través de dispositivos móviles, y se consideró una estimación final de 5 billones de imágenes hasta este el año 2018 [22].

Para utilizar esta información de manera eficiente, es necesario un sistema CBIR, ayudando a los usuarios a encontrar imágenes relevantes en función de sus características de auto-contenido o de las que se "ven" relacionadas con ellas, a partir de nuestra percepción visual, incluso cuando no existe un conocimiento previo de la base de datos, como el etiquetado manual sobre las imágenes.

[1] Se dice que un paisaje es natural si la imagen no tiene ninguna intervención o alteración de la mano del hombre.

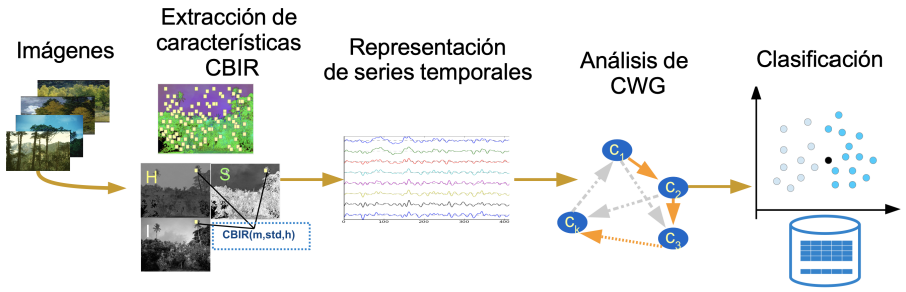


Fig. 1. Metodologia general propuesta.

En este trabajo se desarrolla un sistema de recuperación de imágenes de paisajes naturales aplicando la teoría de la causalidad de Wiener-Granger (WGC) [11] como herramienta de análisis de imágenes a través de la información auto-organizada. Se identificaron las relaciones de causalidad entre las texturas locales contenidas en una imagen, lo que lleva a caracterizar un patrón descriptivo de un conjunto de paisajes dentro de un conjunto de datos de imágenes. Las principales etapas involucradas en el sistema desarrollado son las siguientes (Ver Fig. 1):

1. Lectura de imágenes: En primer lugar, se leen las imágenes del conjunto de datos y, a continuación, se aplica un cambio de formato de color del espacio Rojo-Verde-Azul RGB a Tono-Saturación-Intensidad HSB.
2. Extracción de características: La extracción estadística de características CBIR se genera aleatoriamente sobre 300 puntos de la imagen.
3. Conformación de series temporales: Las características de textura se organizan como una serie temporal para cada imagen.
4. Análisis de causalidad: El análisis WGC se aplica para calcular la matriz de relación causal entre las diferentes texturas.
5. Aplicación del clasificación Se usa el algoritmos de clasificación a k-próximos vecinos KNN, para encontrar las texturas más próximos o similares a la de búsqueda.

El trabajo propone un análisis de causalidad de las clases de paisaje natural basado en un diccionario de texturas preestablecido y el análisis WGC de la metodología CBIR [21, 7] para proporcionar la caracterización del conjunto de datos. La metodología actual se probó con dos bases de datos de paisajes naturales:

— Vogel y Shiele (V_S) [23]. con 700 Imágenes clasificadas como: 144 costa, 103 bosque, 179 montaña, 131 pradera, 111 río/lago y 32 cielo/nube.

— Oliva y Torralba (O_T) [18]. 1472 Imágenes clasificadas como: 360 costa, 328 bosque, 374 montaña y 410 pradera.

Finalmente, se reporta una mejora en la precisión de la clasificación obtenida por nuestra estrategia propuesta, obteniendo un 100 % en la re-sustitución y hasta un 96 % para las metodologías de validación cruzada.

Visualizando en el futuro implementar un sistema autónomo de reconocimiento de escenas naturales montado en un coche, que será gestionado por nuestra propuesta como sistema autónomo [4, 5, 1]. El hecho de reconocer escenarios naturales en la navegación de un coche autónomo o posiblemente un dron, con un 100 % de certeza, este sistema propuesto será un elemento importante en la seguridad de los vehículos autónomos.

El resto del trabajo se organiza como sigue. En la sección 2 se presenta el soporte teórico del modelo WGC a aplicar. En la sección 3, se presenta la metodología. Los resultados se presentan en la sección 4. Finalmente, las conclusiones y perspectivas se presentan en la sección 5.

2. Fundamentos de la teoría de causalidad de Wiener-Granger

El paradigma de la inferencia causal se ha utilizado en diferentes campos de la ciencia, por ejemplo, en neurología se utiliza la teoría WGC [6] para examinar áreas del cerebro y las relaciones causales entre ellas. El análisis de la WGC se realizó mediante sensores [17, 16], y, últimamente en imágenes de resonancia magnética [26, 10, 13], la teoría de la WGC se está utilizando para el estudio de las relaciones causales entre las áreas que realizan actividades del cerebro.

Otros campos de la ciencia donde se ha aplicado la teoría WGC es en el procesamiento de vídeo para la indexación y recuperación [8]. El procesamiento de vídeo para la identificación masiva de personas y vehículos [14, 19, 25] y el análisis de escenarios complejos [9]. En esta propuesta, por primera vez, se aplica la teoría WGC a la recuperación de elementos naturales y escenas naturales.

En esta sección se establece el marco teórico del WGC. Por simplicidad y para evitar extendernos matemáticamente, la teoría se presenta sólo para dos procesos aleatorios, siendo extensible a $n-$procesos. En nuestro enfoque, un proceso aleatorio corresponde a una lectura de señal asociada a un tipo de textura dentro de un escenario natural. Así, para el presente análisis, cada lectura de textura corresponde a un proceso estocástico representado por Ci, siendo i la $i-$ésmia textura que tiene un comportamiento estocástico dentro de un escenario.

2.1. Stochastic autoregressive model

Asumimos que cada textura puede ser representada por un modelo autorregresivo en series temporales. En el presente análisis y por simplicidad didáctica, sólo lo llevaremos a cabo con dos señales, $\{C1$ y $C2\}$, siendo fácilmente extensible a n señales-texturas. Cada proceso estacionario representa una textura del escenario y, puede representarse mediante un modelo autorregresivo de la siguiente manera:

$$C1(t) = \sum_{k=1}^{\infty} K_{C1}^{1}(k) \cdot C1(t-k) + \eta_{C1}^{1}, \qquad \text{con} \qquad \sum_{C1}^{1} = \text{var}(\eta_{C1}^{1}), \qquad (1)$$

$$C2(t) = \sum_{k=1}^{\infty} K_{C2}^{1}(k) \cdot C2(t-k) + \eta_{C2}^{1}, \qquad \text{con} \qquad \sum_{C2}^{1} = \text{var}(\eta_{C2}^{1}), \qquad (2)$$

siendo η^1_{C1} y η^1_{C2} ruido aleatorio Gaussian a valor media cero y desviación estándar unitaria; $K^1_{C1}(k)$ y $K^1_{C2}(k)$ son los coeficientes del modelo de regresión para la texturas $C1$ y $C2$, respectivamente. El modelo autorregresivo conjunto para las dos texturas está definido por las ecuaciones 3 y 4:

$$C1(t) = \sum_{k=1}^{\infty} K^{1,1}_{C1}(k) \cdot C1(t-k) + \sum_{k=1}^{\infty} K^{1,2}_{C2}(k) \cdot C2(t-k) + \eta^2_{C1}, \quad \text{con } \sum\nolimits^2_{C1} = \text{var}(\eta^2_{C1}), \quad (3)$$

$$C2(t) = \sum_{k=1}^{\infty} K^{2,1}_{T1}(k) \cdot C1(t-k) + \sum_{k=1}^{\infty} K^{2,2}_{C2}(k) \cdot C2(t-k) + \eta^2_{C2}, \quad \text{con } \sum\nolimits^2_{C2} = \text{var}(\eta^2_{C2}), \quad (4)$$

donde $\sum^2_{C1}$ y $\sum^2_{C2}$ son las varianzas de los términos residuales; η^2_{C1} y η^2_{C2}, respectivamente. Por otro lado, los términos $K^{i,j}_{Cl}(k)\ \forall i, j, l \in [1, 2]$, son los coeficientes de la regresión para las regiones texturadas $C1(t)$ and $C2(t)$, respectivamente. Ahora analicemos las varianzas/covarianzas de los términos residuales η^2_{Ci} mediante la siguiente ecuación de forma matricial Σ (5):

$$\Sigma = \begin{pmatrix} \Sigma^2_{C1} & \Upsilon_{1,2} \\ \Upsilon_{2,1} & \Sigma^2_{C2} \end{pmatrix}, \quad (5)$$

donde $\Upsilon_{1,2}$ es la covarianza entre η^2_{C1} and η^2_{C2}, definida como $\Upsilon_{1,2} = \text{cov}(\eta^2_{C1}, \eta^2_{C2})$. A partir de las condiciones anteriores, y utilizando el concepto de independencia estadística entre dos procesos aleatorios al mismo tiempo (por parejas), se puede definir la causalidad en el tiempo. Un ejemplo de la causalidad entre $C1$ y $C2$ es la expresión dada por la ecuación 6:

$$F_{C2,\, C1} = \ln \left[\frac{\Sigma^1_{C1} \times \Sigma^1_{C2}}{\Sigma^2_{C1} \times \Sigma^2_{C2}} \right]. \quad (6)$$

La ecuación 6 se conoce comúnmente como la causalidad en el dominio del tiempo. A partir de esta ecuación, si los procesos aleatorios $C1(t)$ y $C2(t)$ son estadísticamente independientes, entonces $F_{C1,C2} = 0$; en caso contrario habrá causalidad de uno a otro. En la ecuación (1), Σ^1_{C1} mide la precisión del modelo autorregresivo para predecir $C1(t)$, establecido en las muestras pasadas.

A su vez, Σ^2_{C1} en la expresión 3 mide la precisión para predecir $C1(t)$ basándose en los valores anteriores de $C1(t)$ y $C2(t)$ al mismo tiempo. Volviendo al caso de tomar sólo 2 texturas al mismo tiempo $C1(t)$ y $C2(t)$ y según [24] y [11], si $\Sigma^2_{C2} < \Sigma^1_{C1}$ entonces se dice que $C2(t)$ tiene una influencia causal sobre $C1(t)$. La causalidad se define por la ecuación 7:

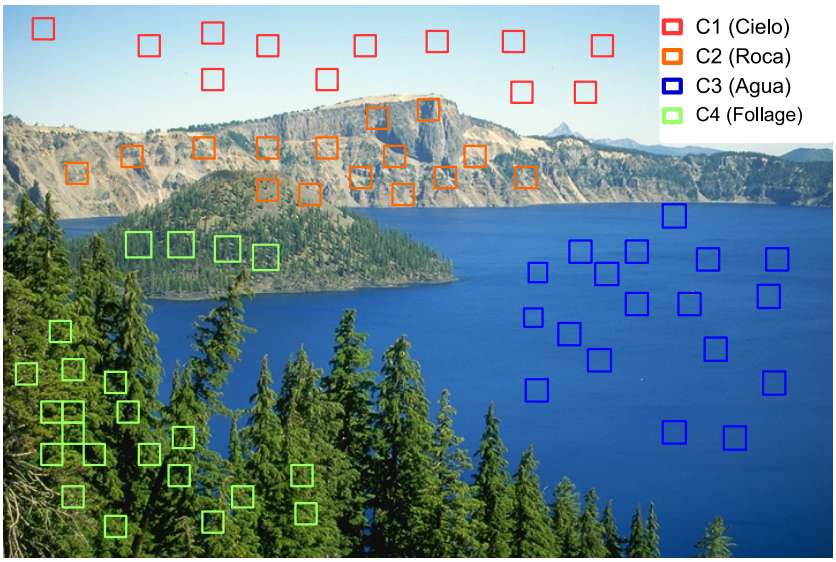


Fig. 2. Ejemplo de una segmentación de zonas texturadas en un escenario natural para k=4 en el algoritmo k-Medias.

$$F_{C2\to C1} = \ln\left[\frac{\Sigma^1_{C1}}{\Sigma^2_{C1}}\right], \tag{7}$$

Es relativamente fácil ver que si $F_{C2\to C1} = 0$ entonces no hay influencia causal de $C2(t)$ hacia $C1(t)$, en cualquier otro valor, el resultado será diferente de cero. Por otro lado, la influencia causal de $C1(t)$ hacia $C2(t)$ se establece mediante la ecuación 8:

$$F_{C1\to C2} = \ln\left[\frac{\Sigma^1_{C2}}{\Sigma^2_{C2}}\right]. \tag{8}$$

3. Metodología

En esta sección describimos la metodología diseñada para el sistema de auto organización de imágenes naturales usando la CWG. Para el uso del CBIR existen diferentes factores determinantes que se deben tener en cuenta a la hora de extraer la información de las imágenes, como la luminosidad, orientación, escala, homogeneidad, etc. Por lo que, la característica principal en los patrones propuestos de cada escena natural, es la textura.

La metodología propuesta, para la etapa de entrenamiento, se presenta en la Figura 3. Comprende desde la lectura de la base de imágenes, el cambio de formato de color, el sembrado de puntos aleatorios, la extracción de atributos, el auto-agrupamiento, la generación de series temporales, el cálculo de causalidad, la clasificación-recuperación de las k-imágenes próximas, y finalmente, el desplegado de las escenas naturales más parecidas.

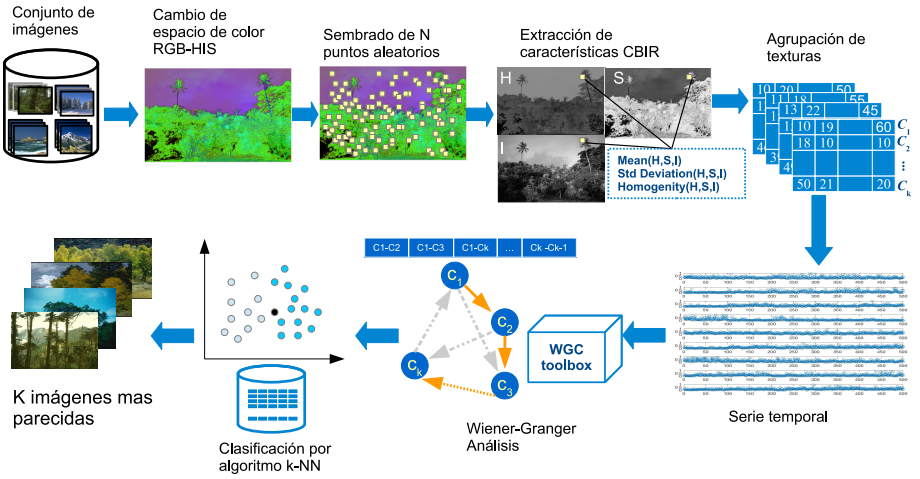


Fig. 3. Arquitectura de entrenamiento y prueba para el sistema de clasificación.

La hipóstesis principal de la metodología propuesta es la generación de un diccionario automático de texturas, las cuales representan elementos propios contenidos en las escenas como son, agua, vegetación, nubes, rocas, arena, etc. (ver Figura 2), depende de la auto-organización de la información por medio del algoritmo de agrupamiento k-means. A continuación se describe en detalle cada bloque comprendido en la metodología.

1. **BD.** Contiene el conjunto de imágenes de escenarios naturales.
2. **Preprocesamiento.** Carga la imagen de la base de datos, se realiza un ecualizado al histograma en las tres capas y finalmente se pasa del espacio de color RGB al HSB (Hue, Saturation, Brightness – Matiz, Saturación, Brillo), el cual proporciona la información relacionada a la textura.
3. **Sembrado de puntos aleatorios**. Se hace un sembrado aleatorio uniforme de 300 puntos aleatorios dentro de la imagen.
4. **Extracción de características.** Esta sección se realiza en dos partes principales y para tener varias muestras de la imagen se repite el proceso un numero r de veces:
   — **Generalización de la vecindad.** Para cada puto aleatorio se crea una ventana de tamaño $p \times p$ pixeles comenzando con el punto de interés en la esquina superior derecha, como se muestra en la Figura 4, en donde $p < \text{image\_rows}$ y $p < \text{imagte\_colums}$.
   — **CBIR feature extraction.** Para cada vecindad se extraen tres características de textura como son, la media, la desviación estándar y la homogeneidad en las tres capas del espacio de color HSB. Esto da como resultado un vector de 1×9 características de textura, para una imagen se crea una matriz de tamaño $N_P \times 9$. Donde N_P es el número de puntos aleatorios y F_{CBIR} son las características CBIR extraídas.

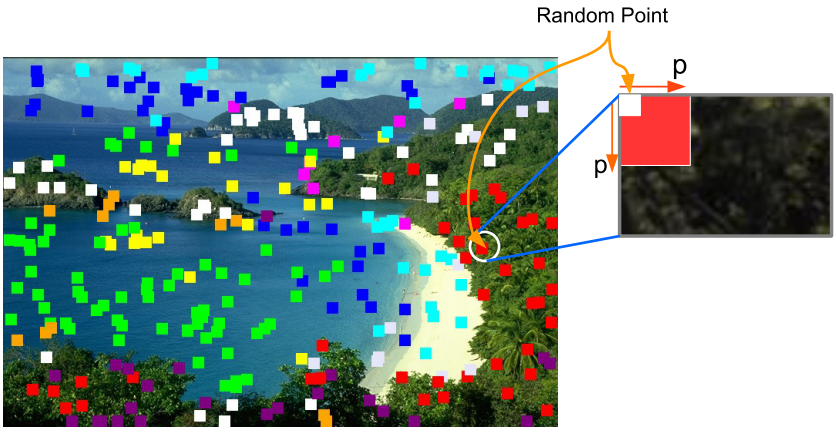


Fig. 4. Imagen con puntos aleatorios, cada punto aleatorio genera una vecindad de tamaño 10×10 pixeles.

5. **Agrupación de texturas CBIR.** Una vez que se extraen las características de textura CBIR de cada imagen en la base datos, se genera una matriz F_{CBIR} de tamaño $(r * N_I * N_P) \times 9$, donde r es el número de repeticiones, N_I es el número de imágenes en la base de datos, N_P es el número de puntos aleatorios y finalmente, F_{CBIR} las características CBIR en cada punto para cada capa de color HSB.

   Por medio del algoritmo K-means se agrupan las características CBIR en k clusters, estos representan las k texturas más representativas de la base de datos, generando una matriz CM_k.

6. **Generación de series de tiempo.** De la matriz F_{CBIR} del paso anterior, cada entrada de la matriz se compara con las entradas de la matriz MC_k del diccionario automático para construir una señal discreta como serie temporal T_S la cual tiene un tamaño de $k \times r \times N_I$. Donde k es el numero de texturas automáticas del algoritmo k-means, r el número de repeticiones del experimento y N_I el número de imágenes en la base de datos.

7. **Análisis de causalidad de Wiener-Granger.** Una entrada de la matriz T_S tiene un tamaño de $k \times r$ para cada imagen en la base de datos. Esta entrada será la que alimenta el análisis de CWG, como se ve en la Figura 5. El análisis de causalidad se calculó con el toolbox de causalidad MVGC [3].

   Una vez realizado el análisis de causalidad para cada una de las imágenes en la base de datos, obtenemos una matriz de relaciones de causalidad η_I de tamaño $k \times k$. La matriz F_{C_i,C_j} representa la matriz de relaciones causales para cada imagen I, en donde $C_i \rightarrow C_j$ representan las relaciones causales de las k variables entre ellas mismas, (como se ve en la Ecuación (9), de forma que si un valor de $F_{C_i,C_j} = 0$ significa que no hay relación causal de la textura $i \rightarrow j$, y en la medida en que el valor aumenta con respecto a otros valores η_I, decimos que la relación causal es significativa con respecto a otras:

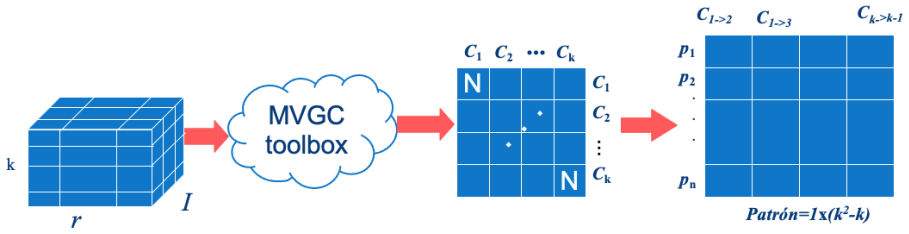


Fig. 5. Generación de la matriz de relaciones causales, η_I, usando el análisis de CWG.

$$\eta_I = \begin{bmatrix} F_{C_1,C_1} & F_{C_1,C_2} & \dots & F_{C_1,C_k} \\ F_{C_2,C_1} & F_{C_2,C_2} & \dots & F_{C_2,C_k} \\ \vdots & \vdots & \vdots & \vdots \\ F_{C_k,C_1} & F_{C_k,C_2} & \dots & F_{C_k,C_k} \end{bmatrix}. \tag{9}$$

Las matrices de causalidad η_I se convierten en vector concatenando sus renglones, en el mismo paso se eliminan los elementos de la diagonal principal debido a que no existe relación causal de una variable con ella misma, como se observa en la teoría. Cada elemento convertido en vector es ahora un patrón representativo de la imagen en la base de datos, dentro de una matriz a la que se le da el nombre de Γ, de tamaño $N_I \times (k * k) - k$, donde N_I es el numero de imágenes, k el número de texturas automáticas del diccionario.

8. Finalmente se realiza una nueva agrupación de la matriz Γ por medio del algoritmo k-medias, para crear un conjunto de clases a las cuales cada patrón de causalidad se puede ver representado por medio de un valor promedio. El valor de k para crear el numero de clases es $k = N_c$. Con esto se busca tener N_c clases dentro de los patrones en la matriz Γ. Por lo tanto la matriz de clases generada tiene un tamaño de $N_c \times (r * r) - r$ elementos.

4. Resultados experimentales

La evaluación de la propuesta se generó utilizando la potencia de un cluster de 19 procesadores de doble núcleo. Cada procesador es un Intel©Xeon©CPU E5-2670 v3 2.30GHz, y 74 GB de RAM.La metodología actual se probó con dos bases de datos de paisajes naturales:

— Vogel y Shiele (V_S) [23]. con 700 Imágenes clasificadas como: 144 costa, 103 bosque, 179 montaña, 131 pradera, 111 río/lago y 32 cielo/nube.

— Oliva y Torralba (O_T) [18]. 1472 Imágenes clasificadas como: 360 costa, 328 bosque, 374 montaña y 410 pradera.

De las dos bases de imágenes anteriormente citadas, se concatenaron, obteniendo una sola base de imágenes amplia. Las escenas naturales son: bosques, cielos, costas, montañas, praderas y ríos.

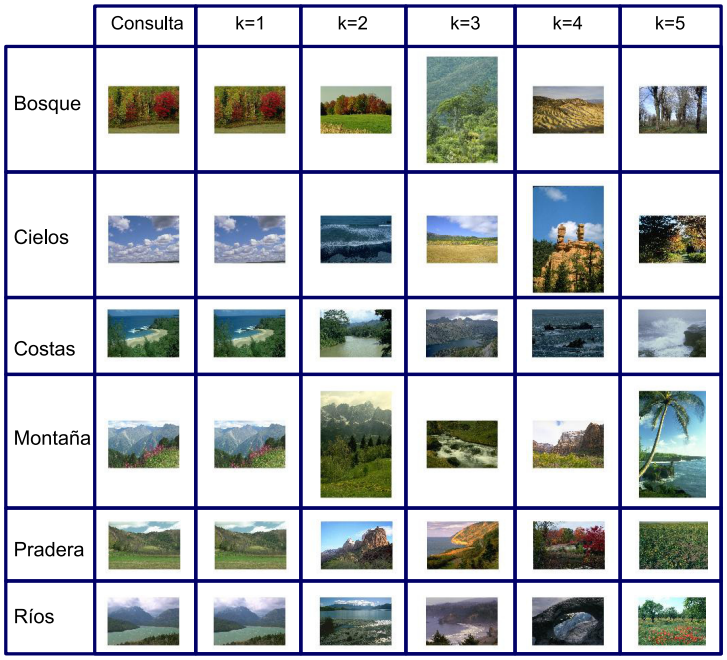


Fig. 6. Prueba con resubstitución sobre toda la base de imágenes.

Tabla 1. Matriz de confusión usando toda la base de imágenes.

Escena$_i$ / Escena$_j$	**Bosques**	**Cielos**	**Costas**	**Montañas**	**Praderas**	**Ríos**
Bosques	100	0	0	0	0	0
Cielos	0	100	0	0	0	0
Costas	0	0	100	0	0	0
Montañas	0	0	0	100	0	0
Praderas	0	0	0	0	100	0
Ríos	0	0	0	0	0	100

Los resultados se tomaron de acuerdo a dos métodos de evaluación de desempeño, por un lado, en resubstitución (ver Fig. 6) y por el otro, en validación cruzada al $70/30\,\%$ (ver Fig. 7). En cuanto al número de centros en el algoritmo k-means, se utilizo un valor de $K = 9$, en virtud que proporcionó los mejores resultados.

Como se puede apreciar en las figuras 6 y 7, se recupera la imagen de búsqueda o consulta y, otras 4 imágenes más parecidas. Si se toma en cuenta el recuperar la imagen más parecida en el método de resubstitución, el presente sistema funciona la $100\,\%$. En cambio, para el método de validación cruzada, las 5 imágenes recuperadas pertenecen al mismo tipo de escena natural.

Finalmente, la Tabla 1 muestra la cuantificación del desempeño del sistema propuesto, vía la matriz de confusión. En la tabla 1 se aprecia una recuperación al $100\,\%$ para cada una de los escenarios naturales.

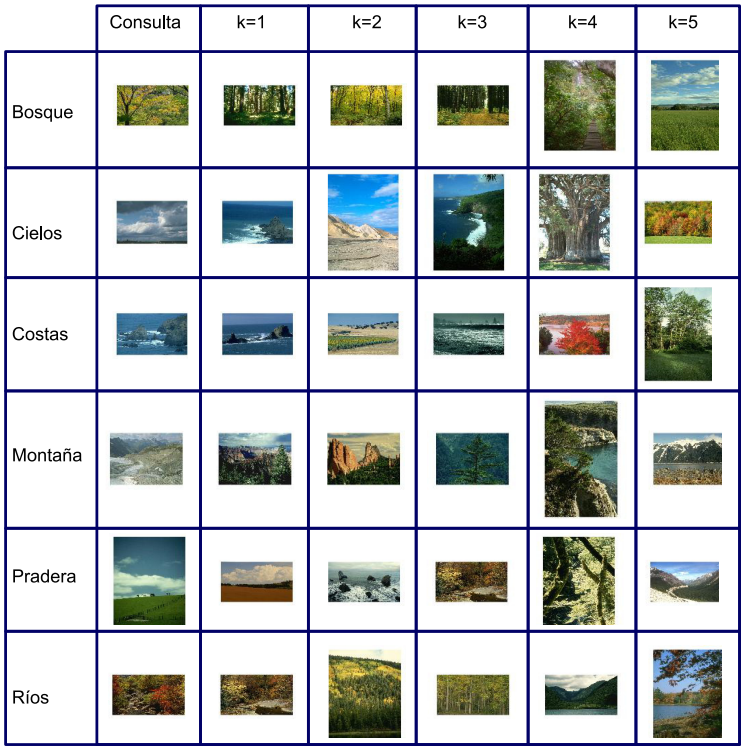


Fig. 7. Prueba con validación cruzada $70/30\,\%$ sobre toda la base de imágenes.

5. Conclusiones y perspectivas

En este trabajo se presentó una novedosa propuesta de uso de la teoría de causalidad de Wiener-Granger, junto con el análisis de auto-organización CBIR, aplicada para la identificación y clasificación de 6 escenarios naturales: costa, bosque, montaña, pradera, río/lago y cielo/nube. Considerando la nueva formulación fue posible encontrar un conjunto de descriptores a partir de las matrices de causalidad para representar una clase de paisaje, a partir de un conjunto automático de texturas de referencia, proponiendo una caracterización de las imágenes basada en la apariencia continua de las texturas dentro de ellas.

Con este enfoque tenemos un $100\,\%$ de clasificación de imágenes para todo el conjunto de datos. Se tiene la ventaja de que no se requiere etiquetado previo ni conocimiento alguno sobre el contenido de las escenas naturales. El trabajo futuro se implementaran las pruebas de rotación, ruido y escala en las imágenes de la base de datos.

Por otro lado, se buscará la implementación de toda la metodología en cómputo paralelo; usando CPUs, la tecnología GPU, las cuales podría rendir eficientemente para la etapa de extracción de características de las imágenes. así como la implementación algoritmos evolutivos, para analizar en conjunto las texturas de la imagen buscando caracterizar el escenario y sus asociaciones con el paradigma de la comprensión visual.

Referencias

1. Alam, A., Jaffery, Z. A.: A vision-based system for traffic light detection. Applications of Artificial Intelligence Techniques in Engineering. Advances in Intelligent Systems and Computing, vol. 698, pp. 333–343 (2016) doi: 10.1007/978-981-13-1819-1_32
2. Ansari, M. A., Singh, D. K.: Human detection techniques for real time surveillance: A comprehensive survey. Multimedia Tools and Applications, vol. 80, no. 6, pp. 8759—8808 (2021) doi: 10.1007/s11042-020-10103-4
3. Barnett, L., Seth, A. K.: The MVGC multivariate granger causality toolbox: A new approach to granger-causal inference. Journal of Neuroscience Methods, vol. 223, pp. 50–68 (2014) doi: 10.1016/j.jneumeth.2013.10.018
4. Begum, R., Halse, S. V.: The smart car parking system using GSM and LabVIEW. Journal of Computer and Mathematical Sciences, vol. 9, no. 2, pp. 135–142 (2018) doi: 10.29055/jcms/740
5. Blaifi, S., Moulahoum, S., Colak, I., Merrouche, W.: Monitoring and enhanced dynamic modeling of battery by genetic algorithm using LabVIEW applied in photovoltaic system. Electrical Engineering, vol. 100, no. 2, pp. 1021–1038 (2017) doi: 10.1007/s00202-017-0567-6
6. Bressler, S. L., Seth, A. K.: Wiener-granger causality: A well established methodology. NeuroImage, vol. 58, no. 2., pp. 323–329 (2011) doi: 10.1016/j.neuroimage.2010.02.059
7. Cortez, J. V., Alvarez, C. B., Alonso, G. R., Cruz, C. A.: Reconocimiento de rostros a partir de la propia imagen usando técnica CBIR. In: Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados, pp. 733–740 (2015)
8. Fablet, R., Bouthemy, P., Perez, P.: Nonparametric motion characterization using causal probabilistic models for video indexing and retrieval. IEEE Transactions on Image Processing, vol. 11, no. 4 (2022) doi: 10.1109/tip.2002.999674
9. Fan, Y., Yang, H., Zheng, S., Su, H., Wu, S.: Video Sensor-Based Complex Scene Analysis with Granger Causality. Sensors, vol. 13, no. 10, pp. 13685–13707 (2013) doi: 10.3390/s131013685
10. Friston, K.: Causal modelling and brain connectivity in functional magnetic resonance imaging. Public Library of Science Biology, vol. 7, no. 2, (2009) doi: 10.1371/journal.pbio.1000033
11. Granger, C. W.: Investigating causal relations by econometric models and cross-spectral methods. Econometrica, vol. 37, no. 3, pp. 424–438 (1969)
12. Jena, B., Nayak, G. K., Saxena, S.: Survey and analysis of content-based image retrieval systems. Control Applications in Modern Power System, Lecture Notes in Electrical Engineering, vol 710, pp. 427–433 (2021) doi: 10.1007/978-981-15-8815-0_37
13. Kim, E., Kim, D. S., Ahmad, F., Park, H.: Pattern-Based Granger Causality Mapping in fMRI. Brain Connectivity, vol. 3, no. 6, pp. 569–577 (2013) doi: 10.1089/brain.2013.0148
14. Kular, D., Ribeiro, E.: Analyzing activities in videos using latent dirichlet allocation and granger causality. Advances in Visual Computing, Springer International Publishing, pp. 647–656 (2015) doi: 10.1007/978-3-319-27857-5_58
15. Li, X., Yang, J., Ma, J.: Recent developments of content-based image retrieval. Neurocomputing, vol. 452, pp. 675–689 (2021) doi: 10.1016/j.neucom.2020.07.139
16. Mannino, M., Bressler, S. L.: Foundational perspectives on causality in large-scale brain networks. Physics of Life Reviews, vol. 15, pp. 107–123 (2015) doi: 10.1016/j.plrev.2015.09.002
17. Matias, F. S., Gollo, L. L., Carelli, P. V., Bressler, S. L., Copelli, M., Mirasso, C. R.: Modeling positive granger causality and negative phase lag between cortical areas. NeuroImage, vol. 99, pp. 411–418 (2014) doi: 10.1016/j.neuroimage.2014.05.063

18. Oliva, A., Torralba, A.: Modeling the shape of the scene: A holistic representation of the spatial envelope. International Journal of Computer Vision, vol. 42, no. 3, pp. 145–175 (2001) doi: 10.1023/A:1011139631724
19. Prabhakar, K., Oh, S., Wang, P., Abowd, G. D., Rehg, J. M.: Temporal causality for the analysis of visual events. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2010) doi: 10.1109/cvpr.2010.5539871
20. Sampath, V., Maurtua, I., Aguilar Martín, J. J., Gutierrez, A.: A survey on generative adversarial networks for imbalance problems in computer vision tasks. Journal of Big Data, vol. 8, no. 1 (2021) doi: 10.1186/s40537-021-00414-0
21. Serrano-Talamantes, J. F., Avilés-Cruz, C., Villegas-Cortez, J., Sossa-Azuela, J. H.: Self organizing natural scene image retrieval. Expert Systems with Aplications: An International Journal, vol. 40, no. 7, pp. 2398–2409 (2012) doi: 10.1016/j.eswa.2012.10.064
22. Tyagi, V.: Content-based image retrieval: Ideas, influences, and current trends. Springer Publishing Company, Incorporated (2017)
23. Vogel, J., Schiele, B.: Performance evaluation and optimization for content-based image retrieval. Pattern Recognition, vol. 39, no. 5, pp. 897–909 (2006) doi: 10.1016/j.patcog.2005.10.024
24. Wiener, N., Masani, P.: The prediction theory of multivariate stochastic processes: I. The regularity condition. Acta Mathematica. International Press of Boston, vol. 98, pp. 111–150 (1957) doi: 10.1007/bf02404472
25. Zhang, C., Yang, X., Lin, W., Zhu, J.: Recognizing human group behaviors with multi-group causalities. In: Proceedings of the 2012 IEEE/WIC/ACM International Joint Conferences on Web Intelligence and Intelligent Agent Technology, vol. 3, pp. 44–48 (2012) doi: 10.1109/WI-IAT.2012.162
26. Zhang, H., Li, X.: Effective connectivity of facial expression network by using Granger causality analysis. Society of Photo-Optical Instrumentation Engineers, vol. 8920 (2013) doi: 10.1117/12.2030912

ISSN 1870-4069

Registro de nubes de puntos por pares con optimización global basado en gráfico de pose para un sistema de reconstrucción 3D

Víctor Beltrán Barrera, Jesús Carlos Pedraza Ortega,
Juan Manuel Ramos Arreguín, Marco Antonio Aceves Fernández,
Saúl Tovar Arriaga, Efrén Gorrostieta Hurtado

Universidad Autónoma de Querétaro,
Facultad de Ingeniería, Santiago de Querétaro,
México

[victor.beltran.barrera, marco.aceves,
efrengorrostieta]@gmail.com,caryoko@yahoo.com,
jsistdig@yahoo.com.mx, saul.tovar@uaq.mx,

Resumen. La reconstrucción 3D se ha convertido en uno de los problemas centrales en visión por computadora y robótica. La principal problemática de estudio en este proceso es el registro de nubes de puntos. El algoritmo iterativo del punto más cercano (ICP) [7] es uno de los más populares para realizar esta tarea. El principal problema de ICP es su susceptibilidad a caer en un mínimo local. En este documento se presenta un algoritmo combinando ICP con métodos globales de optimización. Para enfrentar problemas como ruido y superposición parcial se emplea el algoritmo [10] como inicialización para ICP punto a plano, éste realiza la alineación y construye la representación de nodos y bordes para el gráfico de pose. Finalmente se realiza la optimización global del gráfico. Los resultados muestran que el algoritmo es capaz de superar la precisión de algoritmos del estado del arte como ICP tradicional y algunas de sus variantes reduciendo el valor RMSE y obteniendo tiempos de procesamiento similares.

Palabras clave: Reconstrucción 3D, gráfico de pose, ICP, registro de nubes de puntos.

Pairwise Point Cloud Registration with Global Optimization based on Pose Graph for 3D Reconstruction System

Abstract. 3D reconstruction has become one of the central problems in computer vision and robotics. The main study problem in this process is point cloud registration. Iterative closest point algorithm (ICP) [7] is one of the most popular to performs this task. The main problem on ICP is its susceptibility to local minima. This paper presents a point cloud registration algorithm combining ICP with global optimization methods. To face problems such as noise and partial overlapping, algorithm [10] is used as initialization for ICP point to plane algorithm, which carries out the alignment and forms the nodes and edges representation for pose graph. Finally graph global optimization is performed.

Results shows that the algorithm improve traditional ICP algorithm and some variants accuracy reducing the RMSE and obtaining similar processing times.

Keywords: 3D Reconstruction, pose graph, ICP, point cloud registration.

1. Introducción

Hoy por hoy, la reconstrucción 3D abarca diversas aplicaciones en distintas áreas. En visión por computadora se han desarrollado sistemas de realidad aumentada utilizando este proceso [1]. En la robótica ha permitido desarrollar grandes avances en sistemas de localización y mapeo simultaneo (SLAM) [2], así como en vehículos de exploración y rescate [3]. En el sector automotriz su mayor auge se está alcanzando con los vehículos autónomos utilizando sensores *LIDAR* para obtener los datos de profundidad y realizar un mapeo de los alrededores [4].

El registro de las nubes de puntos toma un papel fundamental dentro del proceso de reconstrucción ya que una buena alineación permitirá obtener un buen efecto [5]. El objetivo del registro de las nubes de puntos es encontrar correspondencias entre dos o más conjuntos de puntos y obtener una matriz de rotación junto con un vector de traslación de tal manera que los puntos se vayan alineando unos con otros.

En aplicaciones reales los métodos para el registro de las nubes de puntos sufren diversos retos debido a la calidad de los datos de entrada. Para realizar la reconstrucción de una escena se requiere tomar capturas de distintas poses, esto puede causar la deformación de las nubes de puntos. Otro factor importante es el ruido ocasionado por los sensores utilizados para la adquisición de los datos ya que pueden ocasionar la aparición de puntos fuera de línea los cuales no tienen correspondencias con otras nubes de puntos [6].

El registro de nubes de puntos se divide en dos categorías: registro por pares y registro por grupos. En la primera categoría se toman dos nubes de puntos únicamente durante el proceso mientras que en la segunda se pueden tomar dos o más nubes de puntos. El algoritmo iterativo del punto más cercano (ICP) [7] se encuentra dentro de la categoría de registro por pares.

El proceso de registro está constituido generalmente por dos fases: pre-alineación y la transformación afín. La pre-alineación tiene dos enfoques: a nivel local utiliza descriptores como los histogramas de características de puntos (PFH) [8] para codificar la variación de la forma en el vecindario de puntos y a nivel global toman todos los puntos en una sola cuenta [9].

ICP es un método local para el registro de conjuntos de puntos que se basa en la distancia euclidiana de pares de puntos entre ambos conjuntos.

En su forma básica ICP empieza con una alineación inicial sin conocimiento alguno de correspondencias entre ambos conjuntos de puntos y después opera de manera iterativa bajo dos pasos: 1) buscar y establecer correspondencias entre los puntos cercanos de ambos conjuntos, y 2) recalcular la estimación de la transformada por medio de mínimos cuadrados para encontrar la matriz de rotación, así como el vector de traslación que permita alinear de mejor manera ambas nubes de puntos.

Una buena inicialización del algoritmo ICP es indispensable para obtener óptimos resultados y de ello dependerá su convergencia y susceptibilidad de caer en un mínimo local [6]. Es por esto que este algoritmo debería ser aplicado sólo en la transformación

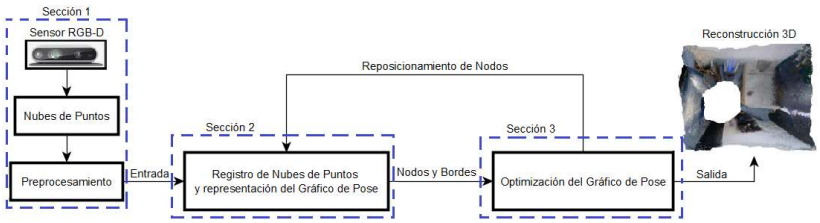


Fig. 1. Modelo del sistema de reconstrucción 3D.

afín. Algunas de las soluciones más comunes ante esta situación son el uso de optimizadores globales como refinamiento y la utilización de métodos de registro globales, los cuales se basan en extraer las características geométricas de las nubes de puntos para realizar el registro. Debido a que los métodos de registro global no requieren una aproximación inicial usualmente son utilizados como inicialización para los métodos locales con el fin de obtener mejores resultados [10].

Es común el uso de gráficos de pose en sistemas de reconstrucción 3D y sistemas SLAM. Este método hace la representación de una estructura de gráficos constituida por nodos y bordes que facilita el procesamiento y es flexible en términos de observación en sistemas que utilizan sensores RGB-D. Por estas razones son una herramienta muy práctica y fácil de implementar cuando se requiere optimizar el mapeo de una escena [11].

Por lo anteriormente descrito, en este trabajo, se propone un sistema de reconstrucción 3D interior utilizando un sensor RGB-D para adquirir los datos de entrada. En el registro de las nubes de puntos se emplea una variante del algoritmo RANSAC como pre-alineación para enfrentar problemas como ruido y superposición parcial. En la transformación afín se utiliza el algoritmo ICP punto a plano el cual se encarga de realizar la alineación de las nubes de puntos y construir la representación de nodos y bordes para el gráfico de pose.

Finalmente se realiza la optimización global del gráfico utilizando métodos de mínimos cuadrados no lineales. Los resultados obtenidos demuestran que el algoritmo es capaz de superar la precisión de algoritmos del estado del arte como ICP tradicional y algunas de sus variantes reduciendo el valor RMSE y obteniendo tiempos de procesamiento similares.

El resto del artículo se encuentra organizado de la siguiente manera: en la sección 2 se analizan brevemente algunos algoritmos para el registro de nubes de puntos, en la sección 3 se presenta el modelo propuesto, los materiales y métodos para el desarrollo de la metodología y la comparación de los resultados obtenidos en la fase experimental. Finalmente, en la sección 4 se presenta la discusión de las conclusiones.

2. Trabajos relacionados

Desde su creación el algoritmo ICP ha sido ampliamente utilizado para realizar la tarea de registro de conjuntos de puntos. En [7] cada uno de los puntos dentro de un conjunto son emparejados con el punto más cercano del otro conjunto. La métrica del error se basa en la minimización de la distancia euclidiana entre cada uno de estos pares

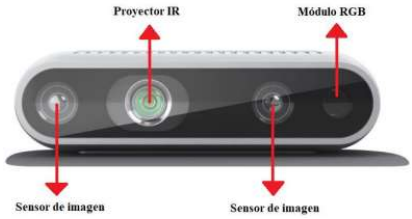


Fig. 2. Sensor de profundidad Intel RealSense D435 [23].

de puntos. Este proceso se repite hasta alcanzar el criterio de término que puede ser por número de iteraciones o cierta magnitud de error alcanzada lo cual hace que el tiempo de procesamiento sea elevado. Por otro lado, en [12] se utiliza una métrica de error basada en minimizar la distancia euclidiana entre un punto y un plano tangente correspondiente a este punto por medio de métodos de mínimos cuadrados no lineales como en [13]. Aunque el tiempo de convergencia del algoritmo ICP punto a plano se incrementa un poco con respecto a ICP punto a punto en cada iteración se logran obtener mejores resultados y una convergencia más rápida.

Para enfrentar los principales problemas que sufre este algoritmo ante factores como puntos sin correspondencias, poca superposición entre ambos conjuntos de puntos, el pre-alineamiento y la susceptibilidad a converger hacia un mínimo local han sido propuestos diferentes métodos y variantes de ICP.

En [14] se hace uso de técnicas probabilísticas en el emparejamiento de puntos para reducir los puntos sin correspondencias, [15] propone un método basado en asignar pesos a los puntos con menor distancia euclidiana para reducir el error de medida en puntos sin correspondencias. En [16] se presenta una variante de ICP junto con el algoritmo RANSAC para remover puntos sin correspondencias y el uso de técnicas de aceleración de búsqueda como *kd-Tree* [24]. En [13] se propone el uso de mínimos cuadrados no lineales junto con ICP para conjuntos de puntos con bajo nivel de superposición. Debido a la necesidad de una buena inicialización, se han implementado técnicas como el uso de descriptores [8, 17] para codificar la variación de la forma en los conjuntos de puntos, y encontrar un movimiento inicial adecuado por medio de RANSAC [18], o su variante [10], cuya principal ventaja es que no necesitan un alineamiento inicial, sin embargo, el tiempo de procesamiento se incrementa.

Para evitar caer en mínimos locales [9] propone un modelo global con baja complejidad computacional basado en ICP utilizando árboles dimensionales (*kd-Tree*) para búsqueda de vecinos cercanos y los vectores normales para realizar la rotación. Otras variantes a nivel global como [19] emplean métodos heurísticos. Go-ICP [20] obtiene la posición global óptima basada en una estrategia de *Branch and bound* para encontrar la transformada rígida, sin embargo, tiene un tiempo de convergencia muy alto debido a la técnica utilizada.

El uso de optimizadores para resolver problemas de mínimos cuadrados no lineales es aplicado como refinamiento en el registro de conjuntos de puntos a nivel global como Levenberg Marquardt (LM) y Gauss Newton (GN). Este tipo de métodos en conjunto con métodos de registro a nivel local hacen una buena combinación para obtener mejores resultados. Una de las técnicas que facilita el proceso de reconstrucción son los gráficos de pose ya que conducen a una estructura fácil de procesar basada en nodos y bordes además de ofrecer una mayor flexibilidad en términos de observación ante la

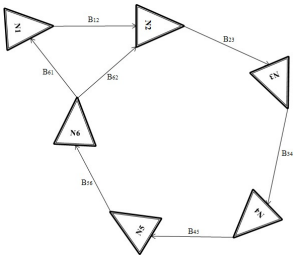


Fig. 3. Representación de un gráfico de pose.

deformación de las nubes de puntos al utilizar sensores RGB-D o LIDAR para la adquisición de datos. En [11] se presenta un sistema de SLAM con un sensor LIDAR basado en gráficos de pose, en este trabajo se utiliza el algoritmo ICP punto a plano para el registro de puntos y se optimiza a nivel global por medio del algoritmo Levenberg Marquardt. En este trabajo se presenta una variación del trabajo de [11] enfocado a la reconstrucción 3D con un sensor RGB-D.

Actualmente los sensores RGB-D son más accesibles que los sensores LIDAR en términos económicos y dependiendo de la aplicación final llegan a ser mucho más prácticos. Además de esto se introduce una pre-alineación con el algoritmo presentado en [10] para mejorar el resultado del algoritmo ICP punto a plano y tener una mejor representación del gráfico de pose.

3. Artefactos propuestos

El modelo del sistema de reconstrucción 3D se muestra en la Fig. 1. El sistema está dividido en 3 secciones: En la primera parte se realiza la adquisición de las nubes de puntos por medio de un sensor RGB-D y se realiza su preprocesamiento.

En la segunda sección se detalla el proceso del registro de las nubes de puntos constituido por dos partes; la pre-alineación por medio del algoritmo propuesto en [10] junto con la transformación afín de las nubes de puntos por medio de ICP punto a plano, y la representación de los nodos y bordes para la construcción del gráfico de pose.

Finalmente, en la tercera sección se muestra la optimización a nivel global del gráfico sobre los nodos y bordes generados. Después de esto se retorna la nueva posición de los nodos al algoritmo de alineación para realizar el registro con una nueva nube de puntos, este proceso se va repitiendo con cada una de las nubes de puntos que se adquieran para el mapeo de la escena interior tomando como nuevo objetivo el resultado del proceso hecho previamente y como fuente una nueva nube de puntos previamente capturada por el sensor de profundidad.

3.1. Adquisición y preprocesamiento de las nubes de puntos

Una parte muy importante dentro del sistema de reconstrucción 3D es la adquisición de los datos. Generalmente los modelos del estado del arte trabajan con información obtenida directamente de una base de datos, esto causa que al llevar dichos algoritmos a aplicaciones con datos del mundo real tengan problemas de funcionamiento y

Tabla 1. Resultados obtenidos en el registro de nubes de puntos.

Algoritmo	RMSE	*fitness score*	Procesamiento
RANSAC	0.05366414	0.8955023	0.471seg
ICP punto a punto	0.04651124	0.5731295	1.159seg
FGR	0.01513006	0.1847563	0.692seg
ICP punto a plano	0.01496852	0.3283060	0.868seg

Tabla 2. Resultados obtenidos registro de puntos con optimización global.

Algoritmo	RMSE	*fitness score*	Procesamiento	Residual
ICP punto a plano	0.0323	0.8287	0.595999seg	$1.2272\ e^{-30}$
FGR-ICP punto a plano	0.0322	0.8287	0.539982seg	$3.9004e^{-31}$

resultados poco eficientes por lo que el preprocesamiento de los datos de entrada es parte fundamental del proceso.

Para este trabajo se realiza la adquisición de las nubes de puntos por medio del sensor de profundidad Intel RealSense D435 [23], mostrado en la Fig. 2. El sensor está compuesto por un sensor RGB, un proyector infra rojo y dos sensores de imagen.

Una vez capturadas las nubes de puntos se realiza el preprocesamiento para su entrada al algoritmo de alineación. En esta parte se reduce el número de puntos contenido dentro de cada conjunto para optimizar el proceso, después se normalizan los datos y por medio de [17] se genera el conjunto de correspondencias iniciales para el algoritmo de pre-alineación.

3.2. Registro de las nubes de puntos y representación del gráfico de pose

El proceso de registro está constituido por tres partes: en las primeras dos se trata el problema del registro de las nubes de puntos dividido en la pre-alineación y la transformación afín y en la tercera parte se detalla la representación del gráfico de pose.

3.2.1. Pre-alineación (Fast Global Registration)

Una vez procesados los datos de las nubes de puntos $P = \{p_1, p_2, \dots, p_N\}$ $\mathbb{R}^3$, $Q = \{q_1, q_2, \dots, q_N\}$ $\mathbb{R}^3$ y obtenido el conjunto de correspondencias iniciales $K = \{(p_i, q_i), \dots, (p_m, q_m)\}$, se realiza la pre-alineación de las nubes de puntos por medio del algoritmo *Fast Global Registration* [10]. El fin de este algoritmo es optimizar la transformación rígida M con la forma mostrada en la ecuación 1, sin embargo, en esta parte el objetivo es ofrecer una buena estimación de M para la transformación afín:

$$E(M) = \sum_{(p_i,q_i)\in K} \vartheta(\|p_i - Mq_i\|), \tag{1}$$

donde ϑ es el estimador escalado de German-McClure para diferentes valores del residuo μ con la forma mostrada en la ecuación 2:

$$\vartheta(\|p_i - Mq_i\|) = \frac{\mu(\|p_i - Mq_i\|)^2}{\mu + (\|p_i - Mq_i\|)^2}. \quad (2)$$

3.2.2. Transformación Afín (ICP punto a plano)

Una vez que se tiene la pre-alineación de [10] se pasa el resultado obtenido de la transformación inicial M y su información al algoritmo ICP punto a plano para realizar la transformación afín.

Cuando se utiliza ICP punto a plano se busca minimizar la distancia de la suma de los cuadrados entre cada punto perteneciente a la fuente y el plano de la nube de puntos objetivo en cada uno de sus puntos de correspondencia en K [21]. De manera formal, si $q_i = (q_{ix}, q_{iy}, q_{iz}, 1)^T$ es un punto perteneciente a la fuente Q, $p_i = (p_{ix}, p_{iy}, p_{iz}, 1)^T$ es el punto de correspondencia perteneciente al objetivo P y $n_i = (n_{ix}, n_{iy}, n_{iz}, 0)^T$ es el vector normal en p_i, como se muestra en la ecuación 3, se desea obtener la transformación M_F:

$$M_F = argmin_M \sum_{(p_i, q_i) \in K} ((Mq_i - p_i) \cdot n_i)^2, \quad (3)$$

donde M y M_F representan las matrices 4x4 de la transformación rígida.

La matriz de transformación de cuerpo rígido M_F está compuesta por una matriz de rotación $R_{PQ}(\alpha, \beta, \gamma)$ y una matriz de traslación $t_{PQ}(t_x, t_y, t_z)$ como se muestra en las ecuaciones 4 y 5 respectivamente:

$$\mathrm{R_{PQ}}(\alpha, \beta, \gamma) = \mathrm{R_{PQ_z}}(\gamma) \cdot \mathrm{R_{PQ_y}}(\beta) \cdot \mathrm{R_{PQ_x}}(\alpha) = \begin{pmatrix} \mathrm{R_{PQ_{11}}} & \mathrm{R_{PQ_{12}}} & \mathrm{R_{PQ_{13}}} & 0 \\ \mathrm{R_{PQ_{21}}} & \mathrm{R_{PQ_{22}}} & \mathrm{R_{PQ_{23}}} & 0 \\ \mathrm{R_{PQ_{31}}} & \mathrm{R_{PQ_{32}}} & \mathrm{R_{PQ_{33}}} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad (4)$$

$$t_{PQ}(t_x, t_y, t_z) = \begin{pmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}, \quad (5)$$

donde

$$\begin{aligned} R_{PQ_{11}} &= cos(\gamma)cos(\beta), \\ R_{PQ_{12}} &= -sin(\gamma)cos(\alpha) + cos(\gamma)sin(\beta)sin(\alpha), \\ R_{PQ_{13}} &= sin(\gamma)sin(\alpha) + cos(\gamma)sin(\beta)cos(\alpha), \\ R_{PQ_{21}} &= sin(\gamma)cos(\beta), \\ R_{PQ_{22}} &= cos(\gamma)cos(\alpha) + sin(\gamma)sin(\beta)sin(\alpha), \\ R_{PQ_{23}} &= -cos(\gamma)sin(\alpha) + sin(\gamma)sin(\beta)cos(\alpha), \\ R_{PQ_{31}} &= -sin(\beta), \\ R_{PQ_{32}} &= cos(\beta)sin(\alpha), \\ R_{PQ_{33}} &= cos(\beta)cos(\alpha). \end{aligned}$$

La salida del algoritmo de alineación es la matriz de transformación M_F que al ser aplicada a la fuente alinea cada uno de sus puntos con los puntos de correspondencia en la nube de puntos objetivo. Debido a que α, β y γ son argumentos no lineales en la

Fig. 4. Ejemplo de Nubes de puntos capturadas con [23].

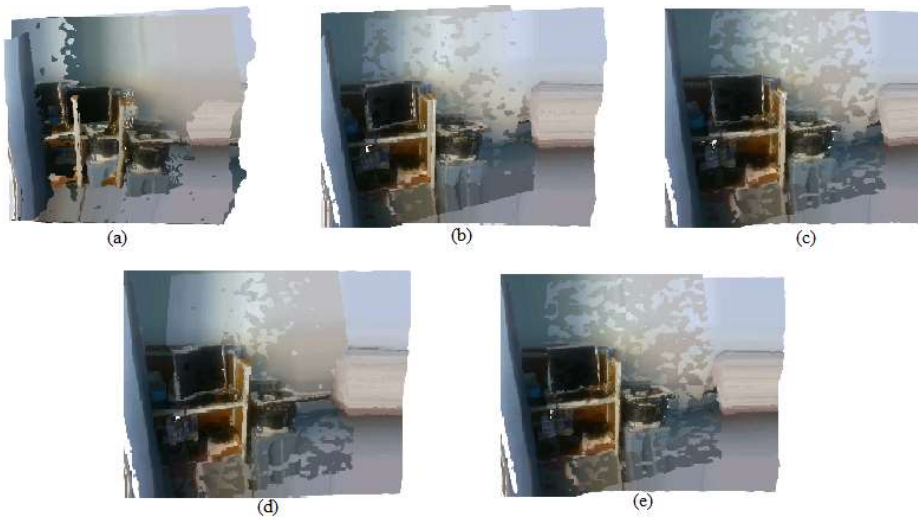


Fig. 5. Alineación obtenida para dos nubes de puntos. (a) Entrada, (b)RANSAC, (c)ICP punto a punto, (d)FGR, (e)ICP punto a plano.

matriz de rotación, se hace uso de técnicas de solución para mínimos cuadrados no lineales en la parte de optimización del gráfico de pose.

3.2.3. Gráfico de pose

Se puede denotar la estructura de un gráfico de pose como $G = \{N, B\}$. Aquí $N = \{N_1, \dots, N_m,\}$ donde cada uno de los N_i nodos representan una pose del sistema en el instante t_i y $B = \{B_{N_i N_j}, \dots, B_{N_{m-1} N_m}\}$ en donde cada término representa la medición de odometría entre cada nodo. En la Fig. 3 se muestra un ejemplo de la estructura de un gráfico de pose.

Cada uno de los nodos es representado como $N_i = [{\tau_i}^T, {S_i}^T]^T$ y corresponde a un punto en la trayectoria de escaneo, contiene información como su traslación en un sistema global de coordenadas el cual es representado por medio de $\tau_i = [x_i, y_i, z_i]^T$. La información como su posición y rotación también es representada en el sistema global de coordenadas por medio del cuaternión $S_i = [sw_i, sx_i, sy_i sz_i]^T$.

Fig. 6. Proceso de registro de las nubes de puntos. ICP punto a plano (Izq.), FGR-ICP punto a plano (Der.).

3.3. Optimización global del gráfico de pose

La optimización del gráfico de pose es un método global. Esto quiere decir que cada estimación realizada para cada uno de los nodos se verá afectada por cada borde dentro del gráfico [22]. De acuerdo a [11] la función objetivo que obtiene la pose óptima para cada una de las variables de pose $\tilde{\mathrm{N}} = \{\tilde{\mathrm{N}}_1, \ldots, \tilde{\mathrm{N}}_m\}$ esta dada por la ecuación 6:

$$\tilde{\mathrm{N}} = argmin_N \sum_{i=1}^{m} \sum_{j=1}^{m} r(N_i, N_j, B_{ij}) \quad i < j, \tag{6}$$

donde m es el número total de nodos en la trayectoria y r el residuo entre cada borde B_{ij} entre los nodos N_i y N_j.

De esta manera, de acuerdo a la literatura en [21], la función objetivo propuesta en la ecuación 3 por el algoritmo ICP punto a plano se modifica tomando la forma de la ecuación 7:

$$M_F(R,t) = min\left\{\sum_{i=1}^{m} \left\|\left((R \cdot q_i + t - p_i) \cdot n_i\right)\right\|_2^2\right\}, \tag{7}$$

donde $q_i = (q_{ix}, q_{iy}, q_{iz}, 1)^T$ es un punto perteneciente a la fuente Q, $p_i = (p_{ix}, p_{iy}, p_{iz}, 1)^T$ es el punto de correspondencia perteneciente al objetivo P, $n_i = (n_{ix}, n_{iy}, n_{iz}, 0)^T$ es el vector normal en p_i, R es el parámetro de rotación y t el parámetro de traslación en la transformada M_F.

Cuando se realiza el registro en las nubes de puntos se obtienen la transformada M_{Fij} con los parámetros de rotación R_{ij} y traslación t_{ij} para cada uno de los nodos en el gráfico de pose, al transformar la parte de rotación en un cuaternión, esto se puede expresar como $[{\tau_{ij}}^T, {S_{ij}}^T]^T$.

Para cada borde $B_{ij} = [{\tau_{Bij}}^T, {S_{Bij}}^T]^T$ entre los nodos N_i y N_j el residual se divide en dos, el residual para la parte de rotación (ecuación 8), y el residual para la parte de traslación (ecuación 9). En cada borde B_{ij} su parte de rotación y traslación corresponden a S_{Bij} y τ_{Bij} respectivamente y la función $V(S)$ retorna la parte del vector del cuaternión S, $[sx, sy, sz]$:

$$r_R(N_i, N_j, B_{ij}) = R(S_i)^T(\tau_i - \tau_j) - \tau_{Bij}, \tag{8}$$

$$r_t(N_i, N_j, B_{ij}) = 2 \cdot V({S_i}^{-1} S_j {S_{Bij}}^{-1}), \tag{9}$$

Fig. 7. Reconstrucción 3D final. (Izq.) Escena 1, (Der.) Escena 2.

Fig. 8. Nivel de detalles en el sistema de reconstrucción 3D.

donde $r_R(N_i, N_j, B_{ij})$ corresponde a la parte residual de la rotación y $r_t(N_i, N_j, B_{ij})$ a la parte residual de la traslación.

De esta manera, el residuo total entre la parte de rotación y traslación está dado por la ecuación 10:

$$r(N_i, N_j, B_{ij}) = w_R \| r_R(N_i, N_j, B_{ij}) \|_2 + w_t \| r_t(N_i, N_j, B_{ij}) \|_2, \quad (10)$$

donde w_R y w_t son los pesos de las partes de rotación y traslación respectivamente.

3.4. Experimentación y resultados

El sistema de reconstrucción 3D fue implementado en una computadora hp notebook con un procesador AMD A10-5745M a 2.1GHz y 6GB de RAM. Con el sensor de profundidad [23] se capturaron distintas instancias de una escena interior, en la Fig. 4 se muestran dos ejemplos de las nubes de puntos obtenidas. Como parte de las pruebas dentro del proceso de registro se compararon los algoritmos RANSAC, ICP punto a plano, ICP punto a punto y el algoritmo FGR [10] en la alineación de un par de nubes de puntos para determinar su desempeño.

Para evaluar el desempeño de los algoritmos se tomaron en cuenta tres métricas: *fitness score* que representa la superficie de superposición entre nubes de puntos, el error cuadrático medio (RMSE) y el tiempo de procesamiento. En la Fig. 5 se puede observar la alineación obtenida por cada algoritmo.

El algoritmo de registro global basado en RANSAC obtiene un valor RMSE de 0.04466 con un *fitness score* de 0.891. Al aplicar el algoritmo ICP punto a punto se

Tabla 3. Resultados finales obtenidos por el sistema de reconstrucción 3D.

Reconstrucción	Modelo	RMSE	*fitness score*	Procesamiento	Residual
Escena 1	Propuesto	0.0310	0.9422	118.55655seg	$3.43\ e^{-04}$
Escena 1	Tradicional	0.0324	0.9422	117.91059seg	$3.74\ e^{-04}$
Escena 2	Propuesto	0.0275	0.9215	286.54123seg	$9.85e^{-05}$
Escena 2	Tradicional	0.0277	0.9045	287.21839seg	$1.13e^{-04}$

obtiene un valor RMSE de 0.011 con *fitness score* de 0.573. FGR obtiene un valor RMSE de 0.01513 con un *fitness score* de 0.184. Por último, el algoritmo ICP punto a plano obtiene un valor RMSE de 0.01496 con un *fitness score* de 0.328. En la Tabla 1 se muestran los resultados de las métricas aplicadas para cada algoritmo.

Como se muestra en la Tabla 1, el mejor rendimiento lo presentan los algoritmos FGR e ICP punto a plano. Aunque el menor tiempo de procesamiento lo tiene el algoritmo basado en RANSAC su desempeño en cuanto al precisión es bajo pues su valor RMSE es elevado. En términos de precisión y superficie superpuesta se puede observar que ICP punto a plano tiene un mejor desempeño que FGR siendo el tiempo de procesamiento el único punto donde esté último lo supera.

Con base en estos resultados se procedió a realizar el proceso del registro de las nubes de puntos y la representación de los nodos y bordes que conforman el gráfico de pose. Primero, como parte del preprocesamiento se hizo un muestreo para reducir el número de puntos y se realizó la normalización para cada dato de entrada, después se realizó la implementación del algoritmo ICP punto a plano en ambas partes del proceso de registro, esto es, tanto en la pre-alineación como en la transformación afín.

Por otra parte, se realizó la implementación del algoritmo FGR como inicialización del algoritmo ICP punto a plano. En la pre-alineación sólo se requiere una buena estimación por lo que el número de iteraciones del algoritmo FGR es reducido a 10 para acelerar el proceso. Este proceso fue aplicado sobre el mismo par de nubes de puntos analizadas previamente. En la Fig. 6 se muestra el registro de nubes de puntos por ambos algoritmos.

Una de las mejoras a simple vista fue la precisión en la alineación de los contornos ya que en el resultado obtenido por el algoritmo conformado solamente por ICP punto a plano se puede observar un pequeño desalineamiento mientras que en el algoritmo inicializado con FGR se disminuye notoriamente, esto se ve reflejado en los resultados de las métricas mostrados en la Tabla 2 incluyendo el valor residual obtenido al realizar la optimización global del gráfico de pose por medio del algoritmo Levenberg Marquardt.

Para validar el desempeño del sistema de reconstrucción 3D se realizó el escaneo de dos escenas interiores. La primera escena consta de un total de 19 instancias y la segunda de 34 instancias adquiridas por el sensor de profundidad utilizado.

En la Fig. 7 (Izq.) se muestran los resultados finales obtenidos para la primera escena teniendo como resultado un valor RMSE de 0.0310 y un *fitness score* de 0.9422 con un residual final de 0.000343. y en la Fig. 7 (Der.) se muestran los resultados para la segunda escena obteniendo un RMSE de 0.0275 y un *fitness score* de 0.9215 con un residual final de 0.0000985.

En la Tabla 3 se muestran los resultados finales obtenidos por el sistema de reconstrucción 3D utilizando el modelo propuesto y el método tradicional ICP punto a plano utilizado en [11].

Algo importante de mencionar sobre el modelo de reconstrucción realizado es que su aplicación podría ser en sistemas que requieran poca resolución de mapeo como sistemas de reconocimiento y exploración o en aplicaciones donde los detalles no sean tan importantes a gran escala. Como se puede observar en la Fig. 8 (Izq.) los detalles más grandes se mantienen un poco asemejados a la realidad en objetos de gran tamaño. Sin embargo, al realizar un acercamiento como se observa en la Fig. 8 (Der.) se ve claramente que estos detalles se van perdiendo a medida que se hace un acercamiento a los objetos más pequeños.

4. Conclusiones y trabajo a futuro

En este trabajo se realizó la reconstrucción 3D de dos escenas interiores por medio de la inicialización a nivel global del algoritmo presentado en [10] junto con el método de registro a nivel local ICP punto a plano y el método de gráfico de pose, demostrando en los resultados obtenidos que la utilización de métodos locales en conjunto con métodos globales de optimización permite obtener mejores resultados que utilizándolos por separado.

Establecer una buena pre-alineación para los algoritmos locales es crucial en su desempeño, por esta razón se utilizó un método global sencillo y capaz de obtener buenos resultados como inicialización con pocas iteraciones. Si bien el uso del método de gráfico de pose incrementa el tiempo de procesamiento, también hace posible obtener una mejor estimación de la transformación rígida y con esto lograr un mejor mapeo tridimensional de la escena demostrando la reducción del error cuadrático medio de manera considerable entre cada una de las nubes de puntos alineadas.

Dentro del mapeo obtenido por el sistema de reconstrucción 3D se puede observar que el uso de una cámara de profundidad presenta varias ventajas, algunas de las más llamativas son su bajo costo en comparación con otro tipo de sensores como LIDAR y su pequeño tamaño lo que hace más fácil su portabilidad e implementación. El mapa generado demuestra mantener las características necesarias para la reconstrucción presentando una buena consistencia en la nube de puntos final generada.

Uno de los principales puntos de mejora en el presente trabajo es la baja resolución de detalles en la reconstrucción, esto hace que el modelo sea aplicable en sistemas en los que los detalles no sean muy importantes sino más bien las formas y los contornos de las estructuras dentro del campo de reconstrucción, además de eso el rango activo de captura está configurado para trabajar de 0.15 a 2.3 metros de distancia, por lo que cuando se escanean superficies fuera de este rango se generan puntos fuera de línea dificultando el proceso de reconstrucción y haciéndolo menos preciso.

Referencias

1. Wu, Y., Chan, L., Lin, W.: Tangible and visible 3D object reconstruction in augmented reality. IEEE International Symposium on Mixed and Augmented Reality (ISMAR), Beijing, China, pp. 26–36 (2019) doi: 10.1109/ISMAR.2019.00-30

2. Durrant-Whyte, H., Bailey, T.: Simultaneous localization and mapping (SLAM) Part 1 The Essential Algorithms. University of Sydney (2006)
3. Chen, W., Ihara, S., Hasegawa, M.: Proposal of a rescue operation support system based on 3D reconstruction, GPS, and digital pen. In: Proceedings International Workshop on Advanced Imaging Technology (IWAIT), vol. 11515 (2020) doi: 10.1117/12.2566964
4. Lee, H., Song, S., Jo, S.: 3D Reconstruction using a sparse laser scanner and a single camera for outdoor autonomous vehicle. In: Proceedings of 19th International Conference on Intelligent Transportation Systems (ITSC), pp. 629–634 (2016) doi: 10.1109/ITSC.2016.7795619
5. Furukawa, Y., Hernández, C.: Multi-view stereo: A tutorial. Foundations and Trends in Computer Graphics and Vision, vol. 9, no. 1-2, pp. 1–148 (2013) doi: 10.1561/0600000052
6. Zhu, H., Guo, B., Zou, K., Li, Y., Yuen, K., Mihaylova, L., Leung, H.: A review of point set registration: from pairwise registration to groupwise registration. Sensors Journal, vol. 19, no. 5, pp. 1191 (2019) doi: 10.3390/s19051191
7. Besl, P. J., McKay, N. D.: A method for registration of 3-D shapes. In Robotics-DL tentative, pp. 586–606 (1992) doi: 10.1117/12.57955
8. Rusu, R. B., Blodow, N., Marton, Z. C., Beetz, M.: Aligning point cloud views using persistent feature histograms. In: Proceedings of 21st IEEE International Conference on Intelligent Robots and Systems (IROS), pp. 22–26 (2008) doi: 10.1109/IROS.2008.4650967
9. Linh, T. N., Hiroshi, H.: Global iterative closest point using nested annealing for initialization. In: Proceedings of 19th International Conference on Knowledge Based and Intelligent Information and Engineering Systems, Procedia Computer Science, pp. 381–390 (2015) doi: 10.1016/j.procs.2015.08.147
10. Zhou, Q., Park, J., Koltun, V.: Fast global registration. European Conference on Computer Vision, pp. 766–782 (2016) doi: 10.1007/978-3-319-46475-6_47
11. Yan, L., Dai, J., Tan, J., Hua, L., Chen, C.: Global fine registration of point cloud in LiDar SLAM based on pose graph. Journal of geodesy and Geoinformation science, vol. 48, no. 3, pp. 313–321 (2019) doi: 10.11947/j.AGCS.2019.20170716
12. Chen, Y., Medioni, G.: Object modeling by registration of multiple range images. In: Proceeding of the 1991 IEEE International Conference on Robotics and Automation, vol. 10, no. 3, pp. 145–155 (1991) doi: /10.1016/0262-8856(92)90066-C
13. Wu, Y., Wang, W., Lu. K., Wei, Y., Chen, Z.: A new method for registration of 3D point sets with low overlapping ratios. In: 13th CIRP conference on Computer Aided Tolerancing. vol. 27, pp. 202–206 (2015) doi: 10.1016/j.procir.2015.04.067
14. Agamennoni, R., Fontana, S., Sorrenti, D.: Point clouds registration with probabilistic data association. In: Proceeding of the International Conference on Intelligent Robots and Systems (IROS), pp. 4092–4098 (2016) doi: 10.1109/IROS.2016.7759602
15. Liu, S., Gao, D., Wang, P., Guo, X., Xu, J., Liu, D.: A depth-based weighted point cloud registration for indoor scene. Sensors Journal, vol. 18, no. 11, pp. 3608 (2018) doi: 10.3390/s18113608
16. Seal, A., Bhowmick, A.: Performance analysis of iterative closest point (ICP) algorithm using modified hausdorff distance. International Research Journal of Engineering and Technology (IRJET), vol. 4, no. 07 (2017)
17. Rusu, R., Blodow, N., Beetz, M.: Fast point feature histograms (FPFH) for 3D registration. In: International Conference on Robotics and Automation, pp. 3212–3217 (2009) doi: 10.1109/ROBOT.2009.5152473
18. Chen, C., Hung, Y., Cheng, J.: Ransac-based darces: A new approach to fast automatic registration of partially overlapping range images. IEEE Trans. Pattern Anal. Mach. Intell, vol. 21, no. 11, pp. 1229–1234 (1999) doi: 10.1109/34.809117
19. Koguciuk, D.: Parallel RANSAC for point cloud registration. foundations of computing and decision sciences. vol. 42, pp. 203–217 (2017) doi: 10.1515/fcds-2017-0010

20. Yang, J., Li, H., Dylan, C., Jia, Y.: Go-ICP: A globally optimal solution to 3D ICP point-set registration. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 11, pp. 2241-2254 (2016) doi: 10.1109/TPAMI.2015.2513405
21. Low, K.: Linear least-squares optimization for point-to-plane ICP surface registration. Department of Computer Science, University of North Carolina at Chapel Hill. Technical report (2004)
22. Stachniss, C.: Graph-based SLAM using pose graphs. https://www.youtube.com/nwatch?v=uHbRKvD8TWg (2020)
23. IntelREALSENSE. https://www.intelrealsense.com/depth-camera-d435
24. Miller, F., Vandome, A., Mcbrewster, J.: KD-Tree. Alpha Press (2009)

ISSN 1870-4069

Sistema para clasificación de texturas en imágenes mediante aprendizaje profundo y características wavelet

Juan Manuel Fortuna-Cervantes[1], Marco Tulio Ramírez-Torres[2], Marcela Mejía-Carlos[1], José Salomé Murguia-Ibarra[1], Juan Martínez-Carranza[3]

[1] Universidad Autónoma de San Luis Potosí, Facultad de Ciencias-IICO, México

[2] Universidad Autónoma de San Luis Potosí, Coordinación Académica Región Altiplano Oeste, México

[3] Instituto Nacional de Astrofísica Óptica y Electrónica, Computer Science Department, México

a158852@alumnos.uaslp.mx, {tulio.torres, marcela.mejia, ondeleto}@uaslp.mx, carranza@inaoep.mx

Resumen. La caracterización de texturas en imágenes digitales, se ha convertido en una herramienta de análisis en el área de visión computacional. La textura dentro de la percepción visual es una propiedad física muy importante, dado que, brinda información sobre la composición estructural de las superficies y de los objetos en la imagen. En este trabajo se realiza un clasificador para las bases de datos: KTH-TIPS-2B (KT2B), Describable Textures Dataset (DTD) y Flickr Material Database (FMD). Y se estudia la adaptabilidad del aprendizaje profundo con la transformada wavelet, en particular una aproximación a la arquitectura Wavelet CNN [6]. Además, se utiliza una metodología empírica y experimental en el desarrollo e implementación de la red neuronal convolucional (CNN) y el análisis wavelet, ambas como métodos de extracción de características. La combinación de estos métodos permite lograr un rendimiento de clasificación aceptable. En la base de datos KT2B se logra un 96 %, en la base de datos DTD un 34 % y por último en FMD se obtiene un 30 % de exactitud. Las gráficas de aprendizaje reflejan que los tres conjuntos de datos muestran una generalización de aprendizaje en las primeras épocas de entrenamiento. Para pequeños conjuntos de imágenes con texturas se recomienda la fusión del aprendizaje profundo con el análisis wavelet. Debido a la limitación de aprendizaje sobre información espectral que se pierde en la CNNs convencionales. Además, es información útil que permite mejorar el rendimiento de clasificación. Los resultados muestran que es posible integrar esta metodología en el desarrollo tecnológico de aplicaciones, como tareas de clasificación o restauración de imágenes y detección de objetos.

Palabras clave: Aprendizaje profundo, clasificación de texturas, métodos de extracción de características, redes neuronales convolucionales, transformada wavelet.

System for Classification of Textures in Images Using Deep Learning and Wavelet Characteristics

Abstract. The characterization of textures in images has become an analysis tool in the area of computer vision. Texture in visual perception is a fundamental physical property since it provides information about the structural composition of surfaces and objects in the image. The aim is to develop a classifier for the databases: KTH-TIPS-2B (KT2B), Describable Textures Dataset (DTD), and Flickr Material Database (FMD). Furthermore, the adaptivity of deep learning with wavelet transform is studied using an approach of Wavelet CNN [6]. An empirical and experimental methodology is used to develop and implement convolutional neural network (CNN) and wavelet analysis, both as feature extraction methods. The combination of both methods allows achieving acceptable classification performance. In the KT2B database with 96%, in the DTD database with 34%, and finally in FMD with 30% accuracy. The learning graphs reflect that all three datasets show a generalization of learning in the early training epochs. For small sets of images with textures, the fusion of deep learning with wavelet analysis is recommended due to the limitation of learning about spectral information lost in conventional CNNs, helpful information that allows for improved classification performance. The results show that it will be possible to integrate this methodology in the technological development of applications, such as image classification or restoration tasks and object detection.

Keywords: Convolutional neural network, deep learning, feature extraction methods, wavelet transform, texture classification.

1. Introducción

La percepción visual es una capacidad humana que permite reconocer la textura de objetos a simple vista, donde interviene el sistema óptico y el sistema nervioso. Los cuales son capaces de captar la información visual, procesarla y obtener un significado, para poder interpretar y comprender de que esta compuesto el objeto. Por otro lado, el análisis de textura dentro del aprendizaje máquina juega un papel importante en tareas de clasificación, detección y localización de objetos.

Este tipo de análisis tiene algunas áreas de aplicación, como el diagnóstico médico asistido por computadora, reconocimiento de frutas utilizando inteligencia artificial, localización y detección en la navegación aérea con drones, por mencionar algunas. En el área de procesamiento de imágenes, se puede definir la textura a partir de los píxeles vecinos y de la distribución de la intensidad sobre la imagen [16].

Además, existen algunos métodos de clasificación para el análisis de la textura como estadísticos, geométricos, de modelo y espectrales. Por otro lado, los métodos espectrales describen la textura en el dominio de la frecuencia. Se basan en la descomposición de una señal en términos de funciones base y utilizan los coeficientes de expansión como elementos del vector de características.

Fig. 1. Proceso del aprendizaje profundo.

Este trabajo se centra en la clasificación de la textura en imágenes, particularmente en conocer la información estructural de las superficies y de los objetos que se encuentran en el plano de imagen. La base del sistema de clasificación es una aproximación a la arquitectura Wavelet CNN, la cual fue propuesta en [6] para clasificación de texturas y tareas sobre etiquetado múltiple en relación con el contenido de la imagen.

La implementación de este sistema se desarrolla con la fusión de dos enfoques; utilizando el dominio espacial, específicamente las redes neuronales convolucionales (CNN por sus siglas en ingles) y el dominio espectral, la transformada wavelet de Haar [9, 3, 11]. Internamente este sistema se divide en dos etapas: la primera corresponde a la extracción de características y la segunda a la etapa de clasificación.

Con respecto a la fase de extracción de características, el tensor creado tiene un conjunto de parámetros numéricos que describen el contenido de la imagen, como el color, la textura o la forma del objeto. Por lo tanto, la etapa de extracción de características es importante para el éxito general de cualquier sistema de clasificación y reconocimiento en imágenes. Las principales contribuciones del trabajo se resumen a continuación:

– Se propone una metodología para mejorar y evaluar el modelo de aprendizaje. Además, se valida con nuevo conjunto de imágenes.
– Se diseña e implementa un sistema de clasificación de texturas en imágenes para evaluar la adaptabilidad del aprendizaje profundo con el análisis wavelet.

En particular, se demuestra que la combinación de ambos métodos de extracción de características (CNN y la transformada wavelet de Haar), alcanzan precisiones competitivas a lo reportado en la literatura, con un número significativamente menor de parámetros entrenables que al utilizar solo un método. Como resultado, el modelo es más fácil de entrenar, generaliza su aprendizaje a la combinación de información, y tiene un menor costo computacional.

El resto del documento está organizado de la siguiente manera en la Sección 2 se muestran los trabajos relacionados, la Sección 3 introduce la metodología para abordar el problema de clasificación de la textura. Posteriormente en la sección 4 se muestran los resultados con los tres conjuntos de datos, que se han usado para probar nuestro enfoque y la parte experimental. Finalmente, en la Sección 5 se presentan las conclusiones.

Fig. 2. Imágenes de ejemplo del conjunto de datos KTH-TIPS-2B.

2. Estado del arte

En el diseño de un sistema de recuperación para imágenes, es fundamental hacer un análisis de las características sobre el contenido del plano de imagen. Para esto en [16], los autores mencionan que la calidad del sistema depende, en primer lugar, de los vectores de características utilizados, además, presentan un estudio de las técnicas más comunes de extracción y representación de las características, donde brindan una clasificación en función de las mismas características, como por color, textura o forma.

En particular, al pensar en la búsqueda por textura, se tienen diferentes métodos de extracción, que en opinión del autor se clasifican en estadísticos, geométricos, de modelo y espectrales. Por otra parte, se menciona que el uso de un enfoque espectral combinado con otros métodos de extracción, mejora los resultados en la solución de problemas sobre clasificación y reconocimiento de texturas.

En cuanto al aprendizaje profundo en la ultima década se ha posicionado como una nueva solución en áreas de la robótica [12], visión computacional [1] y el lenguaje natural [15]. En particular, las redes neuronales convolucionales son una categoría del aprendizaje profundo, ya que se adaptan al análisis de objetos mediante el aprendizaje y la extracción de características complejas.

Por otro lado, aunque la CNN es un extractor universal, en la práctica, no esta claro si la CNN puede aprender a realizar análisis espectrales. Para tener este enfoque dentro de la CNN, en [2] los autores proponen una arquitectura llamada Textura CNN. Su idea se centra en que la información extraída por las capas convolucionales es de menor importancia en el análisis de la textura.

En consecuencia, utilizan una métrica estadística de energía en la etapa de extracción de características. Esta información se concatena con la etapa de clasificación, la capa totalmente conectada. En concreto, la arquitectura muestra una mejora en el rendimiento y una reducción en el costo computacional.

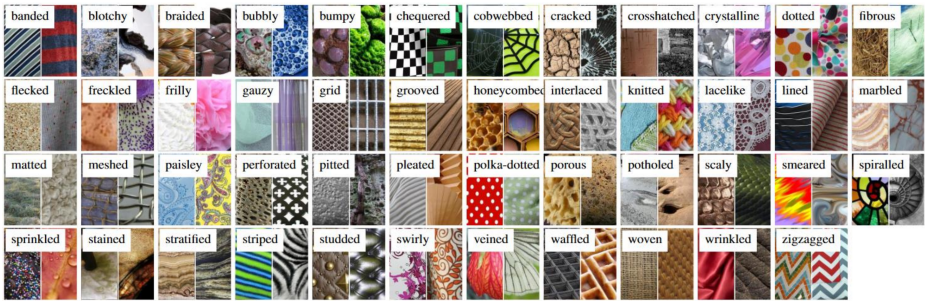


Fig. 3. Imágenes de ejemplo del conjunto de datos DTD.

Fig. 4. Imágenes de ejemplo del conjunto de datos FMD.

Los enfoques espaciales y espectrales son dos de los principales métodos para las tareas de procesamiento de imágenes. Dentro de esta linea de investigación, en [6] los autores proponen una novedosa arquitectura CNN, llamada Wavelet CNN, que combina un análisis multiresolución y el aprendizaje de la CNN en un modelo. Basándose en esta idea, complementan las partes que faltan con el análisis multiresolución mediante la transformada wavelet de Haar en su representación bidimensional, y la integran como componentes adicionales en toda la arquitectura.

La arquitectura Wavelet CNN permite utilizar información espectral que se pierde en su mayoría en las CNN convencionales, pero que es información útil en la mayoría de las tareas de procesamiento de imágenes. El rendimiento alcanzado en la clasificación de texturas y etiquetado de imágenes, muestran una mayor exactitud de clasificación que al utilizar AlexNet, donde unicamente utilizan las CNN convencionales. Así como, reducir el numero de parámetros a entrenar.

3. Materiales y métodos

El aprendizaje profundo es un subcampo del aprendizaje automático, una nueva manera de aprender características a partir de los datos, como texto, audio e imágenes [8]. El término profundo en esta área no hace referencia a ningún tipo de arquitectura; más bien, representa la idea de capas sucesivas de representaciones en diferentes niveles.

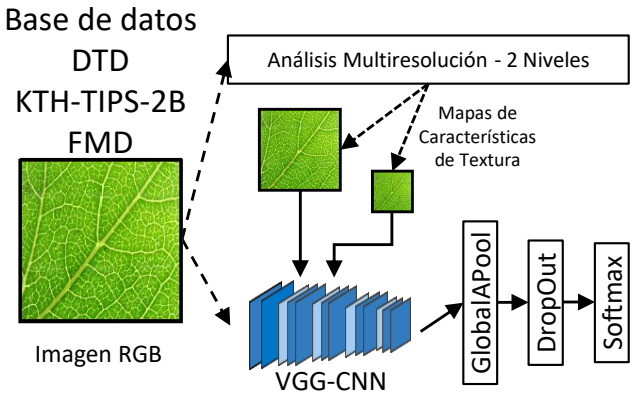
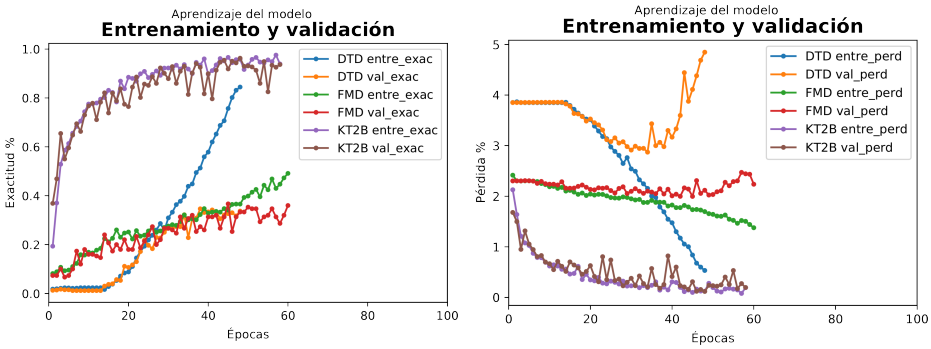


Fig. 5. Arquitectura para el sistema de clasificación de imágenes con textura.



(a) Exactitud del modelo.

(b) Pérdida del modelo.

Fig. 6. Evaluación de las métricas de exactitud y pérdida para los conjuntos de entrenamiento y validación.

Estas nuevas representaciones son cada vez más significativas. Por otro lado, dentro del aprendizaje profundo están la redes neuronales convolucionales o CNN, que son un tipo especializado de red neuronal para el procesamiento de datos. El nombre red neuronal convolucional indica que la red emplea una operación matemática llamada convolución.

La convolución es un tipo especializado de operación lineal, que es utilizada en lugar de la multiplicación general de matrices dentro de la red. La metodología propuesta para mejorar el rendimiento sobre el modelo de aprendizaje, se resume en tres etapas, las cuales se muestran en la Fig. 1. Cada etapa se describe a continuación.

Para la primer etapa, se necesita seleccionar los datos con los cuales vamos a generalizar el conocimiento del modelo, en nuestro caso se utilizan imágenes de tres conjuntos de datos que existen en la literatura, KTH-TIPS-2B, DTD y FMD. Los cuales contienen imágenes con diferentes texturas y materiales en condiciones naturales. Estos conjuntos de datos serán descritos a continuación.

Tabla 1. Capacidad de aprendizaje en las tres etapas de evaluación.

	Entrenamiento	Validación	Prueba
Exactitud DTD	0.3974	0.3221	0.3451
Exactitud FMD	0.3414	0.3667	0.3000
Exactitud KT2B	0.9580	0.9629	0.9678

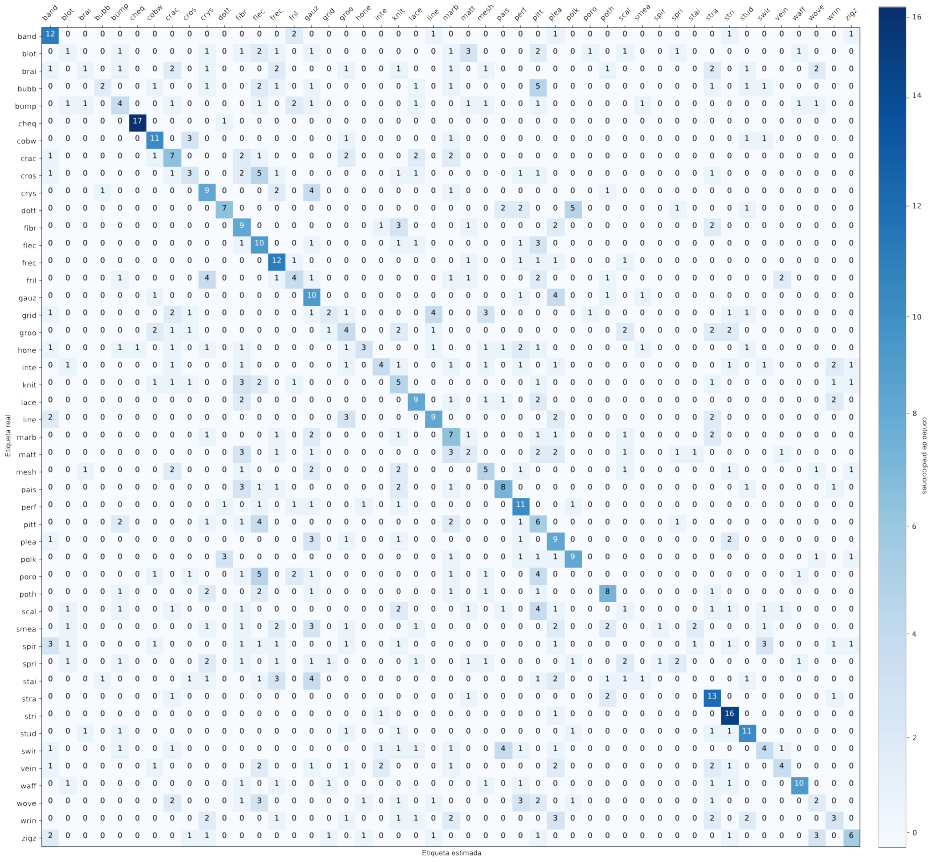


Fig. 7. Matriz de confusión para el conjunto de datos DTD.

3.1. Conjunto de datos

El primer conjunto de datos seleccionado es KTH-TIPS-2B (KT2B) el cual contiene 432 imágenes clasificadas en 11 clases. Cada clase consta de cuatro muestras y cada muestra tiene 108 imágenes [7]. La Fig. 2 ilustra las 11 clases con 4 muestras aleatorias. El segundo conjunto de datos es Describable Textures Dataset (DTD) el cual contiene 47 clases de 120 imágenes en la naturaleza, esto significa que las imágenes fueron adquiridas en condiciones no controladas [5]. Este conjunto de datos incluye 10 divisiones disponibles con 40 imágenes de entrenamiento, 40 imágenes de validación, y 40 imágenes de prueba para cada clase.

Tabla 2. Resultados de clasificación y comparación con otras arquitecturas del estado del arte, en términos de exactitud (%).

	AlexNet	T-CNN	Wavelet CNN	Propuesta
DTD	22.7	27.8	35.6	34.5
KT2B	48.3	49.6	63.7	96.7
FMD	–	–	–	30.0

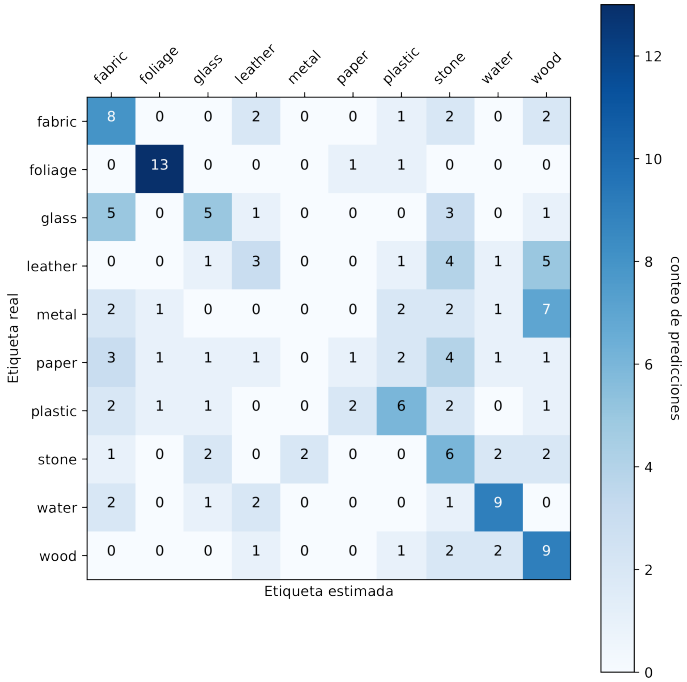


Fig. 8. Matriz de confusión para el conjunto de datos FMD.

En la Fig. 3 se muestra algunas imágenes de este conjunto. Por último, el tercer conjunto de datos seleccionado es Flickr Material Database (FMD). El cual esta construido con una gama de materiales comunes (por ejemplo vidrio, plástico, etc.). Cada imagen de esta base de datos (100 imágenes por categoría, 10 categorías) está seleccionada manualmente de Flickr.com (bajo licencia Creative Commons) para garantizar una variedad de condiciones de iluminación, composiciones, colores, textura y subtipos de materiales [13]. Se muestran algunas imágenes de esta base de datos en la Fig. 4.

3.2. Preparación de datos

Dado que inicialmente las imágenes dentro de los conjuntos tienen diferente tamaño. En la etapa de preprocesamiento, las imágenes son redimensionadas a un tamaño de 300×300 para los tres conjuntos, KT2B, FMD y DTD. También, al utilizar una arquitectura CNN se debe procesar las entradas.

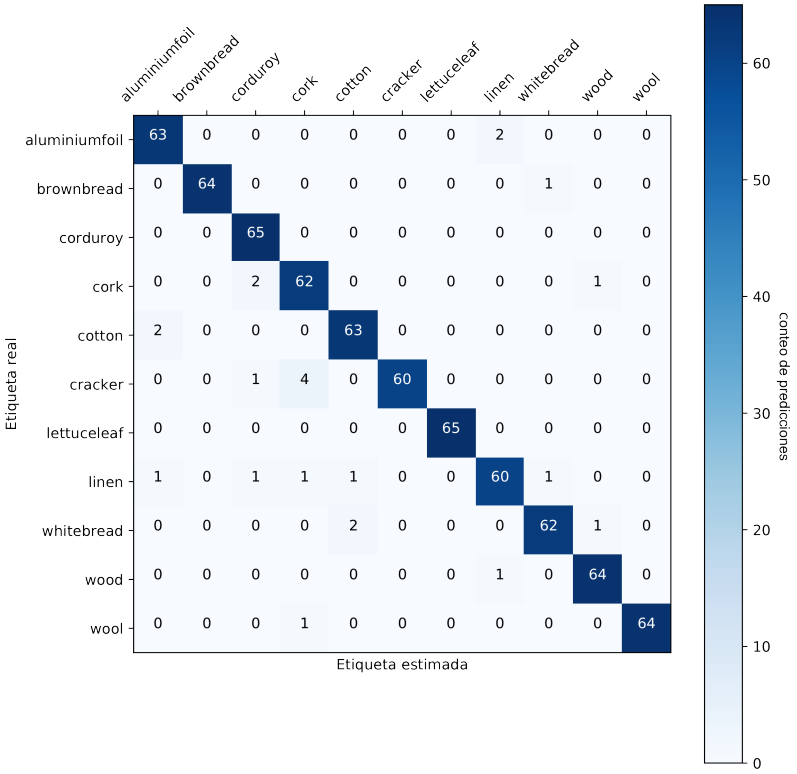


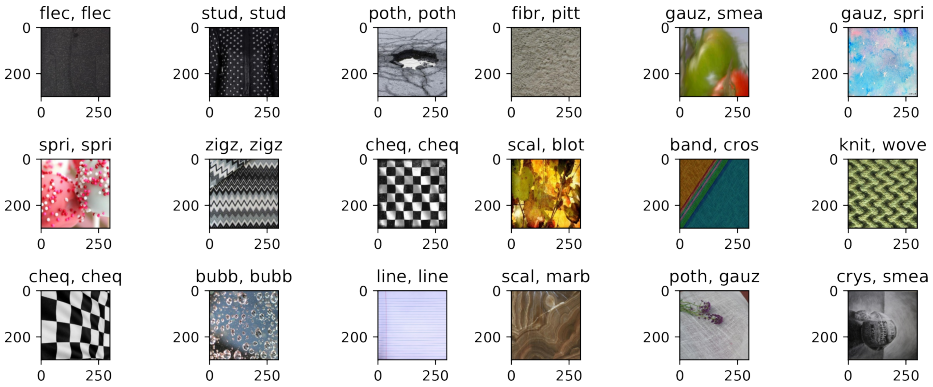
Fig. 9. Matriz de confusión para el conjunto de datos KT2B.

Por esa razón, al trabajar con imágenes, es conveniente normalizar los valores de los píxeles en un rango de 0 a 1, con la finalidad de que nuestro modelo converja rápidamente a un mínimo local, ya que las entradas con valores enteros grandes pueden ralentizar el proceso de aprendizaje.

3.3. Evaluación del modelo

En esta etapa se genera un conjunto de datos de manera aleatoria, el cual se divide en tres subconjuntos, uno de entrenamiento con el 75 % de las imágenes, otro de validación con el 15 % y el 15 % restante se utiliza para la etapa de prueba. Esta nueva distribución de imágenes para la evaluación del modelo durante el entrenamiento y validación, se aplicará a los tres conjuntos de datos, KT2B, FMD y DTD.

Después, se seleccionan los parámetros de entrenamiento, con la finalidad de ir evaluando el modelo durante cada época o iteración de aprendizaje. En la búsqueda de los filtros kernel los cuales establecen los rasgos característicos de cada textura, se debe seleccionar el que tenga un mejor desempeño de manera automática. Por lo tanto, este proceso permite ajustar y actualizar el modelo, conforme se esté entrenando la arquitectura.



(a) 292 Clasificadas correctamente. (b) 554 Clasificadas incorrectamente.

Fig. 10. Clasificación de texturas de manera aleatoria (de un total de 846 imágenes) utilizando el modelo de predicción DTD.

Por otro lado, al tener clases con una misma cantidad de imágenes, es posible utilizar una métrica de desempeño. En este caso, se puede calcular la exactitud, una métrica de desempeño muy relevante en tareas de clasificación. La exactitud (*accuracy*) es calculada como un porcentaje de imágenes que están correctamente etiquetadas por el modelo creado.

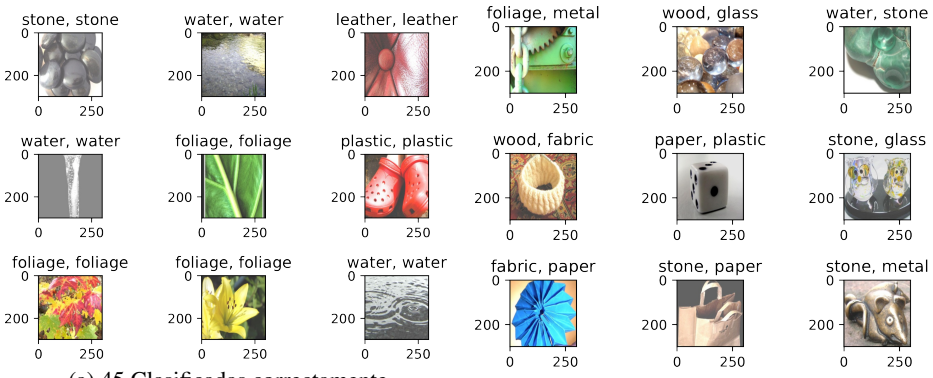
En relación con el rendimiento del modelo, una manera de determinar algunos patrones de error en la predicción o clasificación de las texturas, es utilizando la matriz de confusión múltiple. La cual es una tabla de N×N, que resume el nivel de éxito de las predicciones de un modelo de clasificación; es decir, la correlación entre la etiqueta y la clasificación del modelo. En este caso N representa el número de clases, un eje de la matriz de confusión es la etiqueta que el modelo predijo, y el otro es la etiqueta real.

3.4. Método para extracción de características

Por lo general, los diferentes métodos de extracción de características conducen a distintos elementos de información sobre la textura dentro de una imagen. Por lo que, retomando la idea central de la arquitectura Wavelet CNN [6], se decide diseñar una aproximación de la arquitectura. Dando como resultado un sistema híbrido para combinar los rasgos que genera la CNN a través de los filtros o kernel, junto con los atributos o mapas de características generados de manera manual con la transformada wavelet de Haar, mediante el análisis multiresolución a dos niveles de descomposición.

El diseño de la red incluye dos procesos separados (extracción de características mediante CNN y extracción de características usando el análisis wavelet) que posteriormente son fusionados en el entrenamiento del modelo. En el primer proceso, la imagen RGB preprocesada en la etapa de preparación de datos, entra como tensor a la arquitectura base VGG para conseguir ciertos patrones de manera automática.

Esto permitirá clasificar las texturas y diferenciar una de otra. El segundo proceso se realiza de manera adicional, dicho de otra manera, antes de ingresar la información espacial a la arquitectura CNN, debemos de generar nuevos datos sintéticos, que serán los atributos (coeficientes wavelet) o mapas de características en el dominio espectral.



(a) 45 Clasificadas correctamente. (b) Textura no detectada.

Fig. 11. Clasificación de texturas de manera aleatoria (de un total de 150 imágenes) utilizando el modelo de predicción FMD.

Conviene subrayar que, el proceso de análisis multiresolución en la etapa de descomposición, permite generar mapas de características que pueden ser adaptadas a los bloques convolucionales de la CNN base. Por lo tanto, con la fusión de estos procesos en la etapa de extracción de características se puede pasar a la siguiente etapa del aprendizaje profundo para la clasificación. En la Fig. 5, se muestra el sistema de clasificación con un enfoque espacial y con la fusión del enfoque espectral.

4. Resultados experimentales

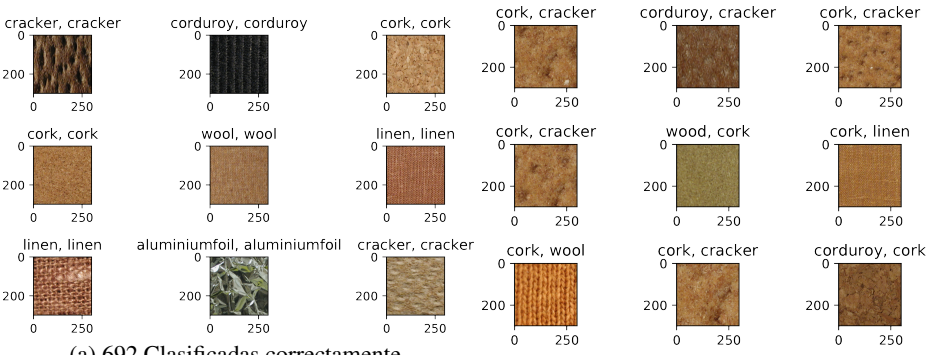
Para validar nuestro enfoque se ha utilizado tres conjuntos de datos KT2B, DTD y FMD. Dos de ellos (KT2B y DTD) son un caso especial de bases de datos de texturas porque contienen imágenes capturadas bajo condiciones no controladas.

4.1. Configuración experimental

En la Fig. 5 se ilustra el sistema de clasificación propuesto. La arquitectura esta diseñada de acuerdo a la red VGG16 [5], internamente tiene 5 bloques convolucionales con kernels de tamaño 3×3 y un padding (same) de manera que la salida tenga el mismo tamaño que la entrada. Además, cada bloque convolucional contiene dos métodos de convolución Conv2D, el primero con un stride de 1 y el segundo con un stride de 2 para asegurar que la salida sea la mitad de tamaño que la entrada.

Esto permite que los bloques convolucionales puedan extraer las características de las texturas en el dominio espacial. Por otro lado, dado que la arquitectura VGG y el análisis multiresolución (etapa de descomposición) tienen la misma característica de reducción, es posible concatenar cada nivel de descomposición (información en el dominio espectral) con los mapas de características de cada bloque convolucional.

En cuanto a determinar los atributos característicos de cada textura, se decide utilizar la transformada wavelet de Haar a 2 niveles. Este factor de 2, depende de la posibilidad de disminuir la imagen en cuanto a su tamaño.



(a) 692 Clasificadas correctamente. (b) Textura no detectada.

Fig. 12. Clasificación de texturas de manera aleatoria (de un total de 150 imágenes) utilizando el modelo de predicción FMD.

También, es importante mencionar que este proceso se aplica para cada uno de los canales RGB de la imagen, realizando la descomposición individual por canal y al final los mapas encontrados son concatenados en un vector.

La nueva información es concatenada en un cubo de características (espectral y espacial), y esté debe ser transformado para cambiar su representación mediante el método GlobalAveragePooling2D. Este nuevo vector a su vez alimenta el método de regularización DropOut para evitar el sobreajuste antes de pasar por la ultima capa de predicción, Softmax.

4.2. Implementación

En cuanto a la implementación del sistema, se utiliza el lenguaje Python y el API de Keras con Tensorflow como Backend [4]. En resumen, el sistema de clasificación tiene un total de 5,441,866 parámetros entrenables o pesos sinápticos de aprendizaje.

También, en la selección de los hyperparámetros se establece un indice de aprendizaje de 0.001, un minibatch de 30, 500 épocas de entrenamiento para el aprendizaje, cuatro Callbacks API para mejorar el rendimiento del modelo (ModelChekpoint, EarlyStopping, CVLogger, ReduceLROnPlateau), además del optimizador Adam, que es una variante de Gradiente Descendente.

Por otro lado, para el procesamiento de imágenes se utiliza las librerías OpenCV, por su fácil manejo y adaptabilidad dentro de la programación. En el caso del método adicional, la transformada wavelet de Haar, usamos la librería Pywt [10].

4.3. Resultados y discusión

La Fig. 6 muestra el comportamiento de aprendizaje para los tres conjuntos de datos. El máximo local alcanzado para la métrica de exactitud se muestra en la Fig. 6(a). Para el conjunto DTD el valor máximo alcanzado se dá alrededor de la época 21, con un valor de 39.74 % en entrenamiento (DTD entre_exac). Y para validación (DTD val_exac) de un 32.21 %.

En el caso del conjunto FMD el máximo local se dá en la época 45, con una exactitud de 34.14 % en entrenamiento (FMD entre_exac) y un 36.67 % para validación (FMD val_exac). Por último, para el conjunto de KT2B el máximo local se dá en la época 48, con un mejor rendimiento, un 95.80 % de exactitud para entrenamiento (KT2B entre_exac) y 96.29 % (KT2B val_exac) para validación.

En la Fig. 6 (b) se ilustra la pérdida del modelo en cada uno de los tres conjuntos de datos. También, se muestra que existe un punto de divergencia entre los dos conjuntos que se están entrenando (entrenamiento y validación). Este punto coincide con el número de época donde se encontró el máximo local de exactitud.

La tabla 1 resume el rendimiento alcanzado por cada modelo para los conjuntos de datos DTD, FMD y KT2B. La exactitud alcanzada para el conjunto de prueba en KT2B es del 96 %, FMD con un 30 % y para DTD es del 34 %. Estos resultados son muy importantes, dado que son imágenes que nunca ha visto el modelo. Por otro lado, a pesar de que las métricas en entrenamiento y validación se dan en diferentes tiempos, se muestra un comportamiento homogéneo con los datos de prueba.

Así que, existe una generalización del conocimiento entre los tres conjuntos, entrenamiento, validación y prueba. Con el fin de evaluar el desempeño de clasificación de nuestro modelo, se determina utilizar la matriz de confusión múltiple. Conviene subrayar, que existe una relación del conjunto de prueba con los resultados de la matriz de confusión. Para DTD en la Fig. 7, muestra una diagonal azul difícil de percibir y el mapa de calor aún distribuido en algunas zonas, esto indica que el modelo desarrollado aún encuentra semejanza en la mayoría de las clases.

En particular, para el conjunto de imágenes FMD, ver Fig. 8, la diagonal azul también es muy difícil de percibir. Se observa que en el centro de la matriz no existe una predicción correcta, ni con la etiqueta real y con ninguna otra clase, por lo que no se completa la diagonal. También, se encuentra que la mayor parte del mapa de calor esta distribuida a la derecha.

En cambio para el conjunto de datos KT2B, ver Fig. 9, muestra que existe una diagonal de color azul completa, esto resume que hay una tendencia positiva de predicción con respecto a la etiqueta original de las texturas KT2B. Las figuras, Fig. 10, 11 y 12 muestran algunas imágenes del conjunto de pruebas.

Estas imágenes son evaluadas por cada modelo, en la parte superior, se puede ver la etiqueta real y la predicción. En concreto, permite tener una idea visual sobre los rasgos de clasificación, y la correlación entre clases. La tabla 2, resume el rendimiento alcanzado de nuestro sistema de clasificación, así como una comparativa contra AlexNet, Textura CNN y Wavelet CNN [6][2].

Adicionalmente, se propone el conjunto de datos FMD para validar el modelo de aprendizaje. Por lo tanto, acorde a los resultados se puede ver que el modelo brinda una generalización de aprendizaje hacia otros conjuntos de datos.

5. Conclusión

En este trabajo se estudia la posibilidad de incorporar el análisis espectral a la arquitectura CNN, de acuerdo con algunas arquitecturas reportadas en la literatura.

Este novedoso sistema de clasificación, brinda un nuevo enfoque en la reestructuración de las capas convolucionales, la forma de generalizar el aprendizaje y la reducción del mapa de características.

También, se representa como una arquitectura de tres entradas - un modelo, dado que se alimenta con la información espacial y los dos mapas de características en el dominio espectral. En particular, con la fusión de los mapas de características creados de manera adicional, no limitamos el aprendizaje a las características espectrales. También, se demuestra que el modelo utilizado logra una mejor exactitud para la clasificación de la textura, con un número menor de parámetros a entrenar que los modelos existentes.

Además, los resultados mostraron que las características de textura creadas de manera adicional, podrán ser de ayuda para las arquitecturas CNN, especialmente en el caso de pequeños conjuntos de datos. Este enfoque, permitirá en un futuro probar otras características de textura y de arquitecturas CNN, en aplicaciones de reconocimiento de patrones en la restauración de imágenes, tareas de clasificación y detección de objetos.

Referencias

1. Alcalá-Rmz, V., Maeda-Gutiérrez, V., Zanella-Calzada, L. A., Valladares-Salgado, A., Celaya-Padilla, J. M., Galván-Tejada, C. E.: Convolutional Neural Network for Classification of Diabetic Retinopathy Grade. Advances in Soft Computing, Mexican International Conference on Artificial Intelligence, vol. 12468, pp. 104–118 (2020) doi: 10.1007/978-3-030-60884-2_8
2. Andrearczyk, V., Whelan, P. F.: Using filter banks in convolutional neural networks for texture classification. Computer Science, Computer Vision and Pattern Recognition, vol. 84, pp. 63–69 (2016) doi: 10.48550/ARXIV.1601.02919
3. Bengio, Y., Goodfellow, I., Courville, A.: Deep learning. MIT press, vol. 1 (2017)
4. Chollet, F.: Deep learning with Python. vol. 361 (2018)
5. Cimpoi, M., Maji, S., Kokkinos, I., Mohamed, S., Vedaldi, A.: Describing textures in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3606–3613 (2014)
6. Fujieda, S., Takayama, K., Hachisuka, T.: Wavelet convolutional neural networks (2018)
7. Hayman, E., Caputo, B., Fritz, M., Eklundh, J. O.: On the significance of real-world conditions for material classification. Computer Vision, European conference on computer vision, vol 3024, pp. 253–266 (2004) doi: 10.1007/978-3-540-24673-2_21
8. LeCun, Y.: Generalization and network design strategies. Connectionism in perspective, vol. 19, pp. 143–155 (1989)
9. LeCun, Y., Bengio, J., Hinton, G.: Deep learning. Nature, vol. 521, pp. 436–444 (2015)
10. Lee, G., Gommers, R., Waselewski, F., Wohlfahrt, K., O'Leary, A.: PyWavelets: A Python package for wavelet analysis. Journal of Open Source Software, vol. 4, no. 36, pp. 1237 (2019) doi: 10.21105/joss.01237.36
11. Mallat, S. G.: A theory for multiresolution signal decomposition: The wavelet representation. In: IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 11, no. 7, pp. 674–693 (1989) doi: 10.1109/34.192463
12. Rojas-Perez, L. O., Martinez-Carranza, J.: Autonomous Drone Racing with an Opponent: A First Approach. Computación y Sistemas, vol. 24, no. 3 (2020)
13. Sharan, L., Rosenholtz, R., Adelson, E.: Material perception: What can you see in a brief glance? Journal of Vision, vol. 9, no 8, pp. 784 (2009) doi: 10.1167/9.8.784

14. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition, pp. 1409–1556 (2014) doi: 10.48550/arXiv.1409.1556
15. Tapia-Téllez, J. M., Escalante, H. J.: Data augmentation with transformers for text classification. Lecture Notes in Computer Science, Mexican International Conference on Artificial Intelligence, vol. 12469, pp. 247–259 (2020) doi: 10.1007/978-3-030-60887-3_22
16. Vassilieva, N. S.: Content-based image retrieval methods. Programming and Computer Software, vol. 35, no. 3, pp. 158–180 (2009) doi: 10.1134/s036 1768809030049

ISSN 1870-4069

Sistema prototipo de monitoreo subacuático automático de peces por visión estereoscópica y aprendizaje profundo

Héctor Carlos Aranda-Martínez[1], Nidiyare Hevia-Montiel[2]

[1] Universidad Nacional Autónoma de México,
Posgrado en Ciencia e Ingeniería de la Computación,
México

[2] Universidad Nacional Autónoma de México,
Unidad Académica del Instituto de Investigaciones en
Matemáticas Aplicadas y en Sistemas en el Estado de Yucatán,
México

carlosaranda@comunidad.unam.mx,
nidiyare.hevia@iimas.unam.mx

Resumen. En este trabajo se presenta la propuesta de un prototipo de monitoreo subacuático mediante visión estereoscópica, técnicas de visión computacional, y aprendizaje computacional para obtener información espacial de peces (posición y longitud) de forma automatizada: su detección y localización en ambas vistas es generada por una Red Neural Convolucional tipo YOLO (You Only Look Once); mediante geometría epipolar se reconstruye la posición espacial de puntos de interés para rastreo de cada pez; finalmente se estima la posición y longitud de cada pez localizado. En un entorno controlado se ha obtenido un rendimiento satisfactorio, con errores de entre el 0.75 y el 2.5 % de la longitud real.

Palabras clave: Monitoreo subacuático, visión estereoscópica, aprendizaje profundo, geometría epipolar.

Prototype System for Automatic Underwater Monitoring of Fish by Stereoscopic Vision and Deep Learning

Abstract. This paper presents a fish underwater monitoring prototype using stereoscopic vision, computer vision, and deep learning techniques to obtain spatial information of the fish (position and length) in an automated way: fish detection and localization from the two steroscopic views is generated by a YOLO (You Only Look Once) Convolutional Neural Network; by epipolar geometry, spatial position of points of interest for fish tracking is reconstructed; position and length of each detected fish is estimated. For controlled environment tests, results show a satisfactory performance with an error range between 0.75 and 2.5% of real length.

Keywords: Underwater monitoring, stereo vision, deep learning, epipolar geometry.

1. Introducción

La obtención de datos sobre la fauna submarina ha sido siempre de gran interés, ya sea económico o científico. Especialmente, hablando de intereses económicos, en la industria acuícola es muy importante conocer datos estadísticos de los peces (peso, dimensiones y conteo) [10]. Sin embargo, estas tareas son todavía realizadas en gran medida manualmente, por lo que su automatización tendría un gran impacto en este sector productivo. En el monitoreo subacuático por video (o imágenes) es posible utilizar una o más cámaras. Para el caso de utilizar una cámara se tiene la dificultad de realizar mediciones del entorno sin elementos de referencia, que permitan recuperar la información espacial del objeto de interés.

El uso de trampas por donde se hacen pasar los peces [15], o de elementos de dimensiones conocidas en las imágenes [5] son algunas medidas utilizadas para sobrepasar tal dificultad. Para el caso de un sistema de dos cámaras, o estereoscópico, la ventaja sobre el sistema monocular es la cantidad de información que se puede obtener del entorno, por lo que hacer mediciones sobre el mismo genera mejores resultados aunque aumenta la complejidad del procesamiento [14, 21]. Uno de los principales inconvenientes para su completa automatización es la detección de los puntos de interés para hacer las mediciones pertinentes.

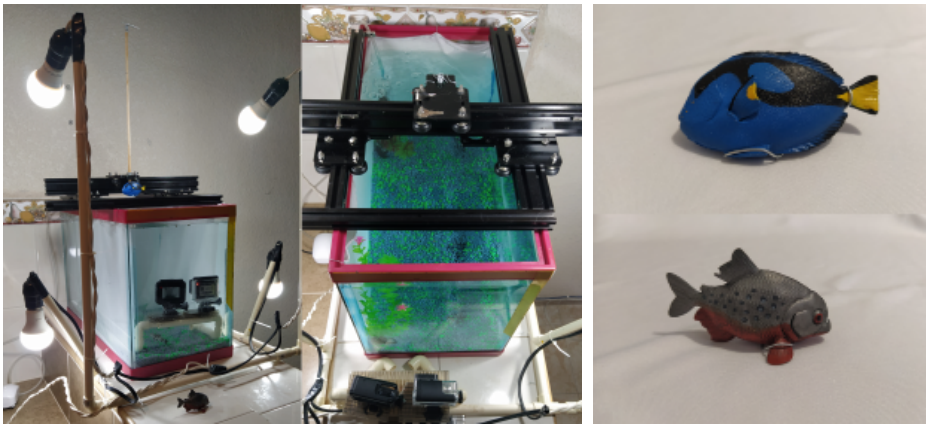
Shortis et al. [19] realizaron un compendio de técnicas y métodos de visión computacional en el monitoreo subacuático. Para la tarea de medición de características de peces, algunos procedimientos se realizan de forma manual [8, 21] o semiautomática [16]. Rodriguez et al. [14] proponen un método de medición de peces por visión estereoscópica que alcanza un error relativo promedio del $10\,\%$. Muñoz et al. [9] proponen un método automático para medir las longitudes de atunes en ambientes reales, utilizando visión estereoscópica y un modelo deformable de la forma del pez, logrando una evaluación del Kolmogorov-Smirnov p-value de $0{,}0183$.

Posteriormente Shi et al. [17] desarrollaron un método basado en detección de movimiento y generación del mapa de disparidad para medir longitudes de peces logrando un error cuadrático promedio relativo del $1.22\,\%$. En este trabajo se desarrolla la propuesta de un prototipo estereoscópico para monitoreo subacuático (de ahora en adelante identificado como PEMS), que a partir de datos de video pueda detectar, localizar y estimar las longitudes de los peces que se encuentran en escena de manera automática. En la primera parte se explican los métodos y materiales utilizados; después el diseño y realización de los experimentos; y finalmente se mostrarán los resultados obtenidos.

2. Materiales y métodos

2.1. Sistema físico de pruebas

Se cuentan con dos cámaras GoPro Hero®, capaces de sumergirse bajo el agua. Se configuraron para trabajar con un campo de visión lineal, a 1080p y 30 cuadros por segundo. Se trabajará en un ambiente controlado (dimensiones, posiciones, luminosidad y turbidez del agua, ver figura 1a) que se compone de una pecera de 26 cm de ancho, 35.5 cm de alto y 51 cm de largo, 4 fuentes de luz y un sistema de rieles.



(a) (b)

Fig. 1. Montaje del ambiente controlado. a) Sistema de rieles y de iluminación para controlar posición de phantoms y uniformidad lumínica. b) Phantoms utilizados para las pruebas: pez cirujano (Paracanthurus hepatus, arriba) y piraña (Serrasalmus nattereri, abajo).

Con los últimos dos elementos se controla la uniformidad lumínica y la posición de los phantoms usados en las pruebas, respectivamente. En la figura 2 se esquematiza la posición de las cámaras C_1 y C_2, así como de las 30 locaciones distintas donde se colocarán los phantoms, marcadas por las **X**. La calibración del sistema de cámaras estereoscópico toma como referencia a la cámara izquierda (C_1), por lo que se utilizarán sus coordenadas para encontrar la posición relativa de cada punto al PEMS.

2.2. Obtención de datos

La base de datos será conformada por archivos de video de 1920×1080 píxeles, en espacio de color RGB. Estos archivos deben contener un evento visual de sincronización, es decir, un evento puntual que pueda ser captado por ambas cámaras y que sirva para sincronizar los archivos. Para esta sincronización se utilizó un evento visual creado por el encendido y apagado de una luz, mediante el cual se ajustó la reproducción de los video. La decisión de utilizar video en lugar de imágenes se fundamenta en el deseo de trabajar con organismos vivos que se mueven en su entorno (para los que no siempre se tendrá la mejor fotografía), y la posibilidad de ajustar los errores en las mediciones según la cantidad de información recibida.

2.3. Calibración

Tal como lo propone [18], se utilizó un patrón de ajedrez de dimensiones conocidas para hacer la calibración subacuática. Se utilizó el algoritmo propuesto por Zhang [23]. Este algoritmo toma un conjunto de puntos $m = [u, v]^T$ conocidos en la imagen de calibración correspondientes a puntos conocidos en el patrón de calibración (ver figura 3) $M = [X, Y, Z]^T$.

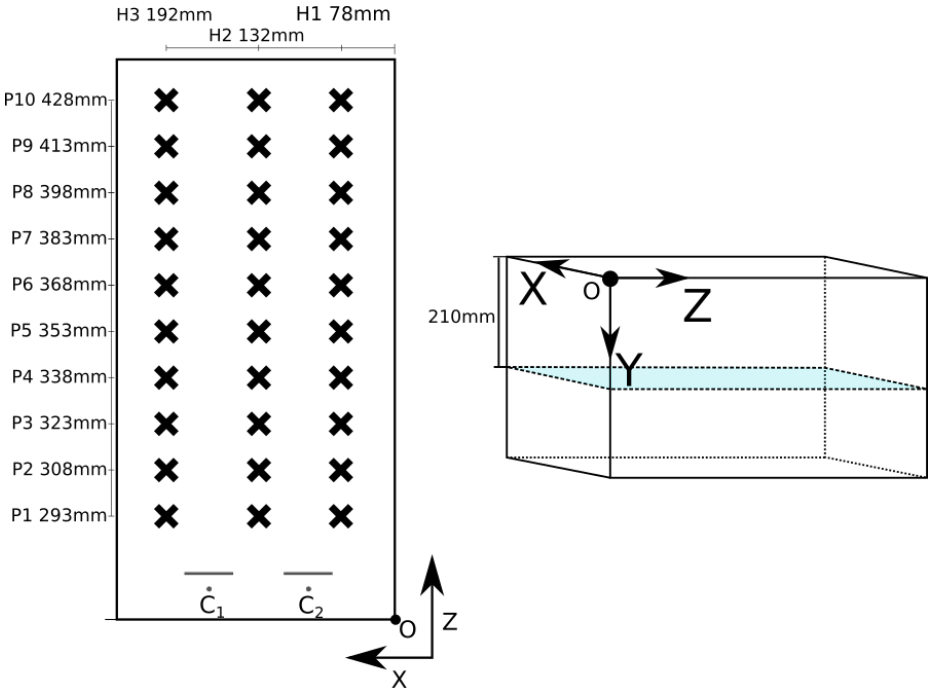


Fig. 2. Posiciones **X** identificadas por las coordenadas (H, P) y utilizadas durante las pruebas. El origen (O) del sistema de referencia espacial se encuentra en la esquina superior derecha de la pecera; el phantom se colocó en el eje Y a 210 mm; C_1 y C_2 son las cámaras izquierda y derecha.

Con estos puntos se estima la matriz de calibración interna A, la matriz de rotación R y el vector de traslación t de la cámara, de manera que se cumpla:

$$s\tilde{m} = A \left[R\, t\right] \tilde{M}, \tag{1}$$

donde s es un factor de escalamiento y $\tilde{m}$ y $\tilde{M}$ son los puntos m y M en coordenadas homogéneas. El objetivo del algoritmo es minimizar el error de proyección de los puntos. Los coeficientes de distorsión radial también son considerados dentro de la calibración.

Una vez hecha la calibración de cada cámara, se procede a calibrar el sistema de dos vistas, lo que implica encontrar:la matriz de rotación (R), que rota el sistema coordenado de la segunda cámara C_2 para que corresponda con la primera C_1; al vector de traslación (t), que traslada el sistema coordenado de C_2 al de C_1, la matriz esencial (E), que describe la relación entre puntos en los sistemas coordenados de ambas cámaras; y la matriz fundamental (F), que describe la relación entre los puntos en los planos de la imagen de ambas cámaras.

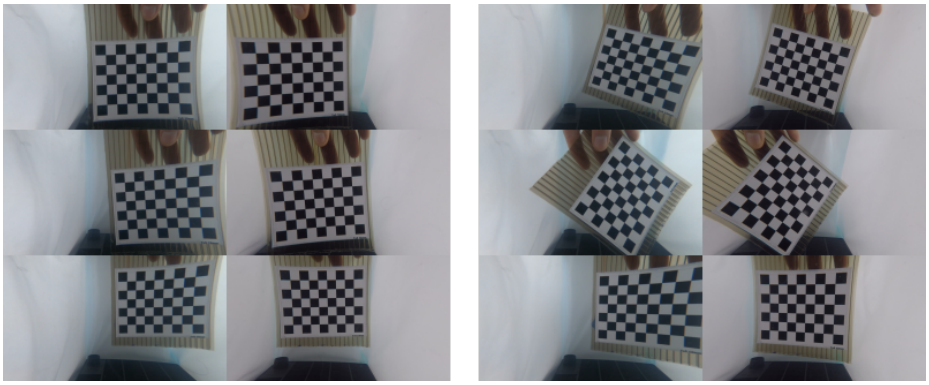


Fig. 3. Ejemplos de capturas subacuáticas del tablero de calibración.

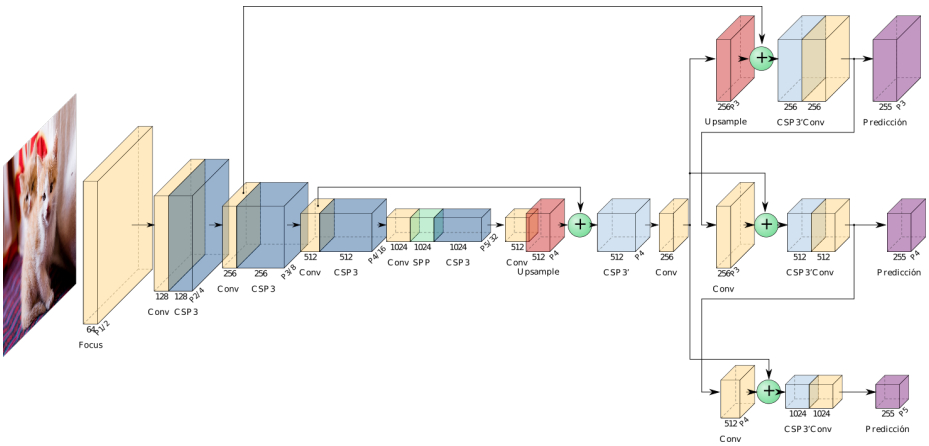


Fig. 4. Arquitectura Yolov5l.

2.4. Localización automática de peces

Para este proceso se optó por utilizar una red neuronal artificial tipo YOLO (You Only Look Once) [12], precisamente un modelo tipo YOLOv5 [7]. Estas arquitecturas de redes han demostrado tener desempeños tan buenos como para ser implementadas en sistemas de detección en tiempo real.

En comparativa con otra familia de arquitecturas llamada EfficientDet [20], la familia YOLOv5 tiene una rapidez de hasta 5 veces mayor sin disminuir su desempeño. Esto deja en evidencia que los tiempos de inferencia para una red tipo YOLOv5 permiten su uso en sistemas de detección en tiempo real.

Este grupo de arquitecturas cuenta con 4 predeterminadas: **s**, **m**, **l** y **x**. Cada una de ellas se diferencia por su profundidad, siendo la **s** la de menor cantidad de capas y la **x** la más profunda; en este trabajo se utilizó la configuración **l** (ver figura 4), que se compone de ciertos bloques que se reutilizan y se explican a continuación.

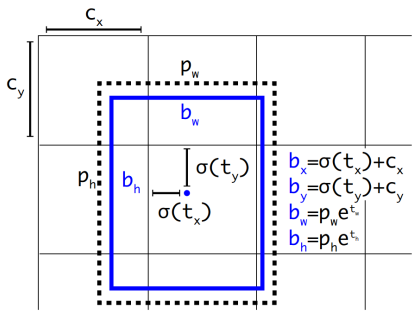


Fig. 5. Cálculo del centro y dimensiones de la caja de detección del objeto. c_x y c_y corresponden a las coordenadas de la celda responsable de la detección. p_w y p_h son las dimensiones predefinidas del anchor box utilizado. Obtenida de [12].

– **Focus.** Esta primer capa hace una reducción tipo SpaceToDepth, tal como se menciona en [13]. Este primer proceso es más rápido y tiene menor pérdida de información que otras capas donde solamente se hace una convolución con reducción de resolución.

– **Conv.** Se compone de una capa de convolución seguida de una normalización por lote (para el entrenamiento, según [6]), y una capa de activación SiLU [11] expresada por:

$$\text{silu}(x) = x * \sigma(x), \tag{2}$$

donde σ es la función sigmoide logística.

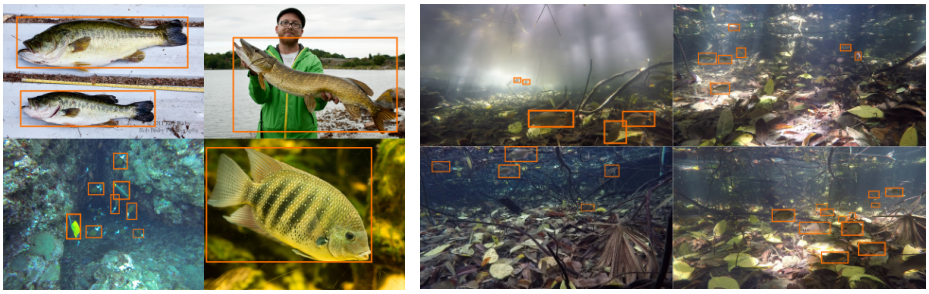
– **CSP 3 (Cross Stage Partial).** Este bloque está basado en la arquitectura propuesta por [22]. Se compone de una capa de cuello de botella (Bottleneck) en conjunto con tres operaciones de convolución. Estos bloques han demostrado disminuir el costo computacional sin sacrificar el desempeño de la red (incluso mejorando la precisión de tareas como la clasificación).

– **SSP (Spatial Pyramid Pooling).** Esta capa permite que la red acepte imágenes de cualquier dimensión, lo que permite manejar cualquier tipo de escalas y razones de aspectos (aspect ratios). Su implementación está basada en [4].

Finalmente, se tienen 3 capas de salida, cada una a diferentes resoluciones ($I3$, $I4$ y $I5$). Esto le permite a la red hacer detecciones de objetos muy pequeños ($I3$), medianos ($I4$) y grandes ($I5$). Cada celda (vector) de estos tensores se compone por 3 secciones formadas por el siguiente vector $(t_x, t_y, t_w, t_h, P_0, p^T)^T$, donde t_x y t_y son las predicciones del centro de la región predicha, t_w y t_h representan la altura y el ancho de la caja, P_0 es la probabilidad que la caja contenga un objeto y p es el vector de clasificación (que predice la clase a la que pertenece el objeto identificado).

Son 3 secciones, ya que se trabajan con 3 tamaños de cajas predefinidas (anchor boxes), y cada celda se encarga de ajustarlas todas.

Para la localización se considera la sección que contenga la mayor puntuación en p^T. Posteriormente se calculan las coordenadas relativas de la detección en la imagen original, como se muestra en la figura 5.



(a) (b)

Fig. 6. Ejemplos de imágenes de la base de datos. a) imágenes de Open IMages Dataset, b) imágenes de datos etiquetados por [1].

Después los resultados son filtrados por el método de supresión no máxima (Non-Maximum Suppresion), para eliminar las regiones propuestas que puedan representar el mismo objeto y estén superpuestas. Este proceso termina generando un arreglo de $n \times (5 + c)$, donde n es el número de detecciones válidas y c el número de clases (para el prototipo propuesto se consideró 1).

Para el entrenamiento se utilizaron 2767 imágenes (1883 de entrenamiento y 884 de prueba) provenientes de: Open IMages Dataset[3], figura 6a; y del conjunto de datos obtenido por [2] y procesado por [1], que está conformado por imágenes de peces en petenes del estado de Yucatán (figura 6b).

El entrenamiento se hizo en un equipo con tarjeta NVIDIA Tesla T4, 8 imágenes por lote, 1000 épocas, el tamaño de la imagen de entrenamiento y de inferencia de 448×448 px. Se utilizó transferencia de aprendizaje con los pesos del modelo YOLOv5m (pre-entrenado con la base de datos COCO val2017).

2.5. Puntos de interés

La red genera la región de interés (ROI) que contiene al pez de tal forma que tanto la boca como la aleta caudal tocan lados opuestos de esta. Esto nos permite tener una buena aproximación a las dimensiones (longitud) del pez. Se toman entonces como puntos de interés los correspondientes al punto medio de la altura de la ROI en cada lado de la misma (ver figura 7).

2.6. Reconstrucción tridimensional

Al hablar de reconstrucción 3D nos referimos a recuperar la información espacial de los puntos de interés encontrados en la escena, no debe interpretarse como la reconstrucción tridimensional del objeto completo.

La reconstrucción se basó en geometría epipolar, utilizando la información de la calibración de las cámaras. Es probable que para los puntos de interés seleccionados en ambas vistas no se cumpla que:

$$m_2^T F m_1 = 0, \tag{3}$$

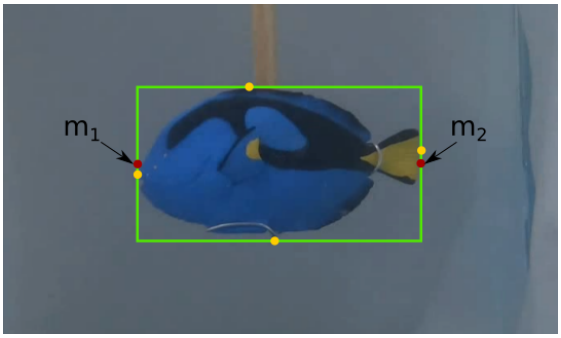


Fig. 7. En verde se muestra la región de interés que contiene al phantom de pez cirujano. Los puntos amarillos representan puntos de contacto del contorno del pez y el recuadro verde (correspondiente con la región de interés que lo contiene). Los puntos rojos m_1 y m_2 son los puntos de interés utilizados.

donde m_1 es el punto en la primera imagen y m_2 en la segunda. Esto puede ser debido a errores en la selección de los puntos o deficiencias en la calibración (cálculo de la matriz fundamental F). Para ello se hace una corrección de la siguiente manera:

1. Considerar que existen los puntos $\hat{m}_1$ y $\hat{m}_2$ tal que $\hat{m}_2^T F \hat{m}_1 = 0$, y que se encuentran cercanos a m_1 y m_2.
2. La distancia euclideana entre dos puntos está expresada por $d(p_i, p_j)$.
3. Definimos una función error que sirva para minimizar la distancia entre los puntos:

$$E(m_1, m_2) = d(m_1, \hat{m}_1)^2 + d(m_2, \hat{m}_2)^2. \tag{4}$$

Siempre que se cumpla $\hat{m}_2^T F \hat{m}_1 = 0$, donde $\hat{m}_1$ y $\hat{m}_2$ son los puntos corregidos.

Con los puntos corregidos se procede a hacer la triangulación de los mismos según el algoritmo descrito por [3]. En él, se genera una ecuación lineal de la forma $BX = 0$ a partir del conocimiento de las matrices de proyección de ambas cámaras (P_1 y P_2) y los puntos corregidos (como se describió anteriormente). El desarrollo de este sistema de ecuaciones considera la eliminación del factor de escala mediante el producto cruz $m_i \times (P_i M_i) = 0$. Esto genera las ecuaciones:

$$m_{i,x}(p_i^{3T} M) - (p_i^{1T} M) = 0, \tag{5}$$
$$m_{i,y}(p_i^{3T} M) - (p_i^{2T} M) = 0, \tag{6}$$
$$m_{i,x}(p_i^{2T} M) - m_{i,y}(p_i^{1T} M) = 0, \tag{7}$$

donde p_i^k es la k-ésima fila de la matriz de proyección P_i, y $m_i = (m_{i,x}, m_{i,y})^T$. De esto observamos que solamente dos de las 3 ecuaciones son linealmente independientes.

[3] https://storage.googleapis.com/openimages/web/index.html

Además, estas ecuaciones son lineales en términos de X, por lo que la ecuación $BX = 0$ tiene como valor de B:

$$B = \begin{bmatrix} m_{1,x}p_1^{3T} - p_1^{1T} \\ m_{1,y}p_1^{3T} - p_1^{2T} \\ m_{2,x}p_2^{3T} - p_2^{1T} \\ m_{2,y}p_2^{3T} - p_2^{1T} \end{bmatrix}. \tag{8}$$

La solución de dicha ecuación dará como resultado el punto espacial Q_w en coordenadas homogéneas.

2.7. Cálculo de la longitud del pez

La triangulación de los puntos anteriores nos permite estimar su posición espacial. Nuestro principal interés es hacer la medición de la longitud del pez localizado, por lo que se consideró que los puntos anteriormente obtenidos ($q_{w,\text{boca}}$ y $q_{w,\text{aleta}}$ en coordenadas no homogéneas) corresponden a los dos extremos horizontales del pez. Así, la longitud de cada pez puede ser estimada según:

$$L_{\text{estimada}} = ||q_{w,\text{boca}} - q_{w,\text{aleta}}||. \tag{9}$$

2.8. Validación de resultados

Los resultados del PEMS se validarán utilizando los datos conocidos de los tamaños de los peces utilizados, así como de las dimensiones del ambiente en el que se hacen las pruebas. Se toman en cuenta dos mediciones: 1) la longitud del pez, y 2) la posición relativa de éste respecto al sistema de cámaras.

Estas estimaciones se harán con base en los puntos de interés ($m_{i,1}$ y $m_{i,2}$) localizados en las imágenes y devueltos por el sistema de detección con los cuales se puede conocer su posición espacial ($M_{w,1}$ y $M_{w,2}$).

Con la información obtenida anteriormente es importante evaluar dos propiedades, considerando que ($M_{w,1}$ y $M_{w,2}$) se encuentran en extremos contrarios del pez. La primera corresponde a la longitud del pez (L_S), calculada de la siguiente manera:

$$L_S = ||M_{w,1} - M_{w,2}||. \tag{10}$$

La segunda corresponde con la distancia del punto espacial donde se estima que se encuentra el pez a la posición donde realmente se encuentra. Para esto se calcula el centro de masa del objeto $M_{w,c}$, considerando que ($M_{w,1}$ y $M_{w,2}$) se encuentran en extremos contrarios del pez, con la siguiente ecuación:

$$M_{w,c} = \frac{1}{2}(M_{w,1} + M_{w,2}). \tag{11}$$

Para evaluar estas métricas se proponen dos funciones de error:

$$e_L = \left| \left(1 - \frac{L_S}{L} \right) \right| \times 100, \tag{12}$$

$$e_D = ||M_{w,c} - M_w||, \tag{13}$$

Tabla 1. Se muestra la longitud estimada promedio y la desviación estándar promedio de cada posición (H, P) (en mm). Las longitudes reales son: 70 mm para el phantom de pez cirujano, y 66 mm para el de pez piraña.

	Cirujano			**Piraña**		
	H3	**H2**	**H1**	**H3**	**H2**	**H1**
P1	71.66± 0.47	71.38± 0.43	**72.17**± 0.93	67.90± 0.63	67.05± 0.31	66.99± 0.25
P2	**72.28**± 0.70	71.55± 0.45	70.43± 1.30	**67.91**± 0.40	**66.99**± 0.21	67.59± 0.29
P3	71.48± 0.70	**72.05**± 1.25	69.55± 0.55	66.96± 0.34	**67.54**± 0.24	**67.70**± 0.34
P4	70.48± 1.70	69.75± 0.71	69.46± 0.70	66.42± 0.43	67.42± 0.22	**66.06**± 0.25
P5	70.49± 0.61	70.12± 0.29	**69.36**± 0.59	**66.11**± 0.51	67.46± 0.32	66.60± 0.26
P6	71.11± 0.45	70.15± 0.50	68.80± 0.96	65.83± 0.37	67.28± 0.23	66.75± 0.18
P7	70.46± 0.87	70.40± 0.53	69.52± 1.33	66.62± 0.71	67.01± 0.32	67.09± 0.26
P8	69.52± 0.25	69.69± 1.39	69.74± 1.10	66.44± 0.60	67.20± 0.40	66.48± 0.48
P9	69.32± 0.50	70.21± 0.57	69.90± 1.21	67.65± 0.76	67.15± 0.44	66.66± 0.55
P10	**68.76**± 0.82	**69.66**± 1.06	70.10± 0.90	66.44± 0.69	67.03± 0.34	66.22± 0.20

donde L es la longitud real y M_w es la posición espacial relativa a las cámaras del phantom utilizado. e_L puede considerarse como un error porcentual que nos permite intuir rápidamente qué tan alejada está la estimación del valor real.

3. Resultados

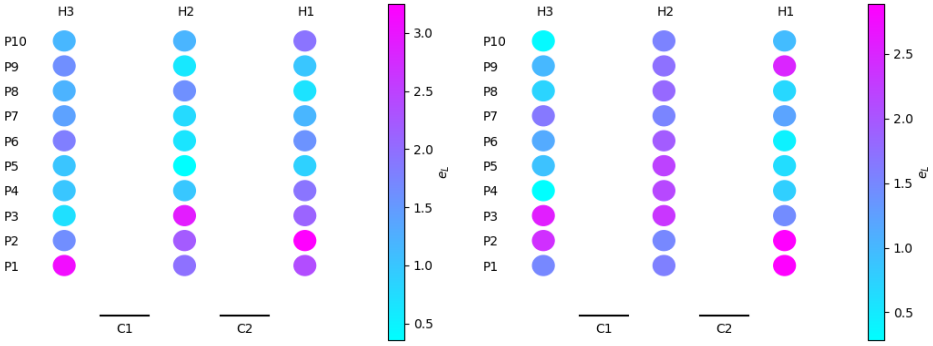
El procedimiento de obtención de datos fue el siguiente: calibrar el sistema estereoscópico de cámaras, colocar el phantom en una de las posiciones, obtener un registro de video de 5 segundos en dicha posición, cambiar a una nueva posición y repetir desde el paso 2 hasta agotarse las posiciones. Los datos de video fueron procesados por el PEMS que se encara de detectar, cuadro a cuadro, el phantom que se encuentra en la escena.

Por cada uno devuelve los puntos de interés $(m_{1,1}, m_{1,2})$ y $(m_{2,1}, m_{2,2})$ de cada vista basándose en la región encontrada anteriormente. Después hace la reconstrucción tridimensional de cada par de puntos de interés $M_{w,1}$ y $M_{w,2}$. Calcula la distancia que existe entre el par de puntos, esta es la longitud estimada del objeto L_S.

Y finalmente calcula los errores para validación e_L y e_D. Es importante mencionar que para este proceso, con las características del equipo utilizado, cada imagen toma 0.8 segundos en promedio en procesarse.

Se realizaron 5 repeticiones de las adquisiciones para cada phantom, de todas estas repeticiones se creó una sola tabla que contiene los valores promedio por cada posición (H, P) (ver la tabla 1).

Esta información está sintetizada en la figura 8, donde podemos darnos una idea de la zona espacial donde la exactitud de nuestro método es mayor para cada uno de los casos. Gracias al resultado anterior podemos darnos cuenta que la variación en las mediciones parece depender más de las posiciones en P que en H.



(a) Errores del phantom de pez cirujano.

(b) Errores del phantom de pez piraña.

Fig. 8. Representación gráfica de los errores por posición. Se muestra la posición relativa a las cámaras C_1 y C_2. Estos puntos corresponden con los esquematizados en la figura 2.

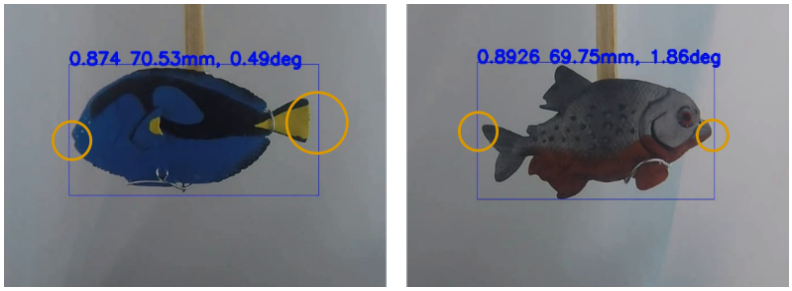


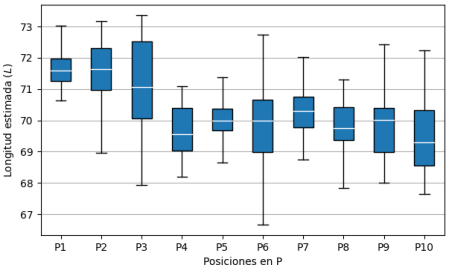
Fig. 9. Se muestran los errores en la localización de los peces en las imágenes. Para ambos casos existen cuadros donde la región de interés no se ajusta bien a la figura del pez. Estas diferencias, aunque parezcan pequeñas, pueden representar errores del orden de 2 o 3mm en las mediciones. En la parte superior de la detección se pueden apreciar tres valores, de izquierda a derecha tenemos: el valor de confianza de la detección, la longitud instantánea, el ángulo de inclinación del pez respecto al plano de la imagen.

Este resultado fue esperado por dos razones: 1) la posición espacial de las imágenes que se utilizaron para realizar la calibración corresponden con las posiciones de menor error (se explica más adelante); y 2) en ocasiones la red neuronal convolucional no devuelve una región que contenga perfectamente al pez, lo que genera errores en la posición de los puntos de interés (ver figura 9).

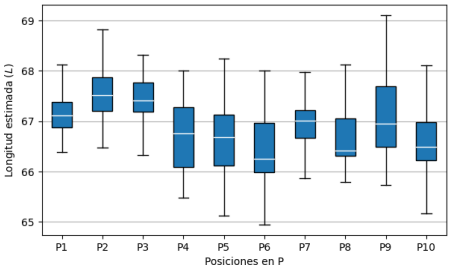
Para poder visualizar mejor el comportamiento de las estimaciones de las longitudes a lo largo de las posiciones en P, podemos revisar los resultados de la figura 10.

Como se espera el comportamiento en ambos casos es similar, comenzando con un error relativamente alto en posiciones cercanas a las cámaras y disminuyendo a medida que se aleja.

El por qué de este comportamiento puede ser explicado por la misma calibración, ya que el patrón de ajedrez utilizado fue posicionado a distancias mayores a los 35 cm de las cámaras (debido a las dimensiones de éste), es por esto que se esperaba obtener menor error en las mediciones cercanas a la región espacial de la calibración.

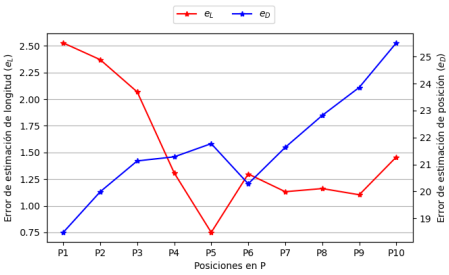


(a) Longitudes estimadas del phantom de pez cirujano.

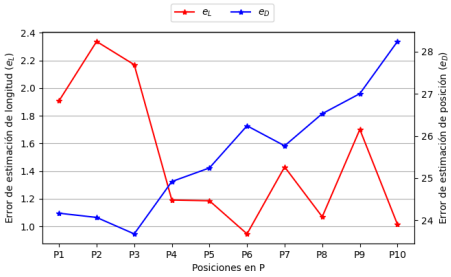


(b) Longitudes estimadas del phantom de pez piraña.

Fig. 10. Presentación de las mediciones de longitudes en gráficos de caja. Para este análisis se utilizaron las mediciones en las posiciones $H1$, $H2$ y $H3$ para cada posición en P. Recordemos que la longitud real del pez cirujano es de 70 mm y del pez piraña de 66 mm.



(a) Errores para las mediciones del phantom de pez cirujano.



(b) Errores para las mediciones del phantom de pez piraña.

Fig. 11. Errores de las mediciones de longitud e_L y posicionamiento e_D.

Finalmente, podemos ver una síntesis de la evaluación de los errores en la figura 11, donde cada punto corresponde al promedio del error evaluado por cada medición de todas las posiciones H para cada posición P. Claramente vemos una relación directa entre los comportamientos de esta figura con los de la figura 10.

De ésta podemos notar dos comportamientos: 1) El error de estimación de longitud e_L disminuye a medida que la posición es más lejana a las cámaras; y 2) El error de estimación de la posición e_D aumenta casi de forma lineal. Lo primero es de esperarse dado que estos valores fueron obtenidos de las longitudes estimadas. Lo segundo parece ser más un problema de la resolución espacial del objeto en la imagen digital.

A medida que éste se aleja de las cámaras, la exactitud de la reconstrucción tridimensional disminuye, lo que provoca errores en la recuperación de las coordenadas espaciales de los puntos de interés.

Ya que la región generada por la red neuronal es consistente, este error también lo es para ambos puntos, lo que hace que se calcule la posición del objeto considerando dicho error (que es relativo al sistema), pero que no afecte directamente a la estimación de la longitud (que es relativa al objeto).

A pesar de todo esto podemos ver que los errores en las estimaciones de longitud e_L obtenidos son relativamente pequeños; el error más grande ronda el 2.5 % de la longitud

real y el más pequeño el 0.75 %. Estos resultados son satisfactorios considerando la inexactitud de la red neuronal al generar la región que contiene al pez.

4. Conclusiones

El prototipo PEMS ha demostrado generar buenos resultados en la detección y estimación de longitud de phantoms de peces, obteniendo errores de entre el 0.75 y el 2.5 % de la longitud real, lo cual se encuentra cercano a lo obtenido por [17] y mejorando lo obtenido por [14]. Es necesario mejorar la velocidad de procesamiento para poder implementarlo en un sistema en tiempo real. Así también, continuar las pruebas para trabajar con peces en vivo en ambientes semi-controlados, lo cual implicará tomar otras consideraciones dada la forma y movimiento de los mismos.

Referencias

1. Castillo-Varguez, E. J.: Redes neuronales profundas para la detección de peces en ambientes subacuáticos (2019)
2. Espinosa-Mendoza, D. A.: Variaciones temporales de la comunidad de peces en un humedal costero de Yucatán, mediante imágenes subacuáticas y técnicas tradicionales. Master's thesis, UNAM, Instituto de Ciencias del Mar y Limnología (2019)
3. Hartley, R., Zisserman, A.: Multiple view geometry in computer vision. Cambridge University Press, 2nd edition (2003)
4. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. Lecture Notes in Computer Science, pp. 346–361 (2014) doi: 10.48550/ARXIV.1406.4729
5. Hsieh, C. L., Chang, H. Y., Chen, F. H., Liou, J. H., Chang, S. K., Lin, T. T.: A simple and effective digital imaging approach for tuna fish length measurement compatible with fishing operations. Computers and Electronics in Agriculture, vol. 75, no. 1, pp. 44–51 (2011) doi: 10.1016/j.compag.2010.09.009
6. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift (2015) doi: 10.48550/ARXIV.1502.03167
7. Jocher, G., Stoken, A., Borovec, J., Changyu, L., Laughing, Tkianai, Hogan, A., Ayush Chaurasia, Diaconu, L., Ingham, F., Colmagro, A., Ye, H., Poznanski, J., Fang, J., Kim, J., Doan, K., Yu, L.: ultralytics/yolov5: v4.0 - nn.silu() activations, weights; biases logging, pytorch hub integration (2021) doi: 10.5281/ZENODO.4418161, URL https://zenodo.org/record/4418161
8. Komeyama, K., Tanaka, T., Yamaguchi, T., Asaumi, S., Torisawa, S., Takagi, T.: Body measurement of reared red sea bream using stereo vision. Journal of Robotics and Mechatronics, vol. 30, no. 2, pp. 231–237 (2018) doi: 10.20965/jrm.2018.p0231
9. Muñoz-Benavent, P., Andreu-García, G., Valiente-González, J. M., Atienza-Vanacloig, V., Puig-Pons, V., Espinosa, V.: Enhanced fish bending model for automatic tuna sizing using computer vision. Computers and Electronics in Agriculture, vol. 150, no. 1, pp. 52–61 (2018) doi: 10.1016/j.compag.2018.04.005
10. Perez, D., Ferrero, F. J., Alvarez, I., Valledor, M., Campo, J. C.: Automatic measurement of fish size using stereo vision. In: Proceedings of the IEEE International Instrumentation and Measurement Technology Conference, pp. 1–6 (2018)
11. Ramachandran, P., Zoph, B., Le, Q. V.: Searching for activation functions (2017) doi: 10.48550/ARXIV.1710.05941

12. Redmon, J., Farhadi, A.: YOLOv3: An incremental improvement (2018) doi: 10.48550/ARXIV.1804.02767
13. Ridnik, T., Lawen, H., Noy, A., Baruch, E. B., Sharir, G., Friedman, I.: Tresnet: High performance GPU-dedicated architecture (2020) doi: 10.48550/ARXIV.2003.13630
14. Rodriguez, A., Rico-Diaz, A. J., Rabuñal, J. R., Puertas, J., Pena, L.: Fish monitoring and sizing using computer vision. In: Fish monitoring and sizing using computer vision (eds. Ferrández-Vicente, J. M., Álvarez-Sánchez, J. R., de la Paz-López, F., Toledo-Moreo, F. J., Adeli, H.), vol. 9108, pp. 419–428 (2015)
15. Sanchez-Torres, G., Ceballos-Arroyo, A., Robles-Serrano, S.: Automatic measurement of fish weight and size by processing underwater hatchery images. Engineering Letters, vol. 24, no. 4, pp. 461–472 (2018)
16. Shafait, F., Harvey, E. S., Shortis, M. R., Mian, A., Ravanbakhsh, M., Seager, J. W., Culverhouse, P. F., Cline, D. E., Edgington, D. R.: Towards automating underwater measurement of fish length: A comparison of semi-automatic and manual stereo–video measurements. ICES Journal of Marine Science, vol. 74, no. 6, pp. 1690–1701 (2017) doi: 10.1093/icesjms/fsx007
17. Shi, C., Wang, Q., He, X., Zhang, X., Li, D.: An automatic method of fish length estimation using underwater stereo system based on LabVIEW. Computers and Electronics in Agriculture, vol. 173, pp. 105419 (2020) doi: 10.1016/j.compag.2020.105419
18. Shortis, M.: Camera Calibration Techniques for Accurate Measurement Underwater, Springer International Publishing, Cham, pp. 11–27 (2019) doi: 10.1007/978-3-030-03635-5_2
19. Shortis, M. R., Ravanbakskh, M., Shaifat, F., Harvey, E. S., Mian, A., Seager, J. W., Culverhouse, P. F., Cline, D. E., Edgington, D. R.: A review of techniques for the identification and measurement of fish in underwater stereo-video image sequences (2013) doi: 10.1117/12.2020941
20. Tan, M., Pang, R., Le, Q. V.: EfficientDet: Scalable and efficient object detection (2020) doi: 10.48550/ARXIV.1911.09070
21. Tanaka, T., Ikeda, R., Yuta, Y., Tsurukawa, K., Nakamura, S., Yamaguchi, T., Komeyama, K.: Annual monitoring of growth of red sea bream by multi-stereo-image measurement. Fisheries Science, vol. 85, no. 6, pp. 1037–1043 (2019) doi: 10.1007/s12562-019-01347-7
22. Wang, C. Y., Liao, H. Y. M., Yeh, I. H., Wu, Y. H., Chen, P. Y., Hsieh, J. W.: CSPNet: A new backbone that can enhance learning capability of CNN (2019) doi: 10.48550/ARXIV.1911.11929
23. Zhang, Z.: A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 11, pp. 1330–1334 (2000) doi: 10.1109/34.888718

ISSN 1870-4069

Interpretación en tiempo real de lengua de señas mexicana con CNN y HMM

Jairo Enrique Ramírez Sánchez, Arely Anguiano Rodríguez,
Miguel González Mendoza

Tecnológico de Monterrey,
Escuela de Ingeniería y Ciencias,
México

{A01750443, A01752068}@itesm.mx, mgonza@tec.mx

Resumen. La lengua de señas mexicana (LSM) es la primordial forma de comunicación de la comunidad sorda en México. La LSM cuenta con una estructura gramatical diferente al español; además, la expresión facial juega un papel determinante a la hora de complementar el significado basado en el contexto. Esto dificulta que una persona oyente sin conocimientos previos de la lengua comprenda lo que se desea transmitir, presentando una importante barrera de comunicación a las personas sordas. Para lidiar con esto, presentamos la primera arquitectura en considerar los rasgos faciales como indicadores del tiempo gramatical para desarrollar un intérprete en tiempo real de LSM a español escrito. Nuestro modelo usa la librería de código abierto de MediaPipe para extraer marcas de la cara, posición corporal y manos. Se utilizan tres redes neuronales convolucionales 2D para codificar individualmente y extraer patrones, las redes convergen en un perceptrón multicapa para la clasificación. Finalmente, se utiliza un Modelo Oculto de Markov para predecir morfosintácticamente la secuencia de palabras más probable con forme a una base de conocimiento precargada. De los experimentos realizados, se obtuvo una precisión del 94.9 % $\sigma = 0{,}07$ para el reconocimiento de 75 palabras aisladas y 94.1 % $\sigma = 0{,}09$ para la interpretación de 20 frases en LSM en contexto médico. Al ser una aproximación basada en entradas de cámara y al observar que incluso con pocos ejemplos se puede lograr una adecuada generalización, nuestra arquitectura resultaría factible para ser escalada a otras lenguas de señas y brindar posibilidades de una comunicación eficiente a millones de personas con discapacidad auditiva.

Palabras clave: Lengua de señas mexicana, redes neuronales convolucionales, modelos ocultos de Markov, intérprete en tiempo real.

Real-Time Mexican Sign Language Interpretation Using CNN and HMM

Abstract. Mexican Sign Language (MSL) is the primary form of communication for the deaf community in Mexico. MSL has a different

grammatical structure than Spanish; furthermore, facial expression plays a determining role in complementing context-based meaning. This turns it difficult for a hearing person without prior knowledge of the language to understand what is to be transmitted, representing an important communication barrier for deaf people. In order to face this, we present the first architecture to consider facial features as indicators of grammatical tense to develop a real-time interpreter from LSM to written Spanish. Our model uses the open source MediaPipe library to extract marks from the face, body position and hands. Three 2D convolutional neural networks are used to encode individually and extract patterns, the networks converge to a multilayer perceptron for classification. Finally, a Hidden Markov Model is used to morphosyntactically predict the most probable sequence of words based on a preloaded knowledge base. From the experiments were carried out, a precision of 94.9% was obtained with $\sigma = 0.07$ for the recognition of 75 isolated words and 94.1% with $\sigma = 0.09$ for the interpretation of 20 sentences in LSM in a medical context. Being an approach based on camera inputs and observing that even with a few examples an adequate generalization can be achieved, our architecture would be feasible to be scaled to other sign languages and offer possibilities of efficient communication to millions of people with hearing disability.

Keywords: Mexican sign language, convolutional neural networks, hidden Markov models, real time interpreter.

1. Introducción

En México, existen 2.3 millones de personas con discapacidad auditiva (PDA) según datos del Instituto Nacional de Estadística y Geografía para la Información y el Consejo Nacional para la Prevención de la Discriminación 2020. La lengua oficial de la comunidad sorda en México es la Lengua de Señas Mexicana (LSM), destacando que cada país posee un sistema de signos nacional designado para su territorio.

Para lograr una comunicación efectiva y eficiente entre una PDA y una Persona Oyente (PO), no es suficiente con que se aprenda a identificar los movimientos de las manos para designar a cada una de las palabras, ya que los rasgos manuales (RM) representan tan sólo el $20\,\%$ de la comunicación total, mientras que los no manuales (RNM) figuran como el resto [**?**]. Por ello, también es necesario tener conocimiento de su gramática y de los rasgos no manuales, en concreto, los faciales, pues estos son los que indican el tiempo verbal en que la acción se llevó, lleva o llevará a cabo.

El tiempo pasado se expresa a través de un semblante de que la acción ha sido consumada, en este se puede observar una expresión relajada, donde el labio inferior sobresale del superior asintiendo suavemente con la cabeza; por otra parte, en el tiempo presente se proyecta una expresión neutra, alzando ligeramente las cejas y acorde con el mensaje; por último, el tiempo futuro se manifiesta a través de una expresión de duda o pensamiento, donde el cuello se rota ligeramente y los ojos se dirigen hacia cualquiera de las dos esquinas superiores del plano visual. Acerca de la gramática LSM, los verbos ser y estar se ven omitidos, por lo que se expresa únicamente el sustantivo y adjetivo o, en su defecto, sustantivo y lugar.

Igualmente, es relevante que el movimiento de cada seña se realiza con los dedos en posición de la letra con la cual inicia el vocablo asignado a la seña, o bien, destacando características sensoriales de la palabra que se trata. Como se puede percibir, para una persona no inmersa en un ambiente y sin un contacto previo a la comunidad sorda, es de suma dificultad alcanzar un entendimiento total de lo que la PDA busque comunicar.

Esto, aunando al hecho de que en México solo existan 42 intérpretes certificados en LSM [**?**], hace que la comunicación para las PDA esté prácticamente cercada. Existe una vasta investigación en lengua de señas americana (ASL, por sus siglas en inglés), lengua de señas británica (BSL, por sus siglas en inglés), lengua de señas china (CSL, por sus siglas en inglés), entre otras; sin embargo, la incursión en el estudio de LSM es más que limitada.

En nuestra investigación, abordaremos el análisis y reconocimiento de señas en tiempo real haciendo uso de redes neuronales de arquitectura convolucional para la clasificación de señas tomando en cuenta tres aspectos: la expresión facial, movimientos de las manos y posición corporal; la interpretación es mejorada por un Modelo Oculto de Markov para el enriquecimiento con contexto y gramática. El presente artículo está organizado de la siguiente manera: La sección dos aborda la revisión del trabajo previo sobre el reconocimiento en tiempo real de lenguas de señas.

La sección tres contiene la descripción de nuestro modelo propuesto, así como las fases del procesamiento. La sección cuatro explica la forma en cómo se realizó la adquisición de los datos, el número de participantes y las condiciones de prueba. En la sección cinco se presentan los experimentos propuestos y sus respectivos resultados. Finalmente, en la sección seis se abordan las conclusiones y el trabajo futuro.

2. Trabajos relacionados

2.1. Métodos de captación

Los estudios sobre la lengua de señas pueden ser divididos en tres aproximaciones: basadas en entradas de cámara [**?**], sensores externos [**?**,**?**] y en entradas por medio de dispositivos como guantes especializados [**?**,**?**]. En orden, el primer tipo de aproximación ofrece la ventaja de ser de fácil y barata implementación a costa de requerir grandes cantidades de datos para generalizar los distintos tamaños de manos y colores de piel.

En segundo lugar, los basados en sensores externos (comúnmente, Microsoft Kinect sensor) presentan una mejora en la generalización, sin embargo, aumentan los costos y la complejidad de implementación. Finalmente, los guantes suelen ser la cúspide de la generalización llegando también a ser la opción de más difícil implementación, además de ser un considerado como un método intrusivo.

2.2. Técnicas utilizadas

Para lograr la clasificación del movimiento en las lenguas de señas han sido propuestas diversas aproximaciones, algunas de ellas basadas en Modelos Ocultos de Markov (HMM) como [**?**], quienes utilizan estos modelos estadísticos para la predicción del gesto manual más probable de la lengua de señas americana.

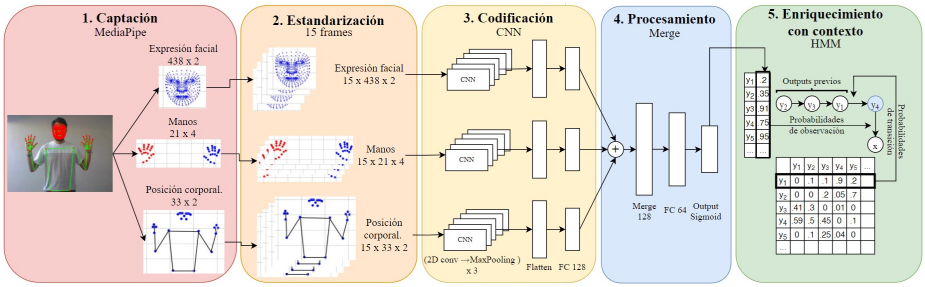


Fig. 1. Etapas de la detección.

Los algoritmos de Data Time Wraping (DTW) presentan una mejora de clasificación ya que, al realizar la comparación contra todos los movimientos almacenados, logran generalizar señas de manera independiente a la duración temporal, en [**?**] fue alcanzada una precisión de 98.57 % en para 20 palabras de la lengua de señas mexicana, sin embargo, por la naturaleza de DTW hace que al aumentar el dataset, el tiempo de cómputo crezca de forma polinómica.

Por último, las Redes Neuronales Convolucionales (CNN) han demostrado obtener un correcto funcionamiento en la clasificación en tiempo real de acciones humanas en general [**?**]. La ventaja que presentan las CNN con respecto a otros métodos es la amplia capacidad de generalización, las capas de convolución extraen las características más importantes identificando patrones en las imágenes y las capas Fully Conected realizan la clasificación.

En [**?**] es utilizado el modelo VGG Network para clasificar 26 letras del alfabeto dactilológico americano alcanzando una precisión de 98.56 %. En [**?**] es propuesta una arquitectura que combina una imagen de la persona que realiza la seña con el extracto de la posición corporal y de las manos realizada por un MS Kinect, el cual consigue una precisión 94.2 % para 25 palabras por separado en lengua de señas americana (caso ASL).

Adicionalmente, en [**?**] se presenta una solución para la lengua de señas china (caso CSL) una arquitectura que combina una codificación convolucional con un procesamiento por medio de una red neuronal recurrente con módulo de atención, la cual fue entrenada con más de 25 mil videos creados por 50 intérpretes, mostrando una precisión de 82.7 %.

2.3. Soluciones para LSM en México

En México existe poco trabajo relacionado a la lengua de señas nacional, enfocándose principalmente en el reconocimiento estático de letras del abecedario dactilológico [**?**,**?**,**?**,**?**,**?**,**?**]. En [**?**] se aborda la identificación de 15 palabras aisladas haciendo uso de un MS kinect para identificar puntos corporales; el algoritmo AdaBoost realizaba la clasificación temporal de cada una de ellas. Por su parte, [**?**] presenta una aproximación al reconocimiento de 20 palabras con el algoritmo DTW obteniendo una precisión media de 98.57 %, sin embargo, se torna inviable para escalar el dataset debido al tiempo de computación.

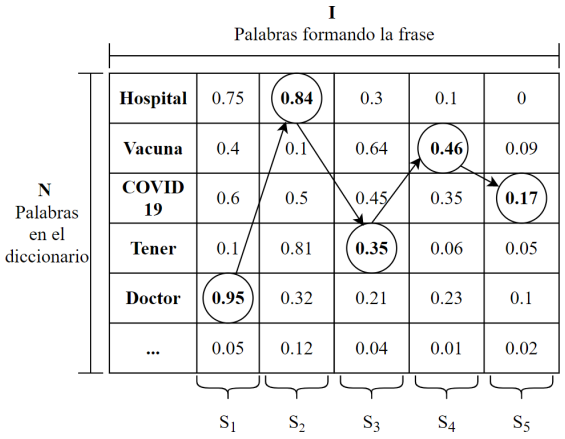


Fig. 2. Ejemplificación del algoritmo de Viterbi aplicado a la determinación de la secuencia de palabras más probable conforme a la gramática de LSM, en este caso: Doctor hospital tener vacuna COVID-19.

Finalmente, sólo [**?**] aborda la interpretación de 5 frases en tiempo real por medio del procesamiento de la posición manual y corporal utilizando momentos geométricos, dichas salidas son integradas por medio de un HMM alcanzando una sensibilidad del 86 %.

En nuestro trabajo, se propone la primera arquitectura que utiliza la expresión facial como determinante del tiempo verbal basado en entradas de cámara con la capacidad de interpretar morfosintácticamente 75 palabras en tiempo real, cuya combinación permite generar más de 50 frases, de las cuales, el rendimiento de 20 frases en el contexto médico es analizado en este estudio.

Para lograr esto, se generó un dataset de 49 palabras (cada uno de los 13 verbos incluidos era conjugado en los tiempos gramaticales dando lugar a tres clases por cada uno) con un promedio de 35 ejemplos para cada una, señadas por seis voluntarios. Adicionalmente, se obtuvo una base de conocimiento con 100 de las frases comunes LSM, la cual aporta los patrones sobre la estructura gramatical propia.

3. Propuesta

El modelo de procesamiento presentado se divide en 5 etapas como se muestra en la figura **??**. La captación utiliza la librería de código abierto MediaPipe presentada en [**?**]. Dicha librería permite construir flujos de percepción con herramientas de visualización de OpenCV [**?**]. Ambas permiten la detección de marcas en las manos, cuerpo y cara en tiempo real con una amplia capacidad de generalización sin importar el color de piel, tamaño de las manos ni altura de la persona.

Así, el sistema de captación mezcla los beneficios de las entradas por cámara (factibilidad económica) y los sensores externos (generalización) en una propuesta no intrusiva. Para la estandarización se extraen las coordenadas de las marcas para ser colocadas en tres matrices, seleccionando 15 frames en cada segundo.

Tabla 1. Señas utilizadas.

Categoría	Seña	Manos	Ejemplos	Categoría	Seña	Manos	Ejemplos
Básicas	Hola	Una	30	Verbo	Hacer	Ambas	40
	Gracias	Ambas	30		Comer	Una	40
	Día	Ambas	40		Pensar	Una	30
	Horario	Una	40		Creer	Ambas	30
Persona	Niño	Una	35		Sentir	Una	30
	Hombre	Una	35		Ir	Ambas	40
	Mujer	Una	35		Contagiar	Ambas	40
	Intérprete	Ambas	40		Pagar	Ambas	40
	Adulto mayor	Ambas	40		Trabajar	Ambas	40
Pregunta	Qué	Una	30		Gustar	Una	40
	Quién	Ambas	40		Poder	Una	40
	Dónde	Ambas	30		Tener	Una	40
	Cómo estás	Ambas	30		Comprar	Ambas	35
Emergencia	Accidente	Una	35	Estado	Feliz	Una	30
	Alergia	Una	40		Bien	Una	30
	Fiebre	Una	35		Mal	Una	30
	Vacuna	Ambas	35		Enfermo	Ambas	40
	Doctor	Ambas	40		Triste	Una	30
	Emergencia	Ambas	40		Enojado	Ambas	30
	Infección	Una	35		Nervioso	Ambas	40
	Medicina	Una	35	Lugares	Escuela	Una	30
	Operación	Ambas	35		Casa	Ambas	30
	COVID-19	Ambas	40		Hospital	Ambas	35
	-	-	-		Calle	Ambas	30

Las coordenadas de la expresión facial con dimensión de 438×2, las manos de 21×4 y la posición corporal 33×2. Posteriormente, las matrices se codifican de manera independiente con una red neuronal con tres capas de convolución 2D cada una seguida por MaxPooling. Esto permite identificar patrones generales, propiciando que señas parecidas, sean codificadas de forma similar.

Las codificaciones se integran por medio de una capa de adición. La codificación de las manos aporta significado, la posición corporal la ubicación espacial y la cara el tiempo verbal. En esta última parte de la arquitectura se utiliza un perceptrón multicapa para realizar la clasificación.

La última capa de la red utiliza un función sigmoide como activación, misma que asigna un vector probabilidades de observación $\vec{O}_i$ cuya entrada O_i^j se refiere a la palabra j en el diccionario de longitud N para cada paso temporal i. Finalmente, se utiliza un HMM para realizar la clasificación con base en el contexto previo. Es empleada una matriz de transición T entre palabras del dataset calculada con 100 frases comunes en LSM extraídas de [**?**].

Tabla 2. Descripción de los participantes en la generación del DataSet.

Persona	Sexo	Edad	Conocimiento de LSM	Captación
1	Hombre	57	Básico	Manual y facial
2	Mujer	35	Intermedio	Manual y facial
3	Hombre	39	Básico	Manual y facial
4	Hombre	19	Básico	Manual y facial
5	Hombre	20	Básico	Manual
6	Mujer	18	Intermedio	Manual

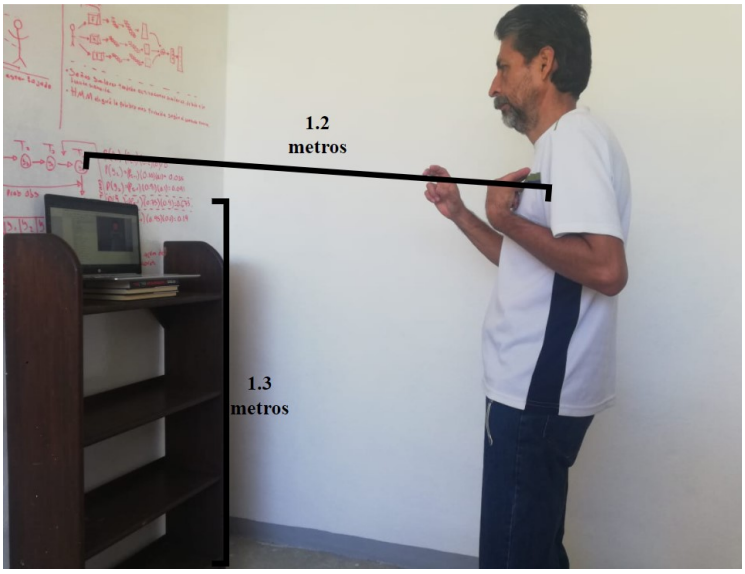


Fig. 3. Ejemplo del montaje para la colección de los datos.

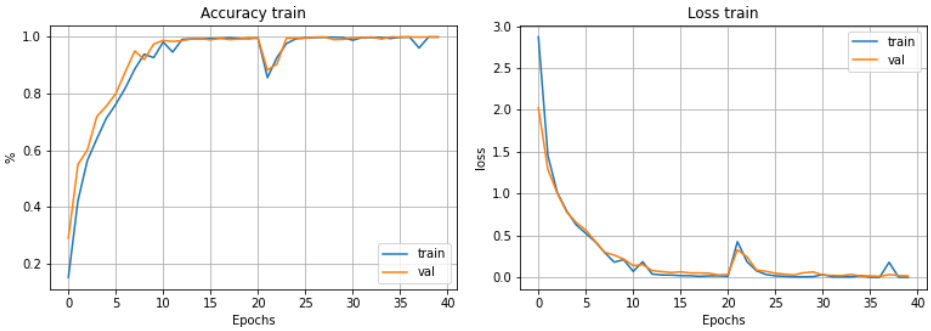
Con $\vec{O_i}$ y T se realiza la selección de la palabra más probable con un sentido morfosintáctico, el conjunto de estados posibles para cada paso temporal se representa por $S = (S_1, S_2, ..., S_I)$, siendo el valor S_1 la palabra más probable. La probabilidad se calcula de forma recurrente como se muestra en la siguiente ecuación:

$$p(O^j, S^j) = \prod_{i=1}^{I} p(O_i^j|S_i^j)p(S_i^j|S_{i-1}^j) \;\; \forall j \in 1, 2, ..., N, \tag{1}$$

donde:

I = total de pasos temporales,
$p(O_i^j|S_i^j)$ = probabilidad de observación,
$p(S_i^j|S_{i-1}^j)$ = probabilidad de transición.

Por último, se ejecuta una implementación manual del algoritmo de Viterbi [**?**] para rastrear el camino de mayor probabilidad. El proceso se ejemplifica en la figura **??**. Dicha implementación reduce el error de interpretación en un 11.7 % en contraste con la identificación de palabras aisladas.



(a) (b)

Fig. 4. Curvas de aprendizaje para set de entrenamiento y validación utilizando dropout (0.3) como regularización. El entrenamiento se consideró suficiente debido a la tendencia asintótica gradual para la reducción del costo.

4. DataSet

4.1. Descripción

El dataset utilizado en el presente trabajo fue generado con ayuda de seis personas utilizando las librerías de MediaPipe y OpenCV. Ver sección **??** para la descripción de los participantes. La captación para las coordenadas de las manos consta de 15 frames que contienen el movimiento de una matriz de 21 puntos para cada mano (15, 21, 4), 33 para el cuerpo (15, 33, 2) y 438 puntos para la cara (15, 438, 2). Se realizó un promedio de 35 ejemplos para una de las 49 señas (13 verbos y 36 palabras) y 15 para cada expresión facial de los tres tiempos verbales utilizados (pasado, presente y futuro). En total, la clasificación constaba de $13 \times 3 + 36 = 75$ clases.

4.2. Participantes

Participaron seis voluntarios con conocimiento de LSM con edades entre 18 y 57 años. Todos los participantes fueron informados del propósito de la investigación y otorgaron su consentimiento. El sexo, nivel de conocimiento, edad y tipo captación se presenta en la tabla **??**.

4.3. Adquisición de datos

Para la obtención de los datos se desarrolló un código en el lenguaje de programación de Python para realizar la identificación de puntos en tiempo real. Cuando el usuario estaba listo, se corría la función y al terminar la captación los resultados eran almacenados. Los participantes se colocaban a 1.2 metros de distancia de la computadora con el montaje que se muestra en la figura **??**.

Tabla 3. Resultados obtenidos para el set de prueba.

Set	F1 Score
Entrenamiento	0.981
Prueba	0.978

Tabla 4. Resultados experimento 1.

Categoría	Precisión media	σ
Básicas	0.987	0.03
Persona	0.943	0.1
Lugar	0.937	0.05
Estado	0.99	0.01
Verbo presente	0.91	0.09
Verbo pasado	0.94	0.06
Verbo futuro	0.89	0.11
Pregunta	0.962	0.07
Emergencia	0.981	0.04

4.4. Estandarización del dataset

Debido al significado espacial particular, cada palabra cuenta con diferente duración temporal en ser señada. Para uniformar el dataset fueron seleccionados n = 15 frames distribuidos a lo largo de los k frames de longitud para cada muestra. Keval H. et al. muestra en [**?**] que el mínimo número de frames por segundo para identificar una acción humana es de 8, así utilizar casi el doble de frames asegura un nivel de detalle aceptable.

5. Experimentos y resultados

5.1. Entrenamiento

Durante el entrenamiento del modelo propuesto se realizó una partición del dataset en 75 % entrenamiento (a su vez dividido en 10 % para validación) y 25 % prueba. El modelo se entrenó por 40 épocas en GPU sustentado por el software de cómputo en la nube de Google Colaboratory [**?**]. La curva de aprendizaje obtenida es mostrada en la figura **??**. Para la medición del rendimiento del modelo se utilizó el F1 Score mostrado en la ecuación **??**. Los resultados se observa en la tabla **??**:

$$\text{Precision} = \frac{T_p}{T_p + F_p}, \tag{2}$$

$$\text{Recall} = \frac{T_p}{T_p + F_n}, \tag{3}$$

$$\text{F1Score} = 2\,\frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}. \tag{4}$$

Tabla 5. Resultados experimento 2.

Frase español	Fase LSM	Precisión media	σ
El niño está en el hospital porque está enfermo	Niño hospital, enfermo	0.95	0.1
El adulto mayor se sintió mal y fue al hospital	Adulto mayor sentir (pasado) mal, hospital ir (pasado)	0.975	0.05
El adulto mayor se sintió bien con la vacuna	Adulto mayor sentir (pasado) bien vacuna	0.975	0.03
El niño se sintió bien y fue a la escuela	Niño sentir (pasado) bien, escuela ir (pasado)	0.925	0.15
El adulto mayor irá al hospital por la vacuna para el COVID-19	Adulto mayor ir (futuro) hospital COVID-19 vacuna	0.962	0.03
El niño enfermo puede contagiar al doctor	Niño enfermo poder (presente) contagiar (presente) doctor	0.962	0.1
El hombre contagió a la mujer en la casa	Hombre contagiar (pasado) mujer casa	0.937	0.12
El doctor tiene medicina para la operación	Doctor tener (presente) operación medicina	0.987	0.02
El niño estará feliz de ir a la escuela	Niño feliz, escuela ir (futuro)	0.977	0.02
El hombre tiene una infección	Hombre tener (presente) infección	0.8	0.14
Yo pagaré la medicina para el COVID-19	Yo pagar (futuro) COVID-19 medicina	0.83	0.11
El hombre fue a su casa porque se sintió mal	Hombre ir (pasado) casa, sentir (pasado) mal	1.0	0.0
El hombre está nervioso porque la mujer tiene COVID-19	Nervioso hombre, mujer tener (presente) COVID-19	0.825	0.12
La intérprete tuvo una emergencia y fue al doctor	Intérprete tener (pasado) emergencia, ir (pasado) doctor	0.95	0.1
El niño se fue a su casa porque se sintió mal en la escuela	Niño ir (pasado) casa, escuela niño sentir (pasado) mal	0.96	0.04
El hospital tendrá vacunas COVID-19	Hospital tener (futuro) COVID-19 vacuna	0.95	0.1
El doctor está feliz porque la mujer se podrá ir a su casa	Doctor feliz mujer poder (futuro) ir casa	1.0	0.0
¿Qué medicina tiene que comprar el hombre?	¿Medicina hombre tener comprar (presente) qué?	0.95	0.1
El niño tiene fiebre por la infección	Niño tener (presente) infección fiebre	0.862	0.11
El doctor irá a su casa porque se siente enfermo	Doctor casa ir (futuro) sentir (presente) enfermo	0.987	0.025

5.2. Resultados experimento 1: Enfoque en palabras aisladas

Cada palabra mostrada en la tabla **??** se señó 10 veces por cada uno de los seis voluntarios, midiendo el rendimiento de manera binaria, es decir, si era predicha correctamente obtenía una calificación de 100 %, caso contrario 0 %.

Para el caso de los verbos, se modificó la forma de medir el rendimiento en aras de obtener una calificación ponderada con base en la complejidad que representaba; así, asignamos un 100 % si la seña y el tiempo gramatical predicho coincidían con el esperado; 70 % si coincidía la palabra, pero no el tiempo. 30 % si el tiempo, pero no la palabra. 0 % si la predicción era totalmente diferente. Se obtuvo una precisión media para las nueve categorías de **94.9 %** con $\sigma = 0{,}07$.

5.3. Resultados experimento 2: Enfoque en frases

Los participantes realizaron la seña y expresión facial de 20 frases tomando cinco muestras. Se utilizó la precisión como métrica de desempeño. Los resultados se muestran en la tabla **??**. Se obtuvo una precisión media para las 20 frases de 94.1 % con $\sigma = 0{,}09$. En contraste, la interpretación con palabras aisladas, es decir, sin enriquecimiento con contexto, alcanzó un **82.4 %** con $\sigma = 0{,}12$.

6. Conclusiones

Como ha sido discutido a lo largo de este trabajo, la interpretación en tiempo real de las lenguas de señas es un proceso complejo en el que intervienen diversos factores como la posición corporal, la expresión facial, los movimientos de las manos y el contexto específico en el que se aborda lo anterior.

Nuestro estudio sienta las bases de un primer acercamiento a la creación de un intérprete completo de lenguas de señas a idiomas hablados, al ser el único trabajo en español hasta el momento que considera la expresión facial como indicador del tiempo gramatical.

En suma, en cuestiones de factibilidad, nuestro modelo demostró que aun con relativamente pocos ejemplos (en promedio, 35 por cada seña y una base de conocimiento de 100 frases) es posible obtener una precisión del **94.9 %** para reconocimiento de 75 palabras aisladas y **94.1 %** para 20 frases de prueba en el contexto médico; mismas que validan tanto la amplia capacidad de generalización de la arquitectura codificadora CNN como el HMM identificador de contexto.

Esto posiciona a nuestro trabajo como una opción económicamente viable - debido a que sólo utiliza una computadora -, de fácil implementación y totalmente escalable a otras lenguas de señas, especialmente las correspondientes a países de bajo o nulo estudio en el campo, mejorando la inclusión de millones de personas con discapacidad auditiva.

Como trabajo futuro, se pretende diseñar, implementar e incluir una máquina de traducción estadística que permita pasar de la estructura gramatical de LSM a español, la cual será una aportación significativa con miras a seguir potenciando las herramientas de comunicación efectiva.

Referencias

1. Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., Baskurt, A.: Sequential deep learning for human action recognition. In: Lecture Notes in Computer Science, vol. 7065, pp. 29–39 (2011) doi: 10.1007/978-3-642-25446-8_4
2. Ben-Jmaa, A., Mahdi, W., Ben-Jemaa, Y., Ben-Hamadou, A.: A new approach for hand gestures recognition based on depth map captured by RGB-D camera. Computación y Sistemas, vol. 20, no. 4, pp. 709–721 (2016) doi: 10.13053/cys-20-4-2390
3. Bisong, E.: Google colaboratory. Building machine learning and deep learning models on google cloud platform: A comprehensive guide for beginners, pp. 59–64 (2019)
4. Carmona-Arroyo, G., Rios-Figueroa, H. V., Avendaño-Garrido, M. L.: Mexican sign-language static-alphabet recognition using 3D affine invariants. In: Machine Vision Inspection Systems, vol. 2 (2021)
5. Cruz, M.: Gramática de la lengua de señas mexicana. Centro de Estudios Lingüísticos y Literarios, Colegio de México (2008)
6. Dong, C., Leu, M. C., Yin, Z.: American sign language alphabet recognition using Microsoft Kinect. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 44–52 (2015) doi: 10.1109/cvprw.2015.7301347
7. Fels, S. S., Hinton, G. E.: Glove-TalkII-a neural-network interface which maps gestures to parallel formant speech synthesizer controls. IEEE Transactions on Neural Networks, vol. 8, no. 5, pp. 977–984 (1998) doi: 10.1109/72.655042
8. Forney, G. D.: The viterbi algorithm. In: Proceedings of the IEEE, vol. 61, pp. 268–278 (1973) doi: 10.1109/proc.1973.9030
9. Galicia, R., Carranza, O., Jimenez, E. D., Rivera, G. E.: Mexican sign language recognition using movement sensor. In: Proceedings of the IEEE 24th International Symposium on Industrial Electronics, vol. 2015, pp. 573–578 (2015) doi: 10.1109/isie.2015.7281531

10. García-Bautista, G., Trujillo-Romero, F., Caballero-Morales, S. O.: Mexican sign language recognition using kinect and data time warping algorithm. In: Proceedings of the International Conference on Electronics, Communications and Computers, pp. 1–5 (2017) doi: 10.1109/conielecomp.2017.7891832
11. Grobel, K., Assan, M.: Isolated sign language recognition using hidden markov models. In: Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, IEEE, vol. 1, pp. 162–167 (1997) doi: 10.1109/icsmc.1997.625742
12. Huang, J., Zhou, W., Li, H., Li, W.: Sign language recognition using 3D convolutional neural networks. In: Proceedings of the IEEE International Conference on Multimedia and Expo, IEEE Computer Society, vol. 2015, pp. 1–6 (2015) doi: 10.1109/icme.2015.7177428
13. Huang, J., Zhou, W., Zhang, Q., Li, H., Li, W.: Video-based sign language recognition without temporal segmentation. In: Proceedings of the 32nd AAAI Conference on Artificial Intelligence, pp. 2257–2264 (2018) doi: 10.48550/ARXIV.1801.10111
14. Kadhim, R. A., Khamees, M.: A real-time american sign language recognition system using convolutional neural network for real datasets. TEM Journal, vol. 9, no. 3, pp. 937–943 (2020) doi: 10.18421/tem93-14
15. Keval, H., Sasse, M. A.: To catch a thief - you need at least 8 frames per second: The impact of frame rates on user performance in a CCTV detection task. In: Proceedings of the 2008 ACM International Conference on Multimedia, with co-located Symposium and Workshops, pp. 941–944 (2008) doi: 10.1145/1459359.1459527
16. Lugaresi, C., Tang, J., Nash, H., McClanahan, C., Uboweja, E., Hays, M., Zhang, F., Chang, C.-L., Yong, M. G., Lee, J., Chang, W.-T., Hua, W., Georg, M., Grundmann, M.: Mediapipe: A framework for building perception pipelines (2019) doi: 10.48550/ARXIV.1906.08172
17. Luis-Pérez, F. E., Trujillo-Romero, F., Martínez-Velazco, W.: Control of a service robot using the mexican sign language. Lecture Notes in Computer Science, Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 7095, pp. 419–430 (2011) doi: 10.1007/978-3-642-25330-0_37
18. Naveenkumar, M., Ayyasamy, V.: OpenCV for computer vision applications. In: Proceedings of the National Conference on Big Data and Cloud Computing, pp. 52–56 (2016)
19. Ordóñez, E.: Asociación de intérpretes en lengua de señas del Distrito Federal: Número de intérpretes de lengua de señas en méxico (2015)
20. Priego-Pérez, F. P.: Reconocimiento de imágenes del lenguaje de señas mexicano. Master's thesis, Centro de Investigación en Computación, Instituto Politécnico Nacional (2012)
21. Rashed, J., Al-Behadili, H.: New method for hand gesture recognition using wavelet neural network (2017)
22. Serafín, M., González, R.: Diccionario de lenguaje mexicano de señas. Revista de Investigación, vol. 38, no. 83, pp. 240 (2011)
23. Solís, F., Martínez, D., Espinoza, O.: Automatic mexican sign language recognition using normalized moments and artificial neural networks. Engineering, vol. 8, no. 10, pp. 733–740 (2016) doi: 10.4236/eng.2016.810066
24. Solís, F., Toxqui, C., Martínez, D.: Mexican sign language recognition using Jacobi-Fourier moments. Engineering, vol. 7, no. 10, pp. 700–705 (2015) doi: 10.4236/eng.2015.710061
25. Sosa-Jimenez, C. O., Rios-Figueroa, H. V., Rechy-Ramirez, E. J., Marin-Hernandez, A., Gonzalez-Cosio, A. L. S.: Real-time mexican sign language recognition. In: Proceedings of the IEEE International Autumn Meeting on Power, Electronics and Computing, pp. 1–6 (2017) doi: 10.1109/ropec.2017.8261606
26. Tolba, A. S., Abu-Rezq, A. N.: Arabic glove-talk (AGT): A communication aid for vocally impaired. Pattern Analysis and Applications, vol. 1, no. 4, pp. 218–230 (1998) doi: 10.1007/bf01234769
27. Álvarez Torres, N.: Kinect V2 como alternativa para desarrollar un traductor de ideogramas de lengua de señas mexicana (2016)

ISSN 1870-4069

Caracterización de emociones y personalidad en el rostro para agentes conversacionales personificados

Eduardo David Martínez-Hernández, María Lucila Morales-Rodríguez, Nelson Rangel-Valdez, Laura Cruz-Reyes, Claudia Gómez-Santillán

TecNM/Instituto Tecnológico de Cd. Madero, México

{g19073015, lucila.mr}@cdmadero.tecnm.mx, {nelson.rangel, lauracruzreyes, claudia.gomez}@itcm.edu.mx

Resumen. En la creación de Agentes Conversacionales Personificados (ACP) algunas de las áreas de estudio que se abordan son el modelado de la personalidad y emociones, debido a que estas áreas tienen una gran influencia en la comunicación no verbal y en el comportamiento que se presenta a través de expresiones faciales. En este artículo se propone asociar los modelos de personalidad y las emociones mediante la identificación de actitudes, las cuales suelen manifestar emociones y se presentan de acuerdo con los niveles de puntaje que del perfil de personalidad. El objetivo principal de identificar actitudes es poderlas aplicar en la caracterización de expresiones faciales de un ACP, debido a que son un factor que contribuyen a la credibilidad, sociabilidad y realismo en la creación de agentes.

Palabras clave: Agente conversacional personificado, expresión facial, emociones, personalidad.

Characterization of Emotions and Personality in the Face for Embodied Conversational Agents

Abstract. In the creation of Embodied Conversational Agents (ECAs), some of the areas of study that are addressed are the modeling of personality and emotions. These areas have a great influence on non-verbal communication and on the behavior that is presented through facial expressions. In this paper, it is proposed the association of emotions and personality models by identifying attitudes, which usually manifest emotions and are presented according to the score levels of the personality profile. The main objective of identifying attitudes is to be able to apply them in the characterization of facial expressions of an ECA because they are a factor that contributes to credibility, sociability, and realism in the creation of agents.

Keywords: Embodied conversational agents, facial expression, emotions, personality.

1. Introducción

El rostro es uno de los medios de comunicación no verbal más importantes, a través de este se pueden expresar sentimientos o emociones, estados de ánimo más duraderos, así como características y rasgos de la personalidad. Dado esto, puede darse cierta complejidad al generarse las expresiones faciales, ya que se presentan en un periodo corto de tiempo la actuación de diversas zonas faciales, las cuales permiten la aparición de una gran variedad de expresiones faciales [1].

La expresión facial es uno de los elementos más importantes que puede llegar a tener un Agente Conversacional Personificado (ACP), a través de ella el usuario final puede percibir las emociones, intenciones y personalidad del agente. Para recrear las expresiones faciales humanas por computadora, se debe emular la forma en la que los humanos se comunican entre ellos. Un ACP puede emular este tipo de comportamiento, ya que estos agentes tienen la capacidad de demostrar cierto comportamiento social, para ello, recurren a su representación visual para reforzar la creencia que el agente es una entidad social [2].

Para modelar las expresiones faciales en un agente es necesario estudiar las emociones, debido a que estas comúnmente se presentan en el rostro y ayudan a comprender su comportamiento. La importancia de estudiar los modelos de personalidad radica en que estos ayudan a dar una mejor interpretación del comportamiento en el agente, así como también se puede llegar a influir en la intensidad de las emociones de acuerdo con el tipo de personalidad del agente.

Para la realización de un modelo de selección de expresiones faciales de un ACP, es necesario trabajar las áreas de personalidad y emociones, debido a que estas influyen en el proceso de comunicación que se da por medio de las expresiones faciales, así mismo, son un factor que ayudan a determinar el nivel de credibilidad, sociabilidad y el realismo en el agente.

2. Trabajos relacionados

Algunos de los trabajos relacionados con la asociación del modelo de personalidad FFM y los estados emocionales es el propuesto por [3], donde se hace un estudio multicultural de la relación que existe en la comparación de las dimensiones del modelo de personalidad FFM con los estados afectivos. La diferencia con este proyecto es la asociación de actitudes con las emociones básicas y en cambio en el otro hacen un enfoque hacia la relación multicultural de los estados afectivos obtenidos de un modelo circunflejo afectivo.

Entre los proyectos de agentes que utilizan el modelo de personalidad FFM uno de ellos es el desarrollado por [4] que modela la influencia de la personalidad en las preferencias de un agente virtual en el contexto de la toma de decisiones multicriterio, cabe mencionar que las emociones no son consideradas en este proyecto. También se tiene el propuesto por [5], en el cual se diseña un agente con el rol de un terapeuta virtual, este cuenta con un modelo comportamental y kinésico donde la personalidad y emociones influyen en ambos modelos para actualizar el estado emocional del agente y su expresividad.

Otro proyecto relacionado con el modelo de personalidad FFM es el propuesto por [6] que utiliza el lenguaje AIML para seleccionar frases acordes al estado emocional del agente, sin embargo, no se tiene un enfoque hacia las expresiones faciales.

Así mismo, también se tiene el proyecto propuesto por [7] el cual es un modelo de selección y caracterización de expresiones no verbales para un agente virtual inteligente. La diferencia con este proyecto es que el propuesto aquí toma en consideración la personalidad y el sistema FACS para la caracterización de las expresiones.

3. Agentes conversacionales personificados (ACP)

El objetivo principal que se tiene al realizar un ACP, es la creación de agentes inteligentes que tengan las capacidades de demostrar un comportamiento social y puedan recurrir a su representación visual con el objetivo de reforzar la creencia de que son una entidad social [2]. La creación de un ACP completo es un trabajo complejo, el cual requiere el conocimiento en diferentes áreas de investigación que van desde arquitecturas de agentes, discurso sintético, procesamiento de lenguaje natural, emociones, gráficos y diseño de interfaz.

Según [2] en la implementación de un ACP se espera que se cubran como mínimo aspectos como la personificación del agente, así como un sistema de reconocimiento de entrada de datos, un motor de comportamiento de algún tipo, un modelo de personalidad y posiblemente de emoción.

La credibilidad es uno de los aspectos más importantes que se toma en cuenta para la creación de un ACP, ya que se enfoca en el problema de crear la ilusión de vida para quienes observan y se relacionan con el agente. Según [2] para tener éxito en este punto, el usuario final es el único que puede definir la calidad del agente mediante su nivel de realismo y atracción. Para darle un comportamiento creíble y sociable al agente, es necesario estudiar los modelos de personalidad y emociones, ya que estos elementos forman parte la definición de lo que es un agente creíble, los cuáles serán plasmados en la representación visual del agente.

4. Modelos de personalidad

La importancia de trabajar con un modelo de personalidad en un agente se debe a la influencia que puede tener la personalidad al manifestarse en aspectos como el habla, expresiones faciales, gestos, posturas, así como en la forma de pensar, actuar y en la toma de decisiones. Existen dos modelos de personalidad que son usados frecuentemente para modelar la personalidad en agentes, el primero de ellos es el modelo MBTI (Myers-Briggs Type Indicator) [8] que está basado en tipos de personalidad y hace un enfoque en la comprensión del comportamiento humano, así como en las motivaciones, habilidades y áreas de desarrollo personal del ser humano.

Este modelo mide cuatro rasgos de la personalidad internamente consistentes y relativamente no correlacionados, los cuales son Extraversión-Introversión (EI), Sensorial-Intuición (SN), Pensar-Sentir (TF) y Juzgar-Percibir (JP). El modelo MBTI

describe las preferencias de cuatro dicotomías, donde cada persona tiene una preferencia natural para un polo opuesto de cada dicotomía.

El segundo modelo de personalidad es el FFM (Five Factor Model) [9] el cual está basado en rasgos de la personalidad y está compuesto por cinco dimensiones (Amabilidad, Responsabilidad, Extraversión, Neuroticismo y Apertura a la experiencia), donde cada una de estas dimensiones contiene seis facetas que definen los rasgos de la personalidad, las cuales contienen cierto nivel de puntaje que van desde puntuaciones altas a bajas.

El modelo de personalidad FFM es uno de los más usados en el campo de evaluación de los ACP, debido a que es considerado como la solución más aceptada para el problema de descripción de estructura de rasgos [10], ya que estos describen las diferencias individuales por medio de adjetivos, los cuales representan actitudes o conductas que están ligadas a los niveles de puntaje, donde un puntaje alto o bajo describe una actitud opuesta a su polo, mientras que en un puntaje medio existe una combinación moderada de ambas actitudes.

5. Emociones en el rostro

Las emociones son fenómenos de corta duración, relacionados con los sentimientos, la estimulación, la intención y la expresión, donde este último se refiere al aspecto comunicativo de la emoción, los cuales se dan por medio de posturas, gesticulaciones, vocalizaciones y expresiones faciales [11]. Las emociones muy a menudo se presentan en el rostro y son un factor esencial en la comunicación, ya que facilitan el flujo de información que reciben los interlocutores. Al expresar emociones, entre la información que se puede comunicar de manera no verbal, se encuentran los estados de ánimo y la manera en que se interpretan las situaciones.

A pesar de que existe una diversidad de emociones donde algunas de ellas pueden ser más complejas que otras, la mayoría de los investigadores limitan sus estudios en siete emociones básicas [12], las cuales son alegría, miedo, tristeza, ira, sorpresa, desagrado y desprecio. Cabe destacar que de las siete emociones solo existe una emoción positiva (alegría), así como existen cinco emociones negativas (ira, tristeza, miedo, asco, desagrado y desprecio) y una emoción neutral (sorpresa).

Existen estudios [13, 14] que se basan en la expresión facial como un medio de estudio hacia la personalidad y emociones, uno de ellos es el Sistema de Codificación Facial conocido como FACS [15] el cual es un sistema integral basado en la anatomía y se encarga de medir todos los movimientos faciales visualmente discernibles.

Para ello se utilizan como base las unidades de acción (AU), las cuales están compuestas por 44 unidades de acción únicas y se identifican por un código numérico. Por medio de estas unidades de acción y por un conjunto de posiciones y movimientos que abarcan los ojos y la cabeza, es posible describir toda la actividad facial que sea visualmente distinguible.

Este sistema permite codificar la intensidad de cada acción facial en una escala de intensidad de cinco puntos, así como el tiempo de las acciones faciales y la codificación de las expresiones faciales en términos de eventos, donde un evento es la descripción basada en unidades de acción de cada expresión facial, que puede consistir en una sola unidad de acción o muchas unidades de acción contraídas como una sola expresión.

Dimensión	Faceta	Puntaje	Actitudes	Emociones
Responsabilidad	Necesidad de logro	Alto	Orgullo	Alegría
			Determinación	Alegría
		Bajo	Conformista	Tristeza
				Miedo
				Alegría
			Inseguridad	Miedo

Fig. 1. Asociación de las actitudes de la faceta Necesidad de logro con las emociones básicas.

Una de las ventajas que tiene el sistema FACS, es que permite la caracterización de las emociones básicas bajo cierto conjunto de unidades de acción [16], donde existe cierto cambio de intensidad para cada unidad de acción y con ellas se facilita la caracterización de las expresiones faciales.

Al estudiar las emociones básicas es posible conocer cuando estas se manifiestan en el rostro y como se combinan distintas de ellas [17], las cuales pueden aportar diversos significados que pueden ser usados en la conversación no verbal.

6. Caracterización de la personalidad y emociones

A continuación, se presentan los análisis de los modelos de personalidad y de las emociones para caracterizar la influencia que tienen en las expresiones faciales del ACP. Se decidió que el modelo de personalidad FFM es el más adecuado para modelar la personalidad en un ACP, debido a que cuando se revisó el modelo MBTI, se llegó a la conclusión de que existe un mayor grado de dificultad para asociar los tipos de personalidad con las emociones, debido a que los 16 tipos de personalidad que contiene este modelo se basan en las preferencias y diferencias a las que tiende cada persona, incluso si existieran dos personas con el mismo tipo de personalidad, el comportamiento de ambos no sería el mismo, debido a que los distinguen sus rasgos de personalidad.

Por otro lado, el modelo FFM puede ser asociado con mayor facilidad a las actitudes y emociones positivas como negativas, debido a que en este existen escalas en las facetas de puntaje alto y bajo.

6.1. Análisis del modelo FFM en las emociones básicas

Al analizar el modelo de personalidad FFM es posible asociar las emociones básicas con las facetas de este modelo. Para ello se revisaron las facetas de cada dimensión, donde cada faceta tiene cierto nivel de puntaje (alto/bajo), a los cuales se le asocian ciertas actitudes o comportamientos que corresponden a ciertas emociones, donde el término "actitud" puede ser definido como la manifestación de un estado de ánimo, o bien como una tendencia a actuar de un modo determinado.

Para asociar las emociones a las actitudes se hizo un listado de actitudes las cuales tienen una mayor expresividad en el rostro, donde algunas de ellas fueron obtenidas directamente de las definiciones del modelo FFM, donde se especifican que actitudes corresponden a cierto nivel de puntaje.

Dimensiones con mayor influencia emocional del Modelo de Personalidad FFM								
Amabilidad			Extraversión			Neuroticismo		
Confianza	A1		Calidez	E1		Ansiedad	A1	
Franqueza	A2		Gregarismo	E2		Hostilidad	A2	
Altruismo	A3		Asertividad	E3		Depresión	A3	
Actitud Conciliadora	A4		Actividad	E4		Autoconciencia	A4	
Modestia	A5		Búsqueda de Emoción	E5		Impulsividad	A5	
Sensibilidad a los demás	A6		Emociones Positivas	E6		Vulnerabilidad	A6	

Fig. 2. Dimensiones del Modelo FFM con alta influencia emocional.

En algunos casos las definiciones no explicitan algunas actitudes, sin embargo, describen cierto conjunto de características las cuales se pueden asociar fácilmente a ciertas actitudes. Por otro lado, hubo otras actitudes que fueron propuestas tomando como punto de partida los comportamientos y características de las definiciones de las facetas, así como la apariencia física (expresiones faciales) que pueden presenciarse con estas actitudes.

En este trabajo las facetas se consideran como dicotomías, asociándose un puntaje alto o bajo, el cual se considera lo opuesto para su polo en cada faceta. Los niveles de puntaje fueron utilizados como punto de partida para hacer la asociación de las actitudes positivas y negativas para cada dicotomía.

Así mismo, las dicotomías son consideradas con los puntajes más bajos o altos de cada faceta, con esto se pueden asociar actitudes con comportamientos extremos, los cuales son más fáciles de usar para asociarlos con las emociones. Para asociar las actitudes con las emociones, se hizo un análisis de las emociones que comúnmente se presentan en cada actitud de acuerdo con la definición de cada faceta (ya sea en su puntaje alto o bajo), donde se analizan las características y comportamientos descritos, los cuales tienen una alta tendencia a experimentar ciertas emociones.

Cabe mencionar que existen actitudes que pueden ser muy similares entre sí, ya que comparten varias de sus características. Sin embargo, entre ese conjunto de características existe alguna de ellas que hace distinguir cierta actitud de las demás. Los estados de ánimo y los rasgos emocionales no poseen sus propias señales distintivas, sino que son fenómenos afectivos que están saturados con las señales de una u otra emoción [18]. Por decir un ejemplo, una alta incidencia de señales relacionadas con la ira puede sugerir un estado de ánimo irritable o un rasgo hostil.

Al solo existir una emoción positiva es fácil asociar la emoción de alegría con las actitudes positivas, sin embargo, también la emoción de alegría es posible asociarla con las actitudes negativas.

Las emociones negativas que son ira, tristeza, miedo, desagrado y desprecio comúnmente se asocian en actitudes negativas, sin embargo, la emoción de tristeza es posible asociarla en actitudes positivas como el dolor empático.

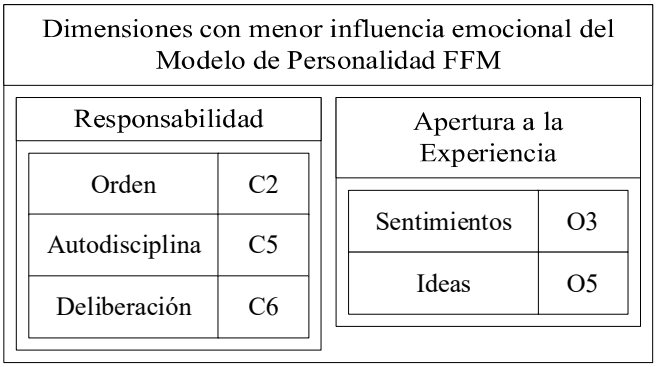


Fig. 3. Dimensiones del Modelo FFM con menor influencia emocional.

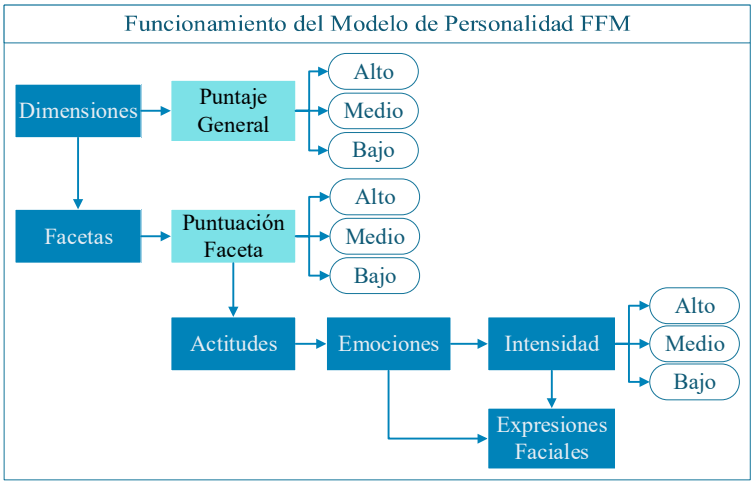


Fig. 4. Funcionamiento general del Modelo de Personalidad FFM.

La emoción de sorpresa es considerada una emoción neutral, por lo tanto, existe una mayor dificultad para asociarla en las facetas, ya que suele presentarse de manera espontánea o acompañada de alguna otra emoción.

A continuación, en la Fig. 1 se presenta un ejemplo de la asociación de actitudes y emociones para la faceta de Necesidad de logro que pertenece a la dimensión de Responsabilidad para ambos puntajes.

El objetivo inicial del análisis del modelo FFM con las emociones básicas, fue el asociar las emociones que comúnmente se presentan en las facetas de cada dimensión del modelo FMM (ya sea con un puntaje alto o bajo).

La asociación de las emociones pudo lograrse al identificar las actitudes que comúnmente se presentan en las diferentes facetas del modelo FFM. Sin embargo, por sí solas las emociones asociadas no contienen algún indicador de cómo pueden ser utilizadas cuando se expresan con un nivel de intensidad.

6.2. Propuesta del modelado de la personalidad en el agente

Como modelo de personalidad se busca usar el modelo FFM, donde se les dio mayor importancia a las dimensiones de Amabilidad (A), Extraversión (E) y Neuroticismo (N), debido a que todas las facetas de estas dimensiones tienen una alta influencia en los cambios emocionales (ver Fig. 2). Por otro lado, esto no significa que no se vayan a tomar en cuenta las dimensiones de Responsabilidad (C) y Apertura a la experiencia (O), sino que ambas dimensiones contienen pocas facetas con una alta influencia en los cambios emocionales (ver Fig. 3).

La personalidad se busca usar con el fin de determinar la manera de actuar (actitudes), así como la manera en la que se reacciona (intensidad de la emoción). Así mismo, se buscan usar las actitudes que fueron identificadas en el análisis de la asociación del modelo FFM con las emociones básicas, debido a que estas ya se encuentran relacionadas con las emociones.

De esta manera, al plasmar las actitudes también se incluyen las emociones, además, al usar las actitudes no solo se transmite una emoción, también se puede dar a conocer un significado más preciso que el de solo una emoción.

A continuación, se muestra en la Fig. 4 la relación que existen entre los distintos elementos del modelo de personalidad FFM en el agente, donde se busca utilizar los puntajes de las dimensiones y facetas obtenidos del test de personalidad.

A partir de los puntajes de cada faceta, se obtienen las actitudes que se encuentran asociadas a las facetas, donde se busca realizar la selección de la emoción final con cierta intensidad, las cuales pueden ser caracterizadas por medio de expresiones faciales.

6.3. Factores que influyen en la personalidad

Bajo ciertas situaciones las personas reaccionan de manera diferente, esto se debe a las diferencias que existen en sus perfiles de personalidad. Para determinar la manera de actuar de una persona dado ciertos eventos, es necesario conocer los factores que influyen en la activación de las distintas facetas del modelo de personalidad, los cuales van desde factores internos a externos.

Los factores externos, son aquellos que pueden ser percibidos del entorno, en una conversación con una persona es posible identificar algunos de ellos. Por otro lado, los factores internos son aspectos personales que influyen en el comportamiento (ver Fig. 5).

En la creación de un ACP se busca que el agente sea capaz de determinar los factores externos a partir de los actos del habla. Así mismo, los factores internos estarán definidos en una base de conocimientos, donde dependiendo de la configuración de los atributos que se le asignen al agente, este actuará de una manera determinada.

Se busca que los factores internos y externos influyan en el modelo de personalidad, mediante la activación de las facetas del perfil de personalidad, donde dependiendo de las circunstancias o eventos, se activen o desactiven un conjunto de facetas que pueden pertenecer a diferentes dimensiones del modelo FFM, las cuales definirán las posibles actitudes que pueden ser caracterizadas en el rostro con cierta intensidad y emoción.

A continuación, en la Fig. 6 se muestra un ejemplo de la activación de las facetas, donde un color rojo se refiere a la activación de una faceta con puntaje bajo y un color

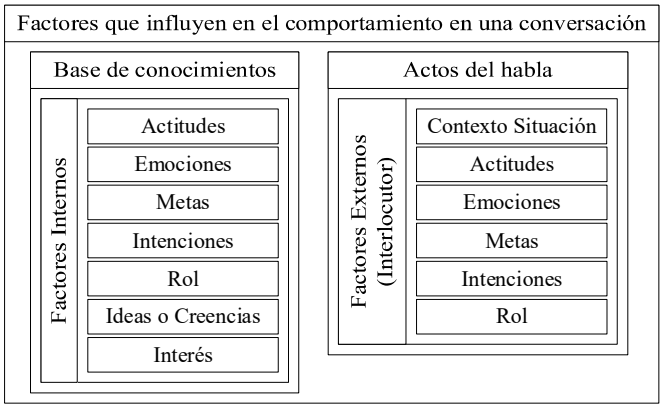


Fig. 5. Factores externos e internos que influyen en el comportamiento.

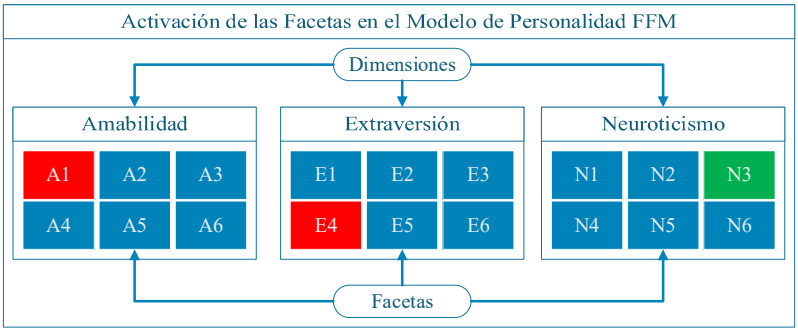


Fig. 6. Ejemplo de activación de distintas facetas del Modelo de Personalidad FFM.

verde se refiere a un puntaje alto. Cabe destacar que el perfil de personalidad no presentará cambios, sino que, dependiendo de ciertos eventos, existirá una activación y desactivación de facetas, donde los eventos que desencadenan la activación de las facetas variaran dependiendo del perfil de personalidad.

6.4. Propuesta de caracterización de emociones e intensidad

Para la caracterización de las emociones (ver Fig. 7), se buscar utilizar el puntaje general de cada dimensión del modelo de personalidad FFM para determinar la emoción final. Cuando suceda algún evento, en cada dimensión existirá la activación de ciertas facetas, de las cuales se busca hacer un listado de ellas, donde se tome en cuenta el puntaje de cada una para determinar la emoción final.

Con la activación de las facetas, las actitudes asociadas a los niveles de puntaje de cada faceta se hacen presentes. Por lo tanto, se busca hacer un listado de las actitudes que comúnmente se presentan para hacer un conteo de cuáles son las que más se repiten.

Así mismo, de este listado de actitudes es posible crear un listado de emociones, ya que estas se encuentran asociadas a las actitudes y se busca hacer un conteo de cuáles son las emociones que más se repiten para determinar una emoción final.

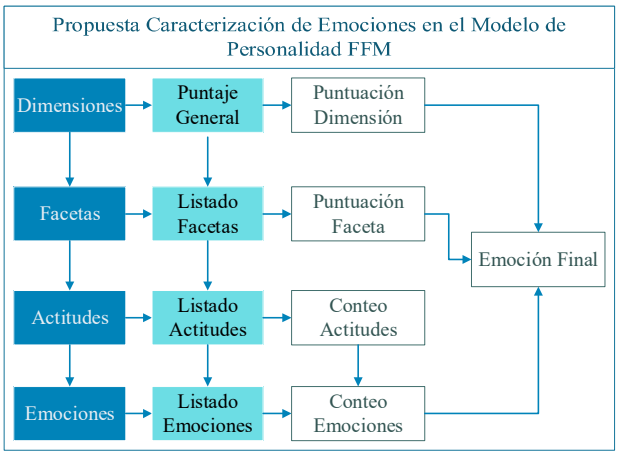


Fig. 7. Propuesta de caracterización de emociones a partir de los puntajes de las dimensiones y facetas, así como del conteo de actitudes y emociones.

Para determinar la intensidad de las emociones se buscan usar como punto de partida los rostros descritos por [17], los cuales ya cuentan con una descripción de las intensidades de las emociones que se presentan en el rostro, lo cual ayuda a la caracterización de expresiones faciales, ya que indica las zonas del rostro que deben ser modificadas para crear el rostro con la intensidad deseada.

Por otro lado, la propuesta que se tiene para caracterizar la intensidad de las emociones busca tomar en consideración la dimensión, la faceta y la actitud con mayor peso.

Cabe destacar que los pesos no son fijos, cambian dependiendo de las circunstancias que existan a través de los factores externos e internos. La dimensión que tendrá el mayor peso dependerá de la faceta a la que está asociada con el mayor peso, así mismo, la actitud con mayor peso estará asociada a la faceta con mayor peso. En la Fig. 8 se muestra el diagrama de la propuesta de caracterización de la intensidad.

7. Conclusiones y trabajo a futuro

En este trabajo se presenta la asociación del modelo de personalidad FFM con las emociones básicas a través de un listado de actitudes, donde dependiendo de la puntuación de las facetas del test de personalidad FFM [19] se identifican las actitudes que comúnmente se presentan.

Con esto se propone la caracterización de expresiones faciales emocionales, ya que estas se encuentran relacionadas con las actitudes. Con esta caracterización se buscan sentar las bases de un modelo de selección de expresiones faciales para un ACP socio- emocional.

La definición de la determinación de la intensidad de las emociones aún se encuentra en desarrollo, dado que existen diferentes emociones que pueden ser asociadas a una misma actitud. Otro punto que se encuentra en desarrollo es la caracterización de

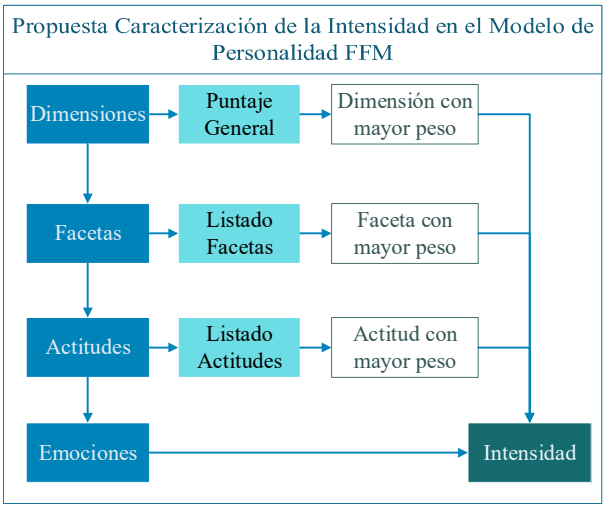


Fig. 8. Definición de la intensidad emocional a partir de la dimensión, faceta y actitud con mayor peso.

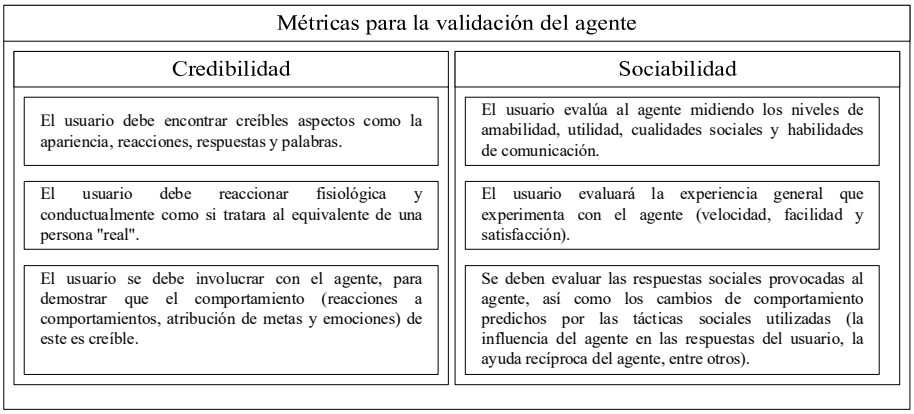


Fig. 9. Métricas de validación para el agente.

actitudes, donde se puede identificar para las diferentes dimensiones del modelo de personalidad FFM más de una actitud en cualquiera de sus facetas.

De momento, se está considerando que la caracterización de las actitudes va a depender de las puntuaciones que tienen las dimensiones del modelo de personalidad FFM, así como de la activación y puntajes de las facetas de este modelo, las cuales se pueden activar en conjunto dependiendo de las situaciones, roles e intenciones tanto del agente como del interlocutor. Así mismo, también se debe analizar cuáles son las facetas que tienen mayor peso para desencadenar ciertas actitudes.

Uno de los contextos en los que se busca usar el agente es en la aplicación de un chatbot informativo sobre el Covid-19. En donde se busca darle el rol de doctor al agente y reforzar mediante el uso de expresiones faciales el diálogo desempeñado.

Con el trabajo propuesto en este artículo, se busca la caracterización de la influencia de la personalidad y emociones como atributos de un proceso de selección de expresiones faciales de un ACP, a fin de contribuir a la credibilidad, sociabilidad y realismo en la creación de éstos. En la Fig. 9 se muestran las métricas que se buscan utilizar para la validación del agente las cuales son definidas por [2].

Agradecimientos. Se agradece al CONACYT el apoyo otorgado a través de la Beca para Estudios de Maestría al CVU 1007913. Este proyecto fue apoyado por el CONACYT mediante el proyecto de Cátedras CONACYT con número 3058 y Ciencia Básica 2017-2018 con número A1-S-11012. Se agradece al Tecnológico de Ciudad Madero (ITCM), así como a los profesores que me han compartido de sus conocimientos, los cuales me han hecho crecer como persona. También se le agradece a mi asesora la Dra. María Lucila Morales Rodríguez, quien me ha estado guiando y me ha tenido paciencia en la creación de este proyecto.

Referencias

1. Ekman, P., Friesen, W. V., Ellsworth, P.: Emotion in the human face: Guidelines for research and an integration of findings. Pergamon Press, INC., pp. 1–190 (1972)
2. Isbister, K., Doyle, P.: The blind men and the elephant revisited. In Ruttkay, Z., Pelachaud, C. (Eds): From Brows to Trust: Evaluating Embodied Conversational Agents. Springer Netherlands, pp. 3–26 (2004) doi: 10.1007/1-4020-2730-3_1
3. Yik, M. S. M., Russell, J. A., Ahn, C. K., Fernández-Dols, J. M., Suzuki, N.: Relating the five-factor model of personality to a circumplex model of affect. Springer (2002) doi: 10.10 07/978-1-4615-0763-5_5
4. Castro, J.: Modelado de la personalidad en modelos preferenciales multicriterio a través de agentes virtuales inteligentes. ITCM, Tesis (2018)
5. Morales, M.: Modèle D'interaction sociale poru des agents conversationnels animés. Application à la rééducation de patients cerebro-lésés, PhD Tesis, Toulouse, Université Paul Sabatier (2007)
6. Florencia, R.: Agente conversacional corpóreo que utiliza AIML para integrar procesos de personalidad. ITCM (2010)
7. Medellín, F.: Modelo de caracterización y selección de expresiones no verbales para su aplicación en agentes virtuales inteligentes. ITCM (2010)
8. Myers, I. B.: Introducción al Type (MBTI). CPP (2001)
9. Costa, P. T., Jr., McCrae, R. R.: Revised NEO personality inventory (NEO PT–R) and NEO fivefactor inventory (NEO–FFI) professional manual. Psychological Assessment Resources (1992)
10. McCrae, R. R., Costa, P. T., Jr.: Introduction to the empirical and theoretical status of the five-factor model of personality traits. In Widiger, T. A., Costa, Jr. P. T., (Eds): Personality disorders and the five-factor model of personality, American Psychological Association, pp. 15–27 (2013)
11. Reeve, J.: Motivación y emoción. McGraw-Hill (2010)
12. Fridlund, A. J., Ekman, P., Oster, H.: Facial expressions of emotion: Review of literature, 1970-1983. In Siegman, A. W., Feldstein, S. (Eds): Nonverbal Behavior and Communication, Lawrence Erlbaum Associates, pp. 143–224 (1987)
13. Tomkins, S. S.: Affect, imagery, consciousness, Vol. 1: The positive affects. Springer (1962)

14. Tomkins, S. S. Affect, imagery, consciousness, Vol. 2: The negative affects. Springer (1963)
15. Ekman, P., Rosenberg, E. L.: What the face reveals: Basic and applied studies of spontaneous expression using the facial action coding system (FACS). Oxford University Press (2005)
16. Farnsworth, B.: Facial action coding system (FACS) – A visual guidebook. IMOTIONS, (2019) http://imotions.com/blog/facial-action-coding-system/#mainaction-units
17. Ekman, P., Friesen, W. V.: Unmasking the face: A guide to recognizing emotions from facial clues. Prentice-Hall (1975)
18. Ekman, P.: Basic emotions. In Dalgleish, T., Power, M. J. (Eds): Handbook of cognition and emotion. John Wiley & Sons Ltd, pp. 45–60 (1999)

ISSN 1870-4069

Sistema de aprendizaje del movimiento de labios utilizando Q-Learning y redes neuronales convolucionales

Leonardo Nevárez Porras, Hernán de la Garza Gutiérrez, Carlos Humberto Rubio Rascón, Arturo Legarda Sáenz, Marisela Ivette Caldera Franco

[1] Tecnológico Nacional de México/Campus Chihuahua II, Chihuahua, México

{mm19550853, hernan.gg, carlos.rr, arturo.ls, marisela.cf}@chihuahua2.tecnm.mx

Resumen. Se describe un sistema de aprendizaje del movimiento de los labios a partir de un audio sin haberle dado tratamiento previo, de una persona hablando en español. Como salida se genera una animación 2D a 30 cuadros por segundo de un personaje que muestra el movimiento de los labios generado a partir de la implementación de la estrategia de Aprendizaje por Refuerzo, que incluye el uso de una red neuronal convolucional profunda, entrenada con el algoritmo de Q- Learning.

Palabras clave: Aprendizaje por refuerzo, Q-Learning, animación, audio, red neuronal convolucional, sincronización de labios.

Lips Movements Learning System with Q-Learning and Convolutional Neural Network

Abstract. We describe a learning system for lips' movements, which takes raw human speech audio in Spanish. The system creates a 2D animation at 30 frames per second of a character that shows the movements of the lips, generated by the implementation of the Reinforcement Learning strategy, that includes a convolutional neural network trained using the Q-learning algorithm.

Keywords: Reinforcement learning, Q-Learning, animation, audio, convolutional neural network, lip synchronization.

1. Introducción

En la actualidad, la generación de material digital es de gran importancia y utilidad, en diferentes ámbitos como son en la enseñanza a distancia y en la creación de contenido de entretenimiento, además la producción de video con personajes animados requiere de herramientas especializadas y de personas que sepan usarlas. Dependiendo

de la naturaleza de la animación y los movimientos de los personajes se puede requerir de una refinación de dichos movimientos para que se asemejen a los movimientos naturales de las personas de manera que se ajusten al contenido [1]. A partir de este tipo de retos, surgen sistemas de automatización de la animación, algunos de ellos apuntados a generar el movimiento de los labios, ya sea a partir del movimiento original del actor de voz [2] o mediante otras técnicas [1, 3].

El sistema que se describe a continuación, pertenece a este grupo y busca generar los movimientos de labios a partir de un audio en español utilizando técnicas de Aprendizaje por Refuerzo. Recientemente han emergido técnicas de aprendizaje de máquina que permiten entrenar redes neuronales profundas utilizando el algoritmo de propagación hacia atrás [4], las cuales pueden aprender a partir de datos sin procesamiento previo, tales como imágenes, video y audio [5].

El proyecto que se presenta, busca aprender a imitar el movimiento de labios a partir de un video, del cual se hace la separación de las imágenes y del audio. El audio se toma como entrada al módulo de Aprendizaje por Refuerzo para obtener una posición de los labios de salida, la cual se compara con las posiciones reales obtenidas de las imágenes del video y como resultado de dicha comparación generar la recompensa que promueva el aprendizaje.

Principalmente se utiliza una Red Neuronal Convolucional (RNC) la cual describiremos a fondo más adelante, para estimar una función que entrega las expectativas de recompensa de un agente al mover unos labios basado en la señal de audio. En el contexto del trabajo actual, nos referimos a un agente como la parte del sistema que percibe un estado del ambiente y toma acciones. En la figura 1, dentro del módulo de aprendizaje se pueden apreciar estos elementos, más el intérprete, que juntos forman parte del aprendizaje por refuerzo.

2. Trabajos relacionados

Se describen algunos de los trabajos publicados y que tienen relación con las tres principales áreas de nuestro proyecto: Sincronización de labios, Aprendizaje de Máquina y Sistemas de visión.

2.1. Sincronización de labios

Existen trabajos que apuntan a crear avatares digitales con sincronización de labios tomando varios enfoques:

El trabajo realizado en [1], genera una animación en 2D basándose en un audio en español, sin embargo, la animación está estilizada como caricatura, de manera que el muestreo es bajo, genera poca credibilidad y la sencillez de la animación generada busca no causar distracciones al usuario. En [3] se utiliza un enfoque similar para audio en inglés. Además de generar el avatar 2D en tiempo real (con pequeño retraso).

En el trabajo desarrollado por [2] se creó un sistema capaz de generar videos realistas de Barack Obama a partir de un audio del mismo Obama. El sistema se entrena primero con hasta 15 horas de video tomado de los reportes presidenciales semanales de Obama. Este sistema solo es capaz de generar contenido con la voz de un solo personaje y

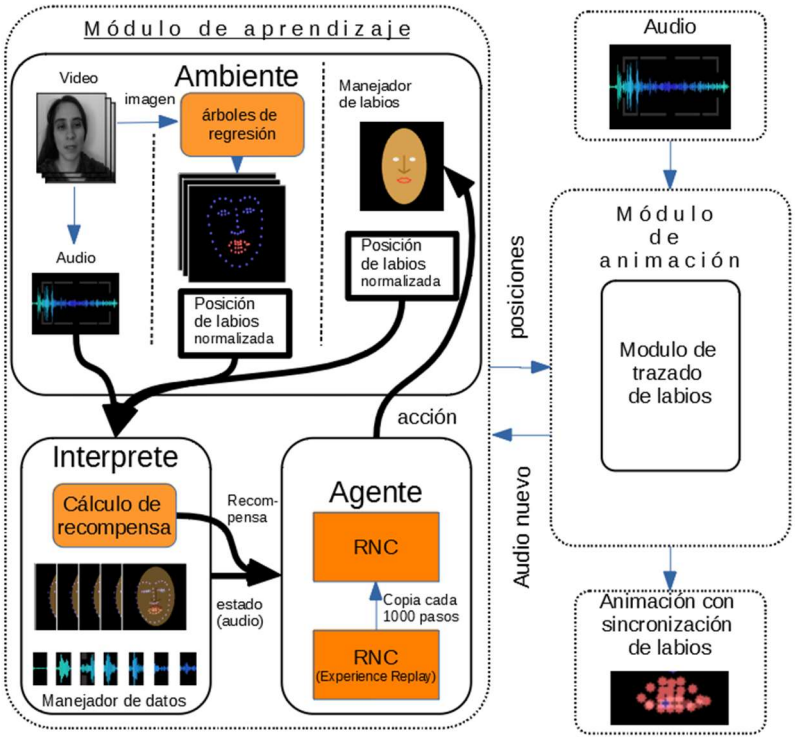


Fig. 1. Esquema general de las partes del sistema.

necesita alrededor de 15 horas de video de referencia además de procesamiento posterior adicional para producir resultados realistas.

Otros trabajos similares de síntesis de avatares a partir de un audio utilizando aprendizaje automático [6, 7, 8] hacen uso de material en otros idiomas para entrenar sus modelos, por lo que son aptos para generar material en otros idiomas distintos al español, además de utilizar técnicas de Aprendizaje Supervisado. Nuestra propuesta crea un avatar capaz de mover sus labios a partir de un audio en español utilizando técnicas de Aprendizaje por Refuerzo.

2.2. Aprendizaje de máquina

Avances recientes en Redes Neuronales Profundas permiten entrenar estas redes para procesar datos sin un tratamiento previo, tales como audio y video [4].

El algoritmo descrito en [2] para entrenar a un agente a jugar al Atari, permite entrenar una RNC que estima una función Q(s,a) que toma como entrada un estado *s* determinado por los pixeles de la pantalla actual y cuya salida representa la recompensa esperada al tomar la acción *a* dado el estado *s*. Al tomar la acción que maximice la recompensa y con suficiente exploración e iteraciones se logra obtener una aproximación lo suficientemente cercana a la función Q(s,a) real para superar retos en distintos juegos de Atari.

Para procesar audio existen enfoques como el de [9] y [10] donde se utiliza una RNC Profunda, es decir, de varias capas. Este tipo de arquitecturas permiten procesar audio en forma de onda directamente y crear a partir del entrenamiento, filtros que extraen de forma secuencial características del audio cada vez más abstractas.

Otros trabajos como [11] utilizan representaciones del audio en espectrogramas los cuales se alimentan a RNC similares a las utilizadas para procesar imágenes y video con convoluciones en 2D.

El sistema aquí presentado hace uso de estas técnicas para extraer características del audio sin ser procesado previamente, y generar animaciones a partir de éste.

2.3. Sistemas de visión

Para entrenar a un agente de Q-learning a que mueva los labios de manera correcta siguiendo un audio, se requiere de una señal de recompensa que es un número real, que funciona como una medida del rendimiento del agente al imitar el movimiento de los labios de manera similar al puntaje de un videojuego como en [5].

Como en otros trabajos de sincronización de labios [6, 7], el modelo se somete a un entrenamiento durante el cual se debe poder cuantificar la diferencia entre la posición de los labios del locutor original del audio y la posición de los labios generada por el agente a partir del mismo audio.

Por esta razón, se debe poder ubicar la posición de los labios tanto originales como los generados por el agente, ya sea ubicando puntos selectos en labios inferior y superior o de un trazado completo de estos. Existen diferentes métodos para ubicar los puntos de referencia en la cara.

Uno de ellos es el descrito en [12], el cual consta de un conjunto de árboles de regresión el cual se puede entrenar en una base de datos de caras etiquetadas con la ubicación de distintos puntos de referencia, incluyendo los labios.

3. Sistema de sincronización de labios

Después de un entrenamiento de preparación y teniendo un audio sin procesamiento previo de una persona hablando en español como entrada, el sistema de labios debe imitar el movimiento de labios del locutor y generar una animación de los labios. La estrategia mencionada se identifican las siguientes etapas:

1. Se captura el video de entrenamiento de una persona hablando de frente a la cámara, con toda la cara visible a 30 cuadros por segundo.
2. Para cada cuadro del video, se extraen las posiciones de puntos clave de los labios mediante árboles de regresión en forma de coordenadas de un plano cartesiano, y se acoplan con el segmento correspondiente de 300 milisegundos de audio.
3. Comenzando por el primer cuadro del video, uno a uno se alimentan las posiciones de labios y audio correspondiente. Con esto, el módulo de aprendizaje hace modificaciones internas a sus propios parámetros de forma que aprende a predecir las acciones que aproximan más cercanamente la posición de labios del locutor. La forma en que se lleva a cabo el aprendizaje se detalla en la sección 3.1.

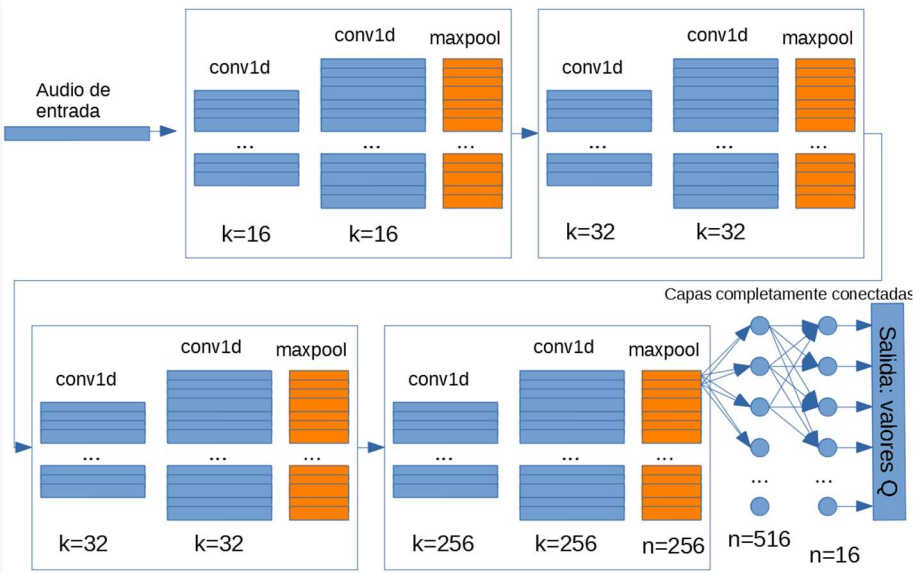


Fig. 2. Arquitectura de la Red Neuronal Convolucional.

4. Una vez llevado a cabo el aprendizaje, el módulo de animación hace uso de la experiencia del módulo de aprendizaje para generar una animación de labios con audio distinto al utilizado en el entrenamiento. La implementación del módulo de aprendizaje se explica a fondo en la sección 3.2.

El modelo general del sistema se puede ver en la Fig. 1. A continuación, se describen los módulos del sistema:

3.1 Módulo de aprendizaje

El Aprendizaje por Refuerzo se puede describir como un agente interactuando en un ambiente que cambia de estado a partir de las acciones tomadas por el agente y a su vez presenta una señal de recompensa la cual el agente busca maximizar [5].

Existen múltiples trabajos de sincronización de labios con personajes digitales basados en técnicas de aprendizaje supervisado tales como [2, 3, 6], sin embargo, buscamos explorar las técnicas del aprendizaje por refuerzo para posteriormente poder compararlas con los resultados obtenidos mediante aprendizaje supervisado.

El módulo de aprendizaje consta de los modelos y algoritmos necesarios para aprender el movimiento de labios después de un entrenamiento y se define en base a el marco de Aprendizaje por Refuerzo descrito en el párrafo anterior y también según [13] que consta de:

- Una política: Es una relación de estados a acciones que le dice al agente cual acción tomar en cada estado.
- Una señal de recompensa: La cual se busca maximizar para encontrar la política óptima.

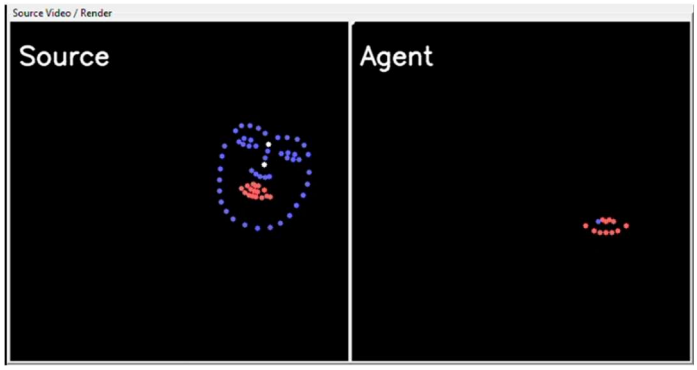


Fig. 3. Interfaz gráfica del módulo de animación.

– Una función de valor (de estado-acción): Que indica la recompensa que se espera recibir a partir de ese estado o estado-acción.

Al igual que en otros trabajos de Aprendizaje por Refuerzo, buscamos maximizar una señal de recompensa, la cual en este caso indica qué tan cercanamente los movimientos de los labios del agente, siguen a los del locutor original. Suponemos que las recompensas en el ambiente están dadas por la función Q(s,a) la cual podemos aproximar iterativamente mediante (1) como se describe en [13]:

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[R_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t)], \quad (1)$$

donde *s* es el estado actual del ambiente dado por la señal de sonido, *a* la acción tomada por el agente en el estado *s*, elegida entre 16 acciones diferentes. $Q(s_t,a_t)$, que representa la recompensa que el agente espera ganar a partir de un estado s_t tomando la acción a_t puede ser inicializada con un valor arbitrario para cada s_t,a_t.

El valor de α es el ritmo de aprendizaje y puede ser cualquier valor entre 0 y 1; R_{t+1} es la recompensa obtenida durante la transición de estado, γ es el valor de descuento que define la importancia que se le da a la recompensa futura mientras que (a)max Q (s_{t+1}, a) - $Q(s_t,a_t)$ es el valor máximo de recompensa a partir del estado siguiente tomando la acción *a*.

Esta actualización iterativa aproxima directamente q_* que es la función acción-valor óptima que maximiza la recompensa [13]. Sin embargo, ya que resulta impráctico el crear una función Q(s,a) debido a los recursos computacionales y el hecho de que buscamos poder generalizar, es práctica común en este tipo de problemas el utilizar funciones de aproximación estando entre ellas las Redes Neuronales de Convolución que dan muy buenos resultados para aproximar la función Q(s,a) como en [5].

Sin embargo, surgen dificultades a partir de usar una Red Neuronal para aproximar la función Q(s,a), principalmente el que la Red Neuronal tiende a no converger en la función óptima, al estar iterando sobre una política cambiante que depende en las salidas de la misma red.

Este problema se resuelve utilizando un mecanismo llamado *experience replay,* que toma muestras aleatorias de experiencias pasadas para entrenar la red, así como también limitando las actualizaciones a la Red Neuronal que representa a la función Q. Se utilizan dos redes, una para tomar las decisiones directamente y otra que se entrena en

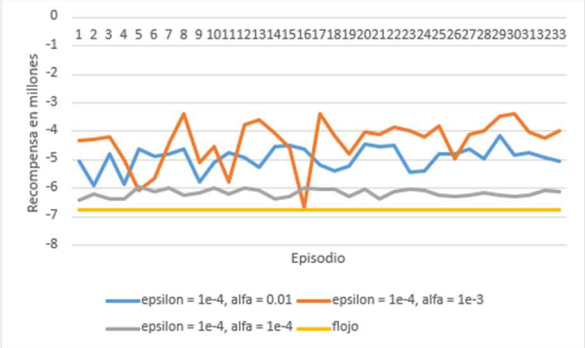


Fig. 4. Recompensa en millones por episodio, con ε = 1e-4 y diferentes valores de α, comparada con el agente flojo.

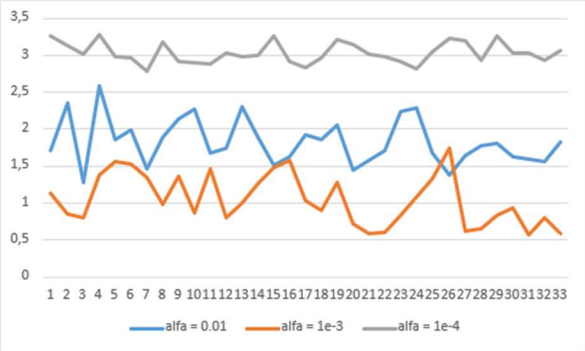


Fig. 5. Pérdida en millones por episodio, con ε = 1e-4 y diferentes valores de α.

cada transición de estado y cuyos cambios se copian a la primera red cada 1000 pasos [5], ilustrado en la Fig. 1., Agente Q-learning.

Como se menciona en [13], cabe destacar que el entrenamiento de la RNC que estima la función Q se puede ver como un problema de aprendizaje supervisado, ya que se cuenta con datos (el estado) y una etiqueta que es el valor de recompensa potencial hacia el cual queremos acercar las predicciones de la RNC.

La forma en que se implementa (1) es en el entrenamiento de la RNC, donde se calcula el valor de salida esperado, el cual se utiliza a su vez para obtener la pérdida con la cual entrenamos a la RNC.

Arquitectura de la Red Neuronal Convolucional. La RNC del sistema, ilustrada en la Fig. 2, está basada en la Red descrita en [14] y consta de convoluciones, agrupamientos y al final, capas completamente conectadas.

A cada dos capas de convolución le siguen una de agrupación tomando los mayores valores de cada segmento (*max-pooling*) y después se aplica una desactivación aleatoria (dropout) del 10% para prevenir sobreajuste (overfitting), todo esto, desde las convoluciones, agrupación y desactivación aleatoria se hace cuatro veces en cadena.

De esta forma tenemos dos capas de convolución con 16 filtros cada una, dos con 32 y dos capas más con 256 filtros, cada par seguido por agrupamiento y desactivación aleatoria.

La salida de las capas de convolución representa las características del audio extraídas a partir de los filtros de convolución y esta salida se alimenta a la segunda parte de la Red Neuronal la cual consiste en dos capas de neuronas completamente conectadas las cuales generan un valor numérico por cada acción disponible en el ambiente. El valor de cada salida representa el estimado de la función Q(s,a) donde *a* y *s* son los datos de entrada a la Red Neuronal.

Cálculo de la recompensa. La recompensa en el Aprendizaje por Refuerzo es un número real que le dice al agente qué tan bien se está desempeñando en el ambiente. Se busca que el agente siga los labios del locutor lo más cercanamente posible, por lo que durante el entrenamiento la recompensa debe ser una medida de qué tan cerca están siguiendo los labios controlados por el agente a los del locutor.

Los labios del locutor se representan como 4 pares de valores que ubican las coordenadas de 4 diferentes puntos de los labios en una imagen. Estos puntos son las comisuras izquierda y derecha, el centro del labio superior y centro del labio inferior. Los puntos de los labios del agente se representan de la misma manera.

La recompensa en cada paso se calcula con el siguiente algoritmo:

Entrada: *Posiciones de los labios del agente y el locutor.*

Para labios del agente y locutor, calcular:

distancia_labio_superior = centro_labio_superior_y – centro_boca_y

distancia _labio_inferior = centro_labio_inferior_y – centro_boca_y

distancia _comisuras = comisura_izq_x – centro_boca_x

a = agente.distancia_labio_superior - locutor.distancia_labio_superior

b = agente.distancia_labio_inferior - locutor.distancia_labio_inferior

c = agente.distancia_comisuras - locutor.distancia_ inferior

recompensa = - (a^2 + b^2 + c^2)

De esta forma, la recompensa es el negativo de la suma de los cuadrados de las distancias entre los puntos clave de los labios del locutor y el agente. Y sirve como una medida de qué tan cerca sigue a los labios del locutor conforme avanza el entrenamiento, especialmente al promediarlo para varios pasos. Se utiliza el cuadrado de las distancias para disminuir la penalización cuando las posiciones de la boca son ligeramente diferentes.

A través del entrenamiento el agente busca maximizar la recompensa, o en este caso, reducir la penalización, al aproximar la recompensa a cero.

Durante el entrenamiento, se alimenta la RNC con 300 milisegundos de audio, sin tratamiento previo, es decir en forma de onda y se busca que la RNC determine la acción que brinde un mayor valor esperado de recompensa. De esta forma se obtiene la política del agente.

Después de tomar la acción con mayor valor, se compara la recompensa real con el estimado generado por la red neuronal, la diferencia se conoce como error y se utiliza para cambiar ligeramente los pesos de la red e irla entrenando.

Uno de los principales problemas del Aprendizaje por Refuerzo está en encontrar un balance entre exploración (de las partes desconocidas) y explotación (del conocimiento) [13]. En este caso, la exploración está representada por ε (épsilon) cuyo valor indica el

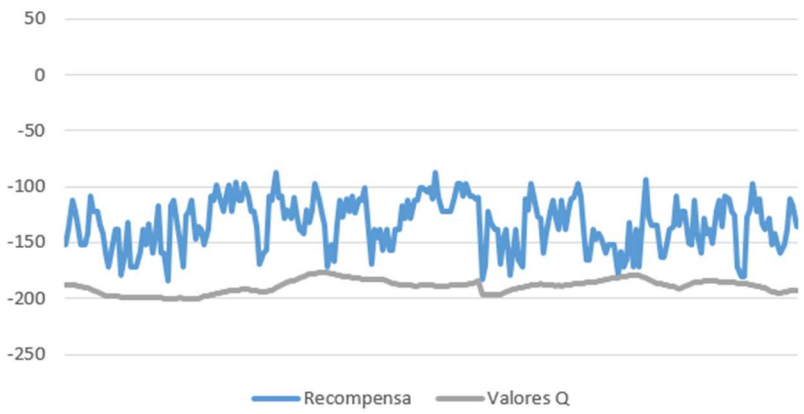


Fig. 6. Recompensa acumulada en millones por episodio, con ε = 1e-4 y diferentes valores de α.

grado de exploración deseado, es el porcentaje de acciones que se escogerán aleatoriamente del espacio de acciones sin importar que exista una acción conocida que entregue una recompensa máxima.

El sistema utiliza un valor de ε que disminuye conforme avanza el entrenamiento y también se experimentó con distintos valores de ε constantes. En la figura 3 podemos observar la recompensa promedio por episodio para 33 episodios de entrenamiento con diferentes valores de α (ritmo de aprendizaje) y un valor de ε fijo. Mientras que en la figura 4 se observa la pérdida correspondiente, la cual se busca disminuir con el entrenamiento.

3.2 Módulo de animación

Este módulo genera una animación de labios siguiendo un audio como entrada, a esto también se le conoce como sincronización de labios. Las entradas de este módulo son:

- Los pesos de la RNC, previamente entrenada.
- Un archivo de audio en formato WAV a 16khz con monólogo en español

La salida del módulo es una secuencia de imágenes (30 imágenes por cada segundo de audio de entrada) que representan el movimiento de los labios generado a partir de las acciones del agente.

A continuación, se describe la forma en que se generan las imágenes:

1. Inicializar el Manejador de Labios que funciona como una interfaz para que el agente manipule la posición de unos labios virtuales, la posición inicial de los labios es cerrada.
2. Cargar el audio en memoria. Inicializar Índice Audio para apuntar al elemento del audio en la posición 16000/30, redondeando hacia abajo.

3. Tomar el segmento de audio con inicio en [Índice Audio - 16000*0.30] y final en Índice Audio y alimentar la RNC con este segmento como entrada, para obtener la acción que maximiza la recompensa.
4. Tomar la acción que maximiza la recompensa, generar una imagen con la nueva posición de labios.
5. Índice Audio <= Índice Audio + 16000*0.30.
6. Ir a paso 3 hasta procesar todo el audio.

Al combinar las imágenes generadas en los pasos descritos anteriormente con el audio de entrada, podemos producir un video con sincronización de labios. Se puede observar la interfaz de usuario en la Fig. 3.

3.3 Experimentos y resultados

Los experimentos realizados consisten en el entrenamiento del modelo con un video de 32 minutos de una persona leyendo un texto de frente a la cámara. El entrenamiento consistió en 33 episodios o 1 millón de pasos.

Se parte de dos premisas relevantes: se alimentará la RNC con el audio sin procesar, buscando que la RNC sea capaz de extraer la información importante para el aprendizaje y la segunda premisa es que no se le proporcionarán las posiciones de los labios del agente, viendo si también la RNC es capaz de determinar la acción correspondiente sin esa información.

Como punto de comparación se toma la recompensa obtenida por un agente sin movimiento que mantiene una posición de boca cerrada, a este agente lo denominamos agente flojo. Con el agente flojo vamos a obtener un nivel base de recompensas cuando no se hacen acciones y permitirá hacer comparaciones con los modelos que sí realicen acciones. En la Fig. 4 podemos comparar la recompensa a través de 33 episodios con diferentes valores de α, el agente busca maximizar la recompensa a través del aprendizaje, explorando diferentes políticas.

Otra forma de medir el aprendizaje de la RNC es por la pérdida promedio durante un número de pasos. La pérdida se puede describir como una medida del error en las predicciones de la RNC por lo que buscamos que disminuya con el entrenamiento. En la Fig. 5 podemos observar la pérdida por episodio del sistema para diferentes valores de α. Una perdida que disminuye a través de los episodios indica que la red va mejorando en sus predicciones.

Al visualizar la recompensa para los últimos pasos del primer episodio, en la Fig. 6, podemos observar que a pesar de que el agente es capaz de estimar los cambios más burdos en la recompensa por episodio, no puede discriminar los cambios más pequeños, lo cual se ve reflejado en las acciones de movimiento que toma, es decir, al observar la animación de labios generada se observa que el agente da preferencia a los movimientos del labio inferior el cual tiene un mayor rango de movimiento y por ende mayor influencia en la recompensa resultante. Esto podría ser corregido utilizando una fracción de la distancia de los labios inferiores en lugar de la distancia total.

4. Conclusiones y trabajo a futuro

A partir de los experimentos realizados se obtiene un movimiento de labios que se asemejan a los del locutor original, demostrando con ello que funciona el Aprendizaje por Refuerzo. A partir de la recompensa total por episodio de varios agentes y comparándola con los resultados de un agente flojo, podemos ver que el sistema de Aprendizaje por Refuerzo, obtiene mayor recompensa con α = 1e-3 y ε = 1e-4. Se observa que la animación sigue de forma general los labios del locutor.

Se planea comparar el rendimiento del sistema con el de otros trabajos que utilizan distintas técnicas y plataformas, por el momento la métrica de comparación es la recompensa promedio por episodio, lo cual permite comparar el aprendizaje de distintos modelos que utilizan el mismo entorno. Otro plan es el de probar la capacidad de generalización de la técnica de aprendizaje por refuerzo aquí aplicada a la sincronización de labios en comparación con otras técnicas como las de aprendizaje supervisado.

Para este trabajo se limitó el entrenamiento a 33 episodios, pero se puede dar tanto tiempo como el agente requiera para quedar completamente entrenado. Una variante probada limitadamente hasta el momento es la duración del audio que recibe el agente en cada paso, para este trabajo fue de 300 milisegundos. Se experimentará con ventanas de audio con diferente duración buscando mejorar los resultados.

Otra forma de incrementar el aprendizaje del agente puede lograrse si a la salida de la última serie de capas de convolución y de agrupamiento de la Red, se le suman las posiciones de los labios del agente, y este nuevo conjunto de datos sea el que se alimente a las capas finales de la red. Esto tendría el potencial de acelerar el entrenamiento, ya que la recompensa depende en parte de la posición de labios del agente.

Adicionalmente, se sugiere crear un modelo similar que reciba el audio en forma de espectrograma, esto daría la facilidad de poder usar modelos adicionales probados en aplicaciones similares tal como en [11].

Referencias

1. Cabiedes F., Pelczer, I., Gamboa F., Bretón J., Rodríguez S.: Sincronización de labios: Método sin visemas. Revista Iberoamericana de Educación a Distancia, vol. 10, no. 1, pp. 37–50 (2007) doi: 10.5944/ried.1.10.1012
2. Suwajanakorn S., Seitz, S. M., Kemelmacher-Shlizerman, I.: Synthesizing Obama: Learning lip sync from audio. ACM Transactions on Graphics, vol. 36, no. 4, pp. 1–13 (2017) doi: 10.1145/3072959.3073640
3. Aneja, D., Li. W.: Real-time lip sync for live 2D animation. ArXiv 1910.08685v1 (2019) doi: 10.48550/arXiv.1910.08685
4. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature no. 521, pp. 436–444 (2015) doi: 10.1038/nature14539
5. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., Hassabis, D.: Human-level control through deep reinforcement learning. Nature 518, pp. 529–533 (2015) doi: 10.10 38/nature14236

6. Fan, B., Xie, L., Yang, S., Wang, L., Soong, F. K.: A deep bidirectional LSTM approach for video-realistic talking head. Multimedia Tools and Applications archive, vol. 75, no. 9, pp. 5287–5309 (2015) doi: 10.1007/s11042-015-2944-3
7. Karras, T., Aila, T., Laine, S., Herva, A., Lehtinen, J.: Audio-driven facial animation by joint end-to-end learning of pose and emotion. ACM Transactions on Graphics, vol. 36, no. 4, pp. 1–12 (2017) doi: 10.1145/3072959.3073658
8. Xu, Y., Feng, A. W., Marsella, S., Shapiro, A.: A practical and configurable lip sync method for games. MIG '13 In: Proceedings of Motion on Games, pp 131–140 (2013) doi: 10.1145/2522628.2522904
9. Dai, W., Dai, C., Qu, S., Li, J., Das, S.: Very deep convolutional neural networks for raw waveforms. In: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 421–425 (2017) doi: 10.1109/ICASSP.2017.7952190
10. Aytar, Y., Vondrick, C., Torralba, A.: SoundNet: Learning sound representations from unlabeled video. Advances in Neural Information Processing Systems 29 NIPS (2016)
11. Wyse, L.: Audio spectrogram representations for processing with convolutional neural networks. In: Proceedings of the First International Workshop on Deep Learning and Music joint with IJCNN, arXiv:1706.09559 (2017) doi: 10.48550/arXiv.1706.09559
12. Kazemi, V., Sullivan, J.: One millisecond face alignment with an ensemble of regression trees. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 1867–1874 (2014)
13. Sutton, R. S., Barto, A. G.: Reinforcement learning: An introduction. Francis Bach (2018)
14. Mansar, Y.: Audio classification: A convolutional neural network approach. https://medium.com (2018)

ISSN 1870-4069

Propuesta para la detección del daño encefálico ocasionado por SARS-CoV-2 por medio de técnicas de visión computacional y Deep Learning

Mitchel A. Gomez, Edward A. Dominguez

Universidad Politécnica del Valle de México,
México

mitchell.gomez@outlook.com,
edward.dominguez.ordaz@upvm.edu.mx

Resumen. A lo largo de esta batalla contra el SARS-CoV-2 (COVID-19), se han detectado una gran cantidad de daños que genera este virus de tipo coronario, a través de diversas investigaciones, algunos médicos observaron comportamientos anormales a nivel neuronal. CoVBrainNet tiene como objetivo identificar las características que muestra el cerebro durante o después de ser infectado con el virus COVID-19, ya que el daño realizado es permanente, al igual como lo que sucede en el tejido pulmonar. Al utilizar herramientas como lo son el Deep Learning y la visión por computadora, es posible utilizar las resonancias magnéticas y analizarlas para la detección por medio de técnicas de inteligencia artificial. CoVBrainNet fue creada mediante la IDE de MATLAB 2021a, con 3 tipos de enfermedades cerebrales como lo son (*cerebros con tumores, sanos y con daño por SARS-CoV*-2) que tuvo resultados contundentes en el entrenamiento y pruebas con precisión de 92.53% en COVID-19, 100% de detección en cerebros sanos y 49.33% en detección de tumores. El entrenamiento fue realizado ocupando una Geforce RTX 3070 y utilizando una base de datos pública con el nombre de *Brain Damage due to Coronavirus*, sin embargo, existe poca información disponible por ser un área poco explorada en el análisis de daños en cerebros de pacientes que contrajeron SARS-CoV-2. Se realizaron técnicas de aumentación para incrementar el número de muestras y mejorar el desempeño de la red por lo que esta propuesta ayudara a promover la investigación sobre el daño que puede generar el COVID-19 al cerebro y agilizar su detección.

Palabras clave: SARS-CoV-2, MATLAB, NYT, CNN, YOLOv2, CoVBrainNet.

Proposal for the Detection of Brain Damage Caused by SARS-CoV-2 by Means of Computer Vision and Deep Learning Techniques.

Abstract. Along this battle versus SARS-CoV-2 (COVID-19), several damages generated by this coronary virus have been detected, through diverse investigations some doctors observed abnormal behaviors at neuronal level. CoVBrainNet aims to identify the characteristics that the brain shows during or

after being infected with the COVID-19, because the detected damage is permanent, just like what happens in lung tissue. By using tools such as Deep Learning and computer vision, it is possible to use CT scans and analyze them for detection using artificial intelligence techniques. CoVBrainNet was created using the MATLAB 2021a IDE, with 3 types of brain diseases (with tumors, healthy and with SARS-CoV-2 damage) that had convincing results in training and testing with 92.53% accuracy in COVID-19, 100% detection in healthy brains and 49.33% in tumor detection. The training was performed using a Geforce RTX 3070 and using a public database with the name Brain Damage due to Coronavirus, however, there is little information available because it is a little explored area in the analysis of damage in the brains of patients who contracted SARS-CoV-2. Augmentation techniques were used to increase the number of samples and improve the performance of the network, so this proposal will help to promote research on the damage that COVID-19 can cause to the brain and speed up the detection.

Keywords: SARS-CoV-2, MATLAB, NYT, CNN, YOLOv2, CoVBrainNet.

1. Introducción

Los virus de tipo coronario son una gran familia la cual afecta principalmente los pulmones, yendo desde un resfriado común hasta enfermedades que pueden terminar con una vida, este tipo de virus viven en varios animales y humanos, se le conoce como tipos coronarios debido a que estas poseen unas puntas que se adhieren a la superficie de cualquier objeto siendo estas orillas la proteína "S" [7], este virus que se ha declarado pandemia tiene origen en Wuhan China en diciembre de 2019.

Todas aquellas personas con algún problema cardiovascular o con alguna deficiencia pulmonar se vieron en riesgo por la forma en la que el SARS-CoV-2 ataca en su mayoría el tejido pulmonar, sin embargo, en estudios recientes [3] se detectó que el cerebro y algunas capacidades motrices se dañan, generando problemas encefálicos, además de los daños a los pulmones.

Para realizar un análisis rápido de este tipo de imágenes CoVBrainNet realiza una detección para identificar si el cerebro del paciente tiene afectaciones derivadas del COVID-19, ya que un estudio reciente indica que, a pesar de ser un índice bajo, el daño al cerebro es permanente, por lo tanto, con la técnica de Deep Learning y You Only Look Once V2 (YOLO V2) se puede encontrar aquellas anomalías en el cerebro.

2. Trabajos relacionados

Con el uso de técnicas de Deep Learning se han creado diversos sistemas para detección de daños ocasionados por COVID-19 en los pulmones mediante radiografías, no obstante, existen estudios médicos en donde se mencionan los daños en la corteza cerebral [8], y la forma de actuar del virus mediante los picos de proteína [7], sin embargo, no existe ningún precedente del uso de Deep Learning para la detección de esta enfermedad a nivel cerebral. En [9] se realizó una investigación que explica las características físicas que influyen en el cerebro cuando una persona contrae la enfermedad.

Tabla 1. Imágenes utilizadas para entrenar CoVBrainNet.

Clases	Números de imágenes
COVID	375
Sano	375
Tumor	375

Tabla 2. Imágenes utilizadas el testeo de CoVBrainNet.

Clases	Números de imágenes
COVID	375
Sano	375
Tumor	375

La gran mayoría de dichas características se observan en las investigaciones médicas que revelan los factores visibles en el cerebro y el efecto neurológico que genera el virus.

Cabe mencionar que toda la información correspondiente al daño que se genera en el cerebro es privativa y se requiere aprobaciones por parte de la comunidad médica para poder hacer uso de las bases de datos, lo cual limita la investigación y la creación de sistemas de detección mediante técnicas de inteligencia artificial.

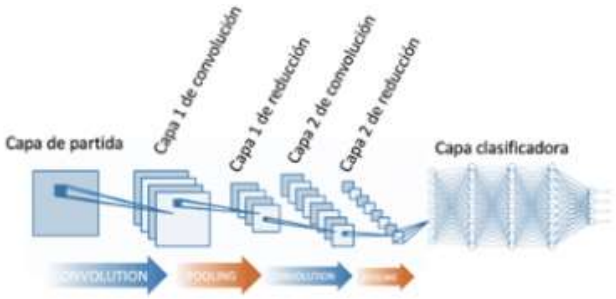
Por lo que en este proyecto se realiza una propuesta novedosa de análisis con los datos disponibles en bases de datos públicas para la detección de daño cerebral a causa del SARS-CoV-2 utilizando Deep Learning y Visión Computacional con la finalidad de agilizar los tiempos de diagnóstico y detección por parte de los especialistas en el área.

3. Metodología y materiales

Motivados por la problemática presentada anteriormente, se creó una red neuronal convolucional llamada CoVBrainNet ocupando el toolbox de Deep Learning de MATLAB 2021a en donde se consideró las dimensiones de la base de datos y la incompatibilidad con redes pre entrenadas existentes por factores de dimensión que no tenían un procesamiento optimo en el manejo de los datos. Los datos recuperados fueron cambiados del formato general (DICOM) al formato de imagen JPG, para permitir que CoVBrainNet procese los datos obtenidos de *Brain Damage due to Coronavirus* base de datos de uso común y la única fuente que brindó su apoyo para el proyecto.

3.1. Categorías y Dataset

Las categorías para clasificar son resonancias magnéticas de cerebros que cuentan con (***tumores, sanos y con daño por SARS-CoV-2***). Posteriormente se realizó una redimensión al conjunto de datos con diferentes ángulos y reflexiones con la finalidad de incrementar la precisión del sistema y evitar el overfitting.



.

Fig. 1. Estructura base de CNN's.

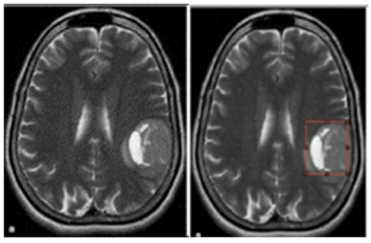


Fig. 2. Proceso de etiquetado para la red CoVBrainNet mediante la aplicación de ImageLabeler de MATLAB2021a.

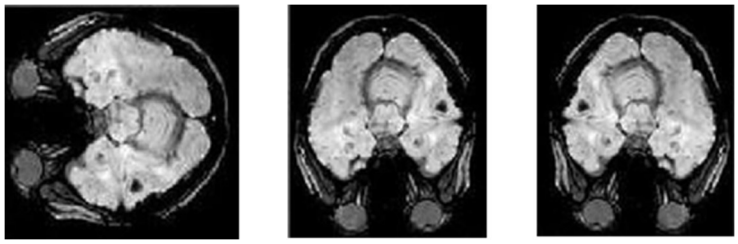


Fig. 3. Muestras de la data aumentación realizadas.

Los datos que se utilizaron para entrenar se muestran en la Tabla 1, como se mencionó anteriormente las Datasets son de dominio público como los datos del COVID-19 [8], si es sano [10, 11] y los datos acerca de tumores [11]. Para realizar las pruebas se ocuparon 1,125 imágenes (Tabla 2), que debido a la aumentación se consideran como datos desconocidos para CoVBrainNet.

3.2. Arquitectura de CoVBrainNet

Las redes neuronales convolucionales (CNN) son herramientas para realizar Deep Learning, además de que este tipo de redes son útiles para la clasificación de imágenes y detección de objetos. Para este tipo de redes se implementan una serie de capas interconectadas [12] un ejemplo de ello se observa en la Fig. 1. Por lo que CoVBrainNet fue diseñada siguiendo esas características.

Tabla 3. Capas convolucionales que compone CoVBrainNet.

Layer	Nombre	Tipos	Activaciones
1	input	Image Input	213x213x1
2	conv_1	Convolution	213x213x8
3	BN1	Batch Normalization	213x213x8
4	relu_1	ReLU	213x213x8
5	maxpool1	Max Pooling	106x106x8
6	conv_2	Convolution	106x106x16
7	BN2	Batch Normalization	106x106x16
8	relu_2	ReLU	106x106x16
9	maxpool2	Max Pooling	53x53x16
10	conv_3	Convolution	53x53x32
11	BN3	Batch Normalization	53x53x32
12	relu_3	ReLU	53x53x32
13	maxpool3	Max Pooling	26x26x32
14	conv_4	Convolution	26x26x64
15	BN4	Batch Normalization	26x26x64
16	relu_4	ReLU	26x26x64
17	maxpool4	Max Pooling	13x13x64
18	conv_5	Convolution	13x13x128
19	BN5	Batch Normalization	13x13x128
20	relu_5	ReLU	13x13x128
21	maxpool5	Max Pooling	6x6x128
22	conv_6	Convolution	6x6x256
23	BN6	Batch Normalization	6x6x256
24	relu_6	ReLU	6x6x256
25	maxpool6	Max Pooling	3x3x256
26	conv_7	Convolution	3x3x512
27	BN7	Batch Normalization	3x3x512
28	relu_7	ReLU	3x3x512
29	yolov2Conv1	Convolution	3x3x512
30	yolov2Batch1	Batch Normalization	3x3x512
31	yolov2Relu1	ReLU	3x3x512
32	yolov2Conv2	Convolution	3x3x512
33	yolov2Batch2	Batch Normalization	3x3x512
34	yolov2Relu2	ReLU	3x3x512
35	yolov2ClassConv	Convolution	3x3x24
36	yolov2Transform	YOLO v2 Transform Layer	3x3x24
37	yolov2OutputLayer	YOLO v2 Output	

Tabla 4. Precisión de las pruebas por categoría.

Clases	Números de imágenes
COVID-19	92.53%
Sano	100%
Tumor	49.33%

Tabla 5. Imágenes detectadas del testeo.

Categoría	Imágenes de Entrada	Imágenes Entrenadas
COVID-19	375	347
Sano	375	375
Tumor	375	185

Tomando como referencia la arquitectura de redes como Alexnet, ResNet18, DarkNet53, entre otras, se creó una red de etapas convolucionales con una profundidad total de 38 capas y de esta forma extraer características necesarias para posteriormente concatenarla con un arreglo de capas YOLO para detección en tiempo real mediante ROI's [13].

CoVBrainNet contribuye a la detección del problema, al realizar la clasificación y distinción de anomalías cerebrales con relación a las categorías con la cual se entrenó a la red, para ello se realizó el etiquetado de cada imagen mediante la aplicación ImageLabeler de MATLAB 2021a seleccionando el área en la que se encuentra esta característica, y colocando los ROI's correctos como se muestra en la Fig. 2.

3.3. Configuración experimental

El sistema está planeado para poder usar la configuración que se muestra en la Fig. 4 y de esta forma obtener los resultados del daño causado al cerebro por el SARS-CoV-2. Las pruebas realizadas contemplan un total de 1125 fotografías, usando la técnica de *data augmentation* modificando las imágenes obtenidas para aumentar la precisión de la red (Fig. 3).

Una vez realizadas las pruebas se determina que el sistema logra reconocer de forma adecuada los daños con COVID-19, sin embargo, genera un poco de confusión con los problemas de tumor, esto se debe a que algunos pacientes que contrajeron COVID-19 a su vez contaban con tumores en la parte cerebral, dichos datos de precisión y de detección se muestran en la Tabla 4 y Tabla 5.

3.4. Resultados

Tras las pruebas realizadas (Tabla 5) muestran la precisión y la tasa de error con respecto a los datos ingresados. Tomando los resultados obtenidos en la prueba, los cuales fueron etiquetados como se observa en Fig. 5a se identifican las zonas del cerebro considerando en comparación con el paciente que sufrió daño por COVID-19

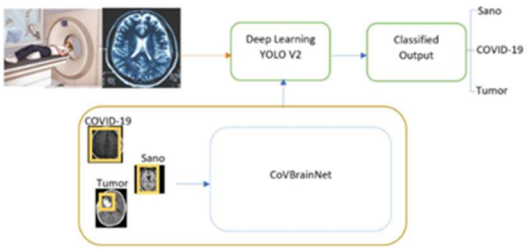
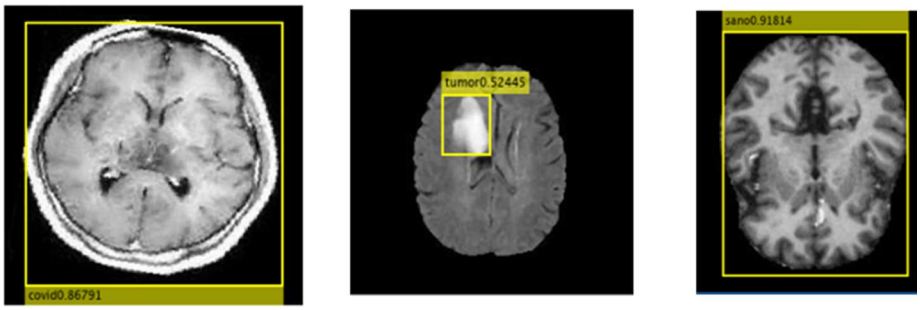


Fig. 4. Configuración para la obtención de datos de CoVBrainNet.



a) b) c)

Fig. 5. Datos detectados de CoVBrainNet.
En a) se visualiza la detección 1 de COVID-19 por medio de CoVBrainNet
En b) se visualiza la detección 1 de tumor por medio de CoVBrainNet
En c) se visualiza la detección Cerebro sano detectado por medio de CoVBrainNet

(Fig. 5b) observando un cerebro con tumores y en Fig. 5c se observa un cerebro sano, sin embargo, también se encuentran aquellos casos en donde se clasifica erróneamente.

4. Conclusiones y trabajo a futuro

En este proyecto con las herramientas de Deep Learning, y YOLO se pueden diferenciar aquellos rasgos característicos del COVID-19 con la información de los tumores en comparación con un cerebro sano, mediante resonancias magnéticas.

El proceso se ve en la capacidad de diferenciar entre un cerebro sano, si es que tuvo daño por COVID-19 o un tumor en su sistema, esta red no busca sustituir o remplazar una prueba de detección, sin embargo, puede caracterizar los datos y dar soporte para poder deducir si hubo una afectación en el cerebro.

Existe la posibilidad de incorporar este sistema a hospitales de primer contacto para la identificación temprana y tratamiento a aquellos afectados encefálicamente por el SARS-CoV-2 con el fin de evitar más daños colaterales.

El trabajo futuro se podría dar colaborativamente con todas aquellas instituciones que apoyen a aumentar la base de datos para mejorar la detección. CoVBrainNet es una alternativa con el uso de inteligencia artificial capaz de detectar los daños encefálicos por COVID-19 y de igual manera monitorear que el daño cerebral no incremente tras el contagio, con el uso de las técnicas desarrolladas a lo largo de esta investigación.

Referencias

1. The New York Times: How the coronavirus attacks the brain. https://www.nytimes.com/2020/09/09/health/coronavirus-brain.html (2020)
2. The Harvart Gazette: Specialized scanning furthers understanding of the virus' potential effects on the brain (2020) de: https://news.harvard.edu/gazette/story/2020/ 11/small-study-reveals-details-of-brain-damage-in-covid-19-patients/
3. Yeung, J., Mascarenhas, L.: La pandemia de coronavirus podría causar una ola de daño cerebral advierten científicos. CNN (2020)
4. Sinning, M.: Clasificación de los tumores cerebrales. Revista Médica Clínica Las Condes, vol. 28, no. 3, pp. 339–342 (2017) doi: 10.1016/j.rmclc.2017.05.002
5. Oxford Academic: The emerging spectrum of COVID-19 neurology: Clinical, radiological and laboratory finding (2020) https://academic.oup.com/brain/article/ 143/10/3104/5868408
6. Kandemirli, S. G., Dogan, L., Sarikaya, Z. T., Kara, S., Akinci, C., Kaya, D., Yildirim, D., Tuzuner, F., Yildirim, M. S., Ozluk, E., Gucyetmez, B., Karaarslan, E., Koyluoglu, I., Demirel-Kaya, H. S., Mammadov, O., Ozdemir, I. K., Afsar, N., Yalcinkaya, B. C., Rasimoglu, S., et al.: Brain MRI findings in patients in the intensive care unit with COVID-19 infection. RSNA, Radiology (2020) https://pubs.rsna.org/doi/10.1148/ radiol.2020201697
7. Idrees, D., Kumar, V.: SARS-CoV-2 spike protein interactions with amyloidogenic proteins: Potential clues to neurodegeneration. Organización Mundial de la Salud, Biochem Biophys Res Commun, pp. 94-98 (2021) https://search.bvsalud.org/global-literature-on-novel-coronavirus-2019-ncov/resource/en/covidwho-1157142
8. Zombori L., Bacon M., Wood, H., Chatterjee, F., Venkateswaran, R., Lampariello, S., Yoong, M.: Severe cortical associated with COVID-19 case report. https://pubmed.nc bi.nlm.nih.gov/33285362/ (2020)
9. Paterson, R.W., Brown, R. L., Benjamin, L., Nortley, R., Wiethoff, S., Bharucha, T., Jayaseelan, D. L., Kumar, G., Raftopoulos, R. E., Zambreanu, L., Vivekanandam, V., Khoo, A., Geraldes, R., Chinthapalli, K., Boyd, E., Tuzlali, H., Price, g., Christofi, G., Morrow, J., McNamara, P., et al.: The emerging spectrum of COVID-19 neurology: clinical, radiological and laboratory findings. Brain a Journal of Neurology, vol. 143, no. 10, pp. 3104-3120 (2020) doi: doi:10.1093/brain/awaa240
10. Sarvesh, D.: Alzheimer's dataset (4 class of Images). Kaggle.com (2019) https://www.kaggle.com/tourist55/alzheimers-dataset-4-class-of-images
11. Bohaju, J.: Brain tumor, extracted features for brain tumor (2020) https://www.kaggle. com/datasets/jakeshbohaju/brain-tumor
12. MathWorks: Getting started with object detection using deep learning (2020) https://la.mathworks.com/help/vision/object-detection-using-deep-learning.html
13. MathWorks: Object detection using Yolo v2O deep learning (2020) https://la.mathworks.com/help/vision/ref/yolov2objectdetector.html

ISSN 1870-4069

Cuento con realidad aumentada para fomentar la lectura

José Hernández Santiago[1,2], José Sergio Ruiz Castilla[2],
Beatriz Hernández Santiago[2]

[1] Tecnológico de Estudios Superiores de Chimalhuacán,
México

[2] Universidad Autónoma del Estado de México,
México

jose_hernandez_santiago@teschi.edu.mx
jhernandezs@uaemex.mx,betty_hsbt@hotmail.com,
jsergioruizc@gmail.com

Resumen. En 2019 el Instituto Nacional de Estadística y Geografía (INEGI) declaró que el 42.5% de la población alfabeta mexicana, con edad mayor o igual a 18 años. Los lectores que leen literatura, con el promedio de 3.3 obras leídas por persona. Actualmente hay una tendencia a migrar las publicaciones impresas hacia un medio digital. Los dispositivos digitales son más atractivos para las nuevas generaciones, pero disminuye el tiempo de atención ante las diferentes distracciones presentadas en los mismos medios como aplicaciones, multimedia e internet. En esta investigación se presenta la integración de la realidad aumentada sobre un libro infantil impreso para fomentar la lectura en edad temprana, haciendo más atractiva la actividad al incluir un medio digital. El sistema despliega en cada página animaciones de los personajes y del texto, es acompañado con audio y video de acuerdo a la escena del libro, respetando el enfoque del escritor. El libro con realidad aumentada permite atraer más lectores al fomentar la imaginación mientras se interactúa con el libro. Los resultados mostraron que los niños preferían leer con la experiencia de realidad aumentada y hubo una mejora de 89% en la comprensión del cuento.

Palabras clave: Libro con realidad aumentada, aplicación móvil, educación, fomento de lectura.

The Story with Augmented Reality for Promoting Reading

Abstract. The *Instituto Nacional de Estadística y Geografía (INEGI)* in 2019 registered 42.5% of the literate Mexican population that reads literature. Readers are around 18 years old. The average is 3.3 works per person. Currently there is a trend towards the migration of printed publications to digital media. The digital format is more attractive for the new generations but it diminishes the attention for example in multimedia or internet. In this work, we present the integration of augmented reality on a children's printed book. With the mission of motivating reading at an early age. Furthermore, reading was more attractive in a digital medium. The system showed texts and personages animated on each page. Also, it added audio and video according to the book scene. Always, rescuing the

writer's approach. The book with augmented reality can attract more readers and motivate the imagination as the reader interacts with the book. The results showed that children prefer to read with the experience of augmented reality. We finally found an 89% improvement in understanding the story.

Keywords: Book with augmented reality, mobile application, education, promotion of reading.

1. Introducción

El nivel de lectura en México, de acuerdo con estudios realizados en febrero del 2019 por el INEGI [1], muestra que 42 de 100 personas, mayores de 18 años, leyeron al menos un libro. El porcentaje de población que leyó algún material considerado por Módulo de lectura (MOLEC), paso de 84.2% en 2015 a 74.8% en 2019, con promedio de 3.3 obras por persona, siendo la falta de interés una de las razones principales, con el 21.7%. La temática más leída en los libros fue literatura con 42.5%, el 78.7% declara que comprende más de la mitad de la lectura, el 33.5% de la población declaró haber tenido fomento de la lectura gracias a que sus padres o tutores les leían.

Para los profesionistas el dominio de la lectura y comprensión de libros representa una capacidad de adaptación en un medio que requiere actualización constante, repercutiendo en mejores condiciones laborales y la posibilidad de auto superación; mientras que para la sociedad representa estar siempre actualizado sobre los hechos, tener cultura general y una actitud autocrítica.

En el ámbito educativo, la lectura de libros es la base de la enseñanza; sin embargo, es una actividad poco fomentada. Solo el 25% de la información es retenida realizando un proceso de lectura, pero puede alcanzarse un 80% si se acompaña con imágenes [2].

En años recientes la Secretaria de Educación Pública realizó un esfuerzo por dotar de *Tablets* a los alumnos de nivel básico. Los dispositivos móviles a pesar de sus beneficios, también han generado algunas desventajas como distracción y reducción del tiempo de atención en los niños.

La Realidad Aumentada es una de las tecnologías emergentes que ha tenido gran auge, gracias a los dispositivos móviles, que permiten integrar elementos virtuales en una escena en tiempo real. Esta tecnología ha tenido muchas aplicaciones en diversas áreas incluyendo en el campo de la educación, a pesar de las mejoras en cuanto a motivación mostrada por los alumnos, aun se debate si realmente beneficia la adquisición de conocimientos frente a las herramientas multimedia que ya son usadas en las clases.

Con la finalidad de colaborar con el desarrollo de software que apoye a los niños en la adquisición del hábito de lectura, en este trabajo se presenta el desarrollo de una aplicación móvil que aprovecha la motivación de los niños frente a esta nueva tecnología como la Realidad Aumentada para resaltar las historias de los cuentos impresos, haciendo más atractiva la actividad de la lectura.

El cuento infantil que se usó como base se titula "*Si yo fuera un gato*" del autor Carlos Silveyra con ilustraciones de Sonia Esplugas. La aplicación solo tiene fines didácticos y de investigación, por lo que no se considera su comercialización.

Los resultados indican que la aplicación ha permitido que los niños se acostumbren más rápido a la actividad de lectura y que el 89% prefieran esta experiencia con la guía de la Realidad Aumentada.

2. Trabajos relacionados

Se han incluido aplicaciones con Realidad Aumentada (RA) para reforzar sus actividades docentes en niveles básicos de educación, por ejemplo, en [3] usaron la RA para enseñar historia y ciencias sociales en las clases del tercer curso de primaria en España; sin embargo, no obtuvieron mejoras significativas en el aprendizaje debido a la metodología empleada, aunque si mejoró la motivación de los alumnos.

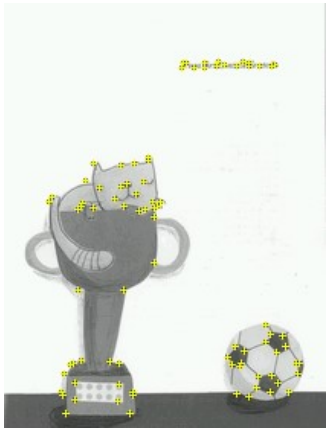
Otra investigación con alumnos de primaria y secundaria se presenta en [4], donde analizaron 21 artículos que emplearon juegos con RA para el proceso de enseñanza, donde los estudiantes mostraron beneficios en memoria a largo plazo, habilidades para resolver problemas, incremento en la motivación y mejora en las habilidades colaborativas; los estudios fueron aplicados en temas de ciencia, tecnología, ingeniería y matemáticas. De forma similar, se ofrece un análisis en [5], donde se explica que también se han detectado consecuencias negativas como "*túnel de atención*", dificultades en la usabilidad, inefectividad en la integración de las clases y diferencias en el aprendizaje.

En [6] se discute si la aplicación de la RA permitirá el acceso universal de la educación en línea, analizan nueve artículos sobre: cursos de arquitectura, botánica, anatomía, arte, inglés, entre otros. Con la conclusión, de que las disciplinas que se enfocan en entrenamiento práctico aprovechan mejor esta tecnología para provocar en los alumnos: interés, motivación, colaboración, actitud analítica de los problemas más allá de mostrar una mejora en su desempeño.

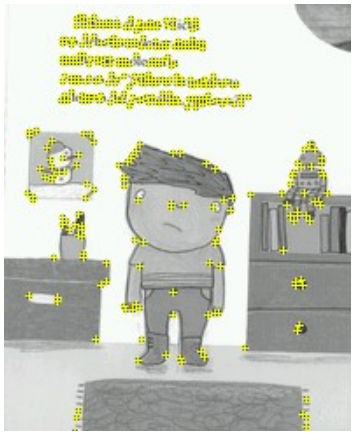
En [7] se usó un entorno con RA para entrenar a los alumnos para reducir el riesgo en el manejo de materiales, mostrando una mejora de 14.25% frente a los que aprendieron los conceptos en un salón. También los investigadores en [2], encontraron que la RA funciona bien para enseñar tareas sobre física y mecánica, ya que les permite a los alumnos experimentar con el fenómeno físico y entenderlo mediante el ajuste de los parámetros.

En esa investigación, 20 alumnos interactuaron con 12 problemas de mecánica del libro de física de nivel secundaria con RA, indicando que la aplicación es: funcional, conveniente, interesante y motivadora, aunque no hay pruebas del desempeño de los alumnos. De forma similar, en [8] se diseñó una aplicación móvil con RA para enseñar experimentos básicos de física a niños de nivel secundaria, teniendo una evaluación favorable por los profesores; sin embargo, solo lo evaluaron dos expertos y tampoco midieron si hubo mejora en el desempeño de los alumnos. Mientras que otros investigadores han desarrollado *frameworks* para permitirle a los profesores crear proyectos con RA destinados a la educación.

En [9] por ejemplo, se presenta una plataforma web para que los profesores puedan desarrollar RA usando los libros de preparatoria, permitiendo agregar imágenes, audio, video, control por *touch* y comandos de voz para que el alumno pueda realizar preguntas sobre el contenido del libro; sin embargo, no se evaluó el desempeño de los alumnos, solo reportan la aceptación de los alumnos sobre el uso de RA en los libros y

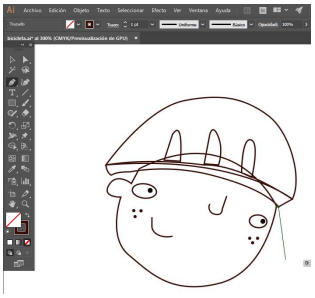


a) Image Target de mala calidad con pocos puntos característicos (marcados en amarillo).

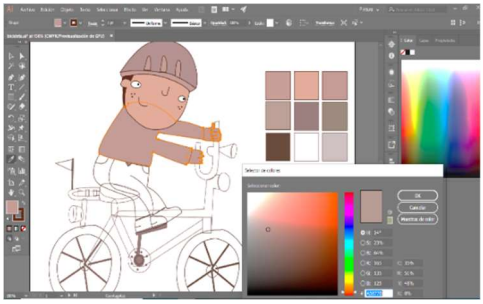


b) Image Target de buena calidad con muchos puntos característicos (marcados en amarillo).

Fig. 1. Calidad de las páginas como Image Targets.



a) Vectorización del personaje.



b) Personaje coloreado.

Fig. 2. Creación del material visual.

mencionan que uno de los mayores problemas es la creación de contenido para la RA. Un *framework* básico es presentado en [10], donde se enfocan en agilizar el proceso de creación de libros con RA por los profesores de educación primaria.

En [11] se presenta un *framework* web para el desarrollo de juegos con RA para reforzar los conceptos vistos en la clase, presentando cuestionarios aleatorios de opción múltiple, sus resultados mostraron que 80% de los alumnos aprobaron los exámenes sobre el contenido del juego.

Otro intento [12], se basa en crear objetos de aprendizaje con contenido de RA, mediante una guía propuesta para los profesores, mostrando una mejora del 20% en la efectividad en comparación con el contenido multimedia tradicional. Un análisis de 98 aplicaciones con RA se muestra en [13], donde se encontró que la mayoría usan la RA para desplegar objetos 3D y datos, repercutiendo en un bajo nivel en el proceso educativo.

En el caso de utilizar los libros con RA, en [9] se resaltan algunos problemas como el hecho de que el profesor debe incluirlo en la planeación de las clases y a veces la RA

no refleja claramente el concepto que esta textualmente en el libro; se requiere una constante actualización del contenido de las aplicaciones debido a las actualizaciones periódicas de los libros de educación pública; requiere que cada alumno cuente con su libro y dispositivo para usar la RA.

Una perspectiva sobre la respuesta emocional de los estudiantes respecto a la RA usada en el aprendizaje es presentada en [14], donde se concluye que es necesario incluir en la planeación del desarrollo de aplicaciones, las emociones positivas que se desean desarrollar para repercutir en la mejora del aprendizaje.

En [15], de forma similar encontraron que 70% de los usuarios que usaron aplicaciones con RA que incluía retroalimentación por audio e interacción, se mostraron interesados, cautivados, emocionados e impresionados; sin embargo, algunos mostraron frustración por la dependencia de las aplicaciones con el Internet, instrucciones poco claras y problemas al manipular los contenidos en 3D.

En los artículos antes citados, las estadísticas de las pruebas experimentales con los alumnos indican que la RA está siendo sobrevalorada en el campo de la enseñanza, concluyen que las herramientas tecnológicas correctamente aplicadas pueden llevar a un aprendizaje significativo, obtener una actitud positiva de los estudiantes respecto al aprendizaje, incrementar su motivación y mejorar su aprendizaje.

Abordando este último enfoque, en el que hay evidencias de la aceptación de la RA en los libros por parte de los niños, en este artículo se presenta una aplicación móvil para el sistema operativo *Android* que permite guiar al usuario en la lectura de un cuento, haciendo pausas para la lectura, mostrando animaciones y efectos de sonidos en las escenas para cada página, reforzando la actividad de lectura y haciéndola atractiva para los niños. El software propuesto puede integrarse fácilmente en el proceso de enseñanza para complementarlo, permitiendo motivar al alumno para practicar la lectura fuera del aula.

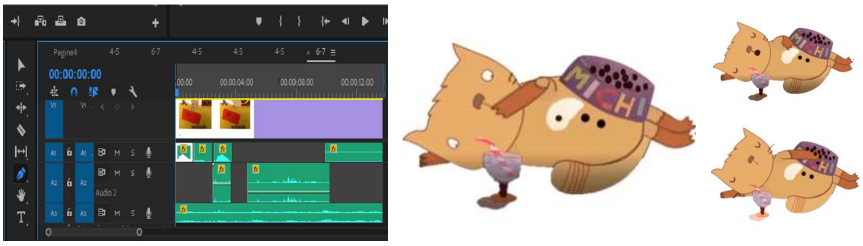
3. Preliminares

3.1. Realidad aumentada

La Realidad Aumentada (RA) tiene su origen en 1992 cuando Tom Caudell definió este término [16]. La RA combina elementos virtuales con una vista del entorno real en una misma experiencia. Para lograrlo, debe ajustar los elementos virtuales al punto de vista de la cámara para integrarlos con la escena real.

En la RA se usa el concepto de *tracking*, cuyo objetivo es seguir un patrón sin importar la escala y orientación. La tecnología base usa *Image Targets*, que consiste en un arreglo de características como representación reducida de una imagen para que el motor de RA pueda detectarla y rastrearla siempre que este al menos parcialmente en el campo de visión de la cámara sin importar la posición, escala y orientación de la imagen original.

Para lograr detectar estas características en la imagen, usa el algoritmo de Visión Artificial *Speeded Up Robust Features (SURF)* que surgió en 2006 como mejora del algoritmo *Scale Invariant Feature Transform (SIFT).* Las variantes actuales permiten definir botones virtuales sobre el *Image Target* para interactuar con el usuario, usar imágenes de diferentes caras para definir *Targets* de tipo prisma y cilindro; al incluir



a) Edición del audio. b) Edición de la animación.

Fig. 3. Edición.

a) Proyecto en Unity-Vuforia. b) Compilación para el S.O. Android.

Fig. 4. Calidad de las páginas como Image Targets.

más vistas del objeto de referencia se pueden crear *Object Targets*, que permiten usar un objeto real en lugar de una imagen plana. Actualmente hay una tendencia a dejar de usar *Image Targets* y en su lugar detectar una superficie plana como el piso para usarlo de referencia y colocar el contenido de la RA.

Existen diversas plataformas para desarrollar RA, las más usadas son la combinación de *Vuforia* con el *IDE Unity* para generar aplicaciones destinadas al Sistema Operativo *Android*.

4. Metodología

4.1. Creación y evaluación de los "*Image Targets*"

De acuerdo a los diferentes tipos de *targets* disponibles para crear RA, se seleccionaron los *Image Targets* ya que concuerdan con el proyecto puesto que el cuento incluye muchas ilustraciones.

El primer paso del método propuesto consiste en evaluar si las imágenes del libro pueden ser usadas como referencia para crear *Image Targets* que servirán como base para ubicar la zona donde se debe colocar el contenido multimedia y ajustarlo a su posición. Debe tenerse en cuenta que una página con colores uniformes, con pocos

a) Portada. b) Contraportada.

Fig. 5. Realidad Aumentada con animación sobre las portada y contraportada.

Fig. 6. Realidad Aumentada con imágenes sin fondo sobre las páginas del cuento.

detalles y muchas áreas vacías, como la Figura 1.a, provocará que el algoritmo *SURF* tenga poca precisión al seguir su ubicación en el espacio, por lo que se prefieren páginas con texto, ilustraciones con alto contraste y detalles que generen una gran cantidad de puntos característicos como se muestra en la Figura 1.b.

Un problema que puede surgir en la ejecución es que el brillo causado por el material de las hojas interfiera con el seguimiento o *traking* de los *Image Targets*, por lo que se debe preferir texturas tipo mate.

4.2. Creación del material audiovisual basado en el libro

El siguiente paso es vectorizar los personajes e ilustraciones de cada página para digitalizarlos, como se muestra en la Figura 2.a y poder animarlos posteriormente. En la Figura 2.b se puede observar que se trató de igualar los colores para respetar la identidad del libro.

El siguiente paso fue agregar los efectos de sonido (Figura 3.a) y animar las escenas (Figura 3.b). Se incluyen animaciones y audios para todas las páginas del libro, donde los personajes realizaban acciones de acuerdo a la imagen o texto original del autor; por ejemplo, en la Figura 3.b se puede ver que en la animación del personaje “Michi”, este come, bebe y deja caer algunas de las croquetas.

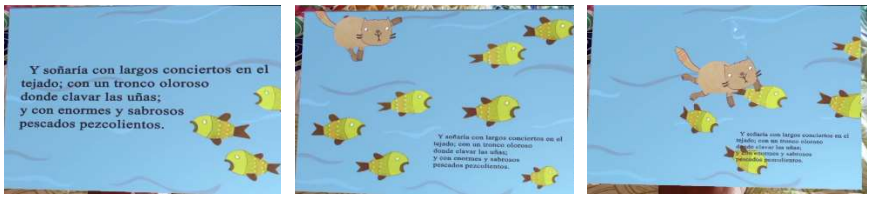


Fig. 7. Realidad Aumentada con animación sobre las páginas del cuento.

a) b) c)

Fig. 8. Realidad Aumentada con efectos de sonido sobre las ilustraciones del cuento.

4.3. Integración del material audiovisual sobre los "*Image Targets*"

Para integrar el material con los *Image Targets*, se empleó el software *Unity* con la librería *Vuforia*, configurando los permisos para usar la cámara del celular. En la Figura 4.a se muestra una vista previa del diseño terminado y en la Figura 4.b la compilación de la aplicación para el Sistema Operativo *Android* versión 4.0 o superior, requiriendo 157 Megabytes para ser instalado en el celular debido a que se incluyeron 43 *Image Targets* con sus respectivas animaciones y sonidos.

En la Figura 5.a se puede ver las pruebas realizadas en la portada, donde se anima la aparición del texto, la luna, las estrellas y la transformación del niño en gato; mientras que en la Figura 5.b se muestra la animación para el gato, la presentación de la foto del autor, así como un video de la entrevista y presentación del cuento.

Estas animaciones tienen como objetivo atraer a los lectores infantiles y que los padres también se interesen por la obra del autor. Otros ejemplos del uso de la aplicación son mostrados en la Figura 6, donde las imágenes de los personajes son resaltados y presentados en vertical usando imágenes con formato PNG sin fondo.

Un ejemplo de las animaciones incluidas en el libro, puede verse en la transición en la Figura 7, donde el personaje sueña con ser un gato y nadar para atrapar los peces. En esta escena el texto aparece gradualmente al principio, estando sincronizado con la

Tabla 1. Resultados de las pruebas con niños.

Pregunta	Lectura con el libro impreso	Lectura con el libro usando Realidad Aumentada
Le interesa la lectura	44%	100%
Comprende la historia	56%	89%
Le interesa el cuento	72%	91%
Pediría el cuento como regalo	71%	100%

Tabla 2. Resultados de las pruebas con adultos.

Pregunta	Porcentaje
Conoce los libros con RA	15%
Le interesa el cuento con RA	95%
Fácil de usar la app	80%
Regalaría el libro con RA	100%

animación para que se pueda leer y reflejar la acción. Las animaciones se ajustan a la posición del libro gracias al *tracking* de los *Image Targets* realizada por la Realidad Aumentada, de esta forma el lector percibe que los personajes del cuento realmente se están moviendo sobre la página.

Se crearon animaciones y sonido para que la mayoría de las páginas resultaran atractivas para los niños, algunos ejemplos representativos se muestran en la Figura 8.a, donde la escena muestra la animación del viento soplando las hojas de otoño y un balón rebotando; en la Figura 8.b se muestra la animación de la banda de pulgas tocando una canción a los oídos del gato; en la Figura 8.c se muestran escenas donde hay sonidos característicos como el ladrido del perro "*sultan*", personajes representando oficios y la celebración al ganar un partido de *football*.

5. Resultados

Las pruebas se realizaron con niños y adultos que asistieron al Planetario Digital en Chimalhuacán, Estado de México. En la primera prueba se encuestaron 27 niños de edades entre 7 y 11 años, quienes leyeron el cuento original de forma impresa y después el cuento con Realidad Aumentada.

El contraste puede notarse en la Figura 9, donde 72% indicó interesarles el cuento impreso debido a las ilustraciones, mostrando un mayor interés de 91% cuando usaron la aplicación con RA y notaron las animaciones; en cuanto a la comprensión de la historia, inicialmente solo el 56% logró identificar personajes e historia en general, subiendo a 89% con el uso de la RA; el 71% pediría el cuento impreso como regalo, pero al leer el cuento con RA, el 100% contestó que preferiría que le regalaran un cuento con RA; finalmente el interés que indicaron tener sobre la lectura paso de 44% a 100% con RA. Ver la Tabla 1.

La segunda prueba se realizó con 23 adultos, familiares de los niños anteriormente encuestados, los resultados se muestran en la Figura 10, de los cuales solo 15% sabían que los libros pueden tener Realidad Aumentada, 95% mostró interés por el cuento con RA, a 80% les pareció fácil de usar la app y 20% tuvo dificultades al no estar cómodo

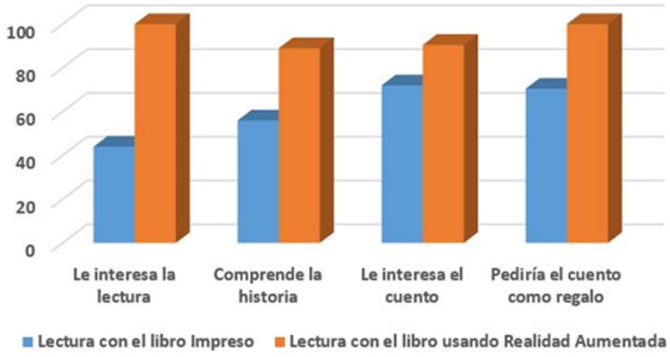


Fig. 9. Resultados de las encuestas con niños.

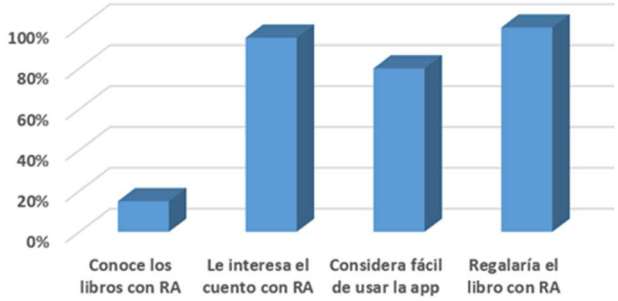


Fig. 10. Resultados de las encuestas con adultos.

con el uso de los *Smartphones*; finalmente el 100% indicó que le regalaría un libro con RA a su hijo para fomentar su hábito de lectura. Ver la Tabla 2.

6. Conclusión

La lectura tradicional requiere libros impresos cuyo costo limita su acceso, incluso con los medios digitales que facilita el acceso a libros gratuitos, solo el 42.5% de la población alfabeta mexicana, con edad mayor o igual a 18 años, lee literatura. En el caso del fomento de la lectura en edades tempranas, de acuerdo al INEGI, solo el 33.5% tuvieron la experiencia de que sus padres o tutores les leyeran.

La inclusión de una aplicación móvil capaz de mostrar animaciones y audios con realidad aumentada sobre las páginas de un cuento infantil permitirá enriquecer el proceso de aprendizaje y fomentar el hábito de lectura.

Esta forma de guiar al lector es preferida por los niños encuestados, llegando a un 89% el nivel de comprensión de la lectura del cuento como lo reflejaron los resultados. En esta versión se usaron audios y animaciones 2D debido a que el estilo de las ilustraciones podría cambiar si se convertían a modelados 3D. Todas las páginas del cuento tienen Realidad Aumentada, incluyendo la portada y contraportada; sin embargo, algunas páginas no tenían los suficientes puntos característicos para crear un

Image Target de calidad, afectando la estabilidad del posicionamiento de las animaciones, esto se resolvió al incluir algunas etiquetas sobre las hojas para incrementar la cantidad de puntos característicos en las zonas vacías y mejorar la precisión en el proceso de reconocimiento de la página por lo que se recomienda trabajar con el diseño del libro antes de la impresión por la editorial.

Otro problema es que la aplicación solo funciona con una versión del libro, ya que, debido a las modificaciones en las nuevas reimpresiones, estas evitan que se reconozca algunas páginas. La aplicación de tecnología actual es una ventaja que los desarrolladores de software educativo pueden seguir aprovechando para proveer a los alumnos de aplicaciones de alta usabilidad, intuitivas, que aumente su interés y mantengan su motivación por aprender.

Aún hay trabajos que realizar, por ejemplo, la evaluación automática de la lectura correcta de los usuarios mediante reconocimiento de voz y el seguimiento del hábito de lectura para que los profesores puedan revisar los avances.

Agradecimientos. Un especial agradecimiento al Tecnológico de Estudios Superiores de Chimalhuacán, institución que nos ha apoyado y que fomenta este tipo de investigación. Se agradece también a las Ingenieras en Animación Digital y Efectos Visuales, Hernández Martínez Blanca Estela y Reséndiz Flores Madai, cuyo trabajo de tesis permitió darle seguimiento a esta investigación.

Referencias

1. Instituto Nacional de Estadística y Geografía. Población lectora en México con tendencia decreciente en los últimos cinco años [Online], (2019) https://www.inegi.org.mx/contenidos/saladeprensa/boletines/2019/EstSociodemo/MOLEC2018_04.pdf
2. Daineko, Y., Ipalakova, M., Tsoy, D., Shaipiten, A., Bolatov, Z., Chinibayeva, T.: Development of practical tasks in physics with elements of augmented reality for secondary educational institutions. Augmented Reality, Virtual Reality, and Computer Graphics, 10850, pp. 404–412 (2018) doi: 10.1007/978-3-319-95270-3_34
3. Piqueras, E. M., Cózar, R., González-Calero, J. A.: Incidencia de la realidad aumentada en la enseñanza de la historia. Una experiencia en tercer curso de educación primaria. Enseñanza & Teaching, vol. 36, no. 1, pp. 23–39 (2018) doi: 10.14201/et20183612339
4. Pellas, N., Fotaris, P., Kazanidis, I., Wells, D.: Augmenting the learning experience in primary and secondary school education: A systematic review of recent trends in augmented reality game-based learning. Virtual Reality, vol. 23, pp. 329–346 (2018) doi: 10.1007/s10055-018-0347-2
5. Papanastasiou, G., Drigas, A., Skianis, C., Lytras, M., Papanastasiou, E.: Virtual and augmented reality efects on K 12, higher and tertiary education students' twenty frst century skills. Virtual Reality, vol. 23, pp. 425–436 (2019) doi: /10.1007/s10055-018-0363-2
6. Tsai, C.W.: The applications of augmented reality for universal access in online. Universal Access in the Information Society, vol. 18, pp. 217–219 (2019) doi: 10.1007/s10209-017-0589-x
7. Guo, W.: Improving engineering education using augmented reality environment. Learning and Collaboration Technologies. Design, Development and Technological Innovation, vol. 10924, pp. 233–242 (2018)
8. Abu, J. A., Gopalan, V., Zulkifli, A. N., Alwi, A.: Design and development of mobile augmented reality for physics experiment. User science and engineering. In: Abdullah, N.,

Wan Adnan, W., Foth, M. (eds) User Science and Engineering, i-USEr 2018, Communications in Computer and Information Science, vol. 886, pp. 47–58 (2018) doi: 10.1007/978-981-13-1628-9_5

9. Lytridis, C., Tsinakos, A.: Evaluation of the ARTutor augmented reality educational platform in tertiary education. Smart Learning Environments, vol. 5, pp. 1–15 (2018) doi: 10.1186/s40561-018-0058-x
10. Ali, L., Ullah, S.: A new framework for easy and efficient augmentation of primary level books. Augmented Reality, Virtual Reality, and Computer Graphics, vol. 10850, pp. 311–321 (2018) doi: 10.1007/978-3-319-95270-3_26
11. Ierache, J., Mangiarua, N. A., Becerra, M. E., Igarza, S.: Framework for the development of augmented reality applications applied to education games. Augmented Reality, Virtual Reality, and Computer Graphics, vol. 10850, pp. 340–350 (2018) doi: 10.1007/978-3-319-95270-3_28
12. Guimarães, M. P., Alves, B. C., Durelli, R. S., Guimarães, R. F. R., Dias, D. C.: An approach to developing learning objects with augmented reality content. Computational Science and Its Applications ICCSA 2018, vol. 10963, pp. 757–774 (2018) doi: 10.1007/978-3-319-95171-3_59
13. O'Shea, P. M., Elliott, J. B.: Augmented reality in education: an exploration and analysis of currently available educational apps. Immersive Learning Research Network, vol. 621, pp. 147–159 (2016) doi: 10.1007/978-3-319-41769-1_12
14. Khairuddin, AN. A., Redzuan, F., Daud, N. A.: Evaluating students' emotional response in augmented reality-based mobile learning using kansei engineering. User Science and Engineering, vol. 886, pp. 79–89 (2018) doi: 10.1007/978-981-13-1628-9_8
15. Irshad, S., Awang, D. R., Muhamad, N. I. A., Mohd, S. R., Omar, Y.: Measuring user experience of mobile augmented reality systems through non-instrumental quality attributes. User Science and Engineering, vol. 886, pp. 349–357 (2018) doi: 10.1007/978-981-13-1628-9_3
16. Caudell, T. P., Mizell, D. W.: Augmented reality: an application of heads-up display technology to manual manufacturing processes. In: Proceedings of the Twenty-Fifth Hawaii International Conference on System Sciences, vol. 2, pp. 659–669 (1992)

ISSN 1870-4069

Un método no invasivo para la clasificación de manzanas

Juan Carlos Olguín-Rojas[1,3], J. Irving Vasquez-Gomez[1,2], Gilberto de Jesus Lopez-Canteñs[3], Juan Carlos Herrera-Lozada[1]

[1] Instituto Politécnico Nacional,
Centro de Innovación y Desarrollo Tecnológico en Cómputo,
México

[2] Consejo Nacional de Ciencia y Tecnología,
Ciudad de México,
México

[3] Universidad Autónoma Chapingo,
Departamento de Ingeniería Mecánica Agrícola,
México

olguinr.juancarlos@gmail.com

Resumen. Actualmente, en muchas empresas agroindustriales de México, la clasificación de manzanas se hace de forma manual lo que en ocasiones genera deficiencias en la calidad del producto. Estos problemas podrían resolverse o reducirse con la implementación de equipos de visión en sitio equipados con algoritmos inteligentes. En este trabajo presentamos como objeto de estudio la clasificación de manzanas Red Delicious, Granny Smith, Golden Delicious y Gala, mediante algoritmos de inteligencia artificial. Comparamos la eficiencia de dos arquitecturas, la primera es una red neuronal convolucional (CNN) Lenet5, y la segunda es la red convolucional VGG16 a la cual se le aplicaron técnicas de transferencia de aprendizaje de entrenamientos previos con Imagenet que contiene 1.4 millones de imágenes con 1,000 clases diferentes, presentamos para nuestros experimentos un conjunto de datos de cuatro categorías de manzanas que en total suman 2,400 imágenes con dimensión de 800x600 píxeles cada una, el 70 % de las imágenes de manzanas fueron para entrenamiento, el 15 % para validación y 15 % para prueba. La red que mejor resultados entrego fue VGG16 con la técnica de entonación fina (fine-tuning) se reentrenaron 8,130,564 parámetros de un total de 15,765,828 y su precisión fue de 99 %.

Palabras clave: Clasificación, transferencia de aprendizaje, manzanas.

A Non-Invasive Method for Apple Classification

Abstract. Currently, in many agribusiness in Mexico, the classification of apples is done manually, which sometimes generates deficiencies in the quality of the product. These problems could be solved or reduced with the implementation of on-site vision equipment based on intelligent algorithms.

In this work, we present as an object of study the classification of Red Delicious, Granny Smith, Golden Delicious and Gala apples, using artificial intelligence algorithms. We compared the efficiency of two architectures, the first is a Lenet5 convolutional neural network (CNN), and the second is the CNN VGG16 network. Both networks were set using transfer learning techniques from previous trainings with Imagenet were applied that contains 1.4 million images with 1000 different classes. We present for our experiments a set of four categories of apples that in total add up to 2,400 images with a dimension of 800× 600 pixels each, 70% of the images of apples were for training, 15% for validation and 15% for testing. The network that gave the best results was VGG16 with fine-tuning, 8,130,564 parameters were retrained out of a total of 15,765,828 and its precision was 99%.

Keywords: Classification, transfer learning, apples.

1. Introducción

En México la manzana se produce en más de 10 entidades federativas, siendo el estado de Chihuahua el principal productor de la zona noreste con aproximadamente 569,000 toneladas al año [9]. Dentro del proceso de la cosecha y postcosecha de manzanas, la correcta clasificación es fundamental ya que los frutos son catalogados por su grado de maduración y los precios de mercado están determinados por dichas inspecciones [4].

Actualmente, existen problemas en el campo mexicano referidos con elementos de mediciones convencionales basados en técnicas físico-químicas para determinar el grado de madurez o de inocuidad de un fruto, principalmente porque estas técnicas requieren instrumental de laboratorio; esto ocasiona que en muchos casos los productores nacionales implementen líneas de inspección visual en donde personas son entrenadas para identificar: clase, grado de madurez, texturas y tamaños de frutos principalmente, sin embargo estos métodos son subjetivos y dificultan la inspección de grandes lotes del fruto o análisis en masa.

Una mala clasificación en postcosecha de la manzana podría evitar cumplir con los estándares de calidad internacional. México presenta en la actualidad un gran déficit comercial en la exportación de manzanas, pues en 2018 se importaron 282,756 toneladas con un valor de 264 millones de dólares y se exportaron sólo 766 toneladas con un valor de 1.1 millones de dólares, además, la variación en porcentaje de las exportaciones del fruto en 2018 redujo 17.7 % con respecto a las exportaciones de 2017 [9].

Nuestro objetivo en este trabajo es desarrollar un sistema de visión en sitio para comparar algunas arquitecturas de aprendizaje profundo y evaluar su rendimiento en clasificación, con la finalidad de clasificar las variedades de manzana: Red Delicious, Granny Smith, Golden Delicious y Gala; que permitan como consecuencia un mejor manejo del fruto en postcosecha o incluso posiblemente en puntos de venta.

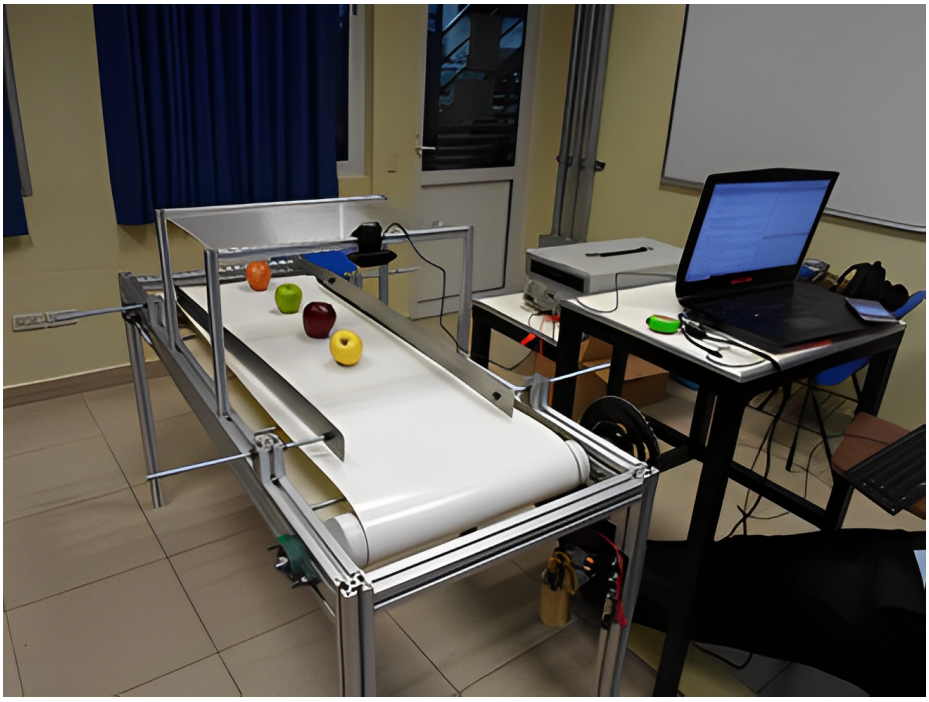


Fig. 1. Banda transportadora para captura de imágenes.

2. Trabajo relacionado

El trabajo de Mohammadi en [1] informa que desarrolló un sistema electrónico que puede clasificar las frutas en masa sobre una banda transportadora con un gramo de precisión, analiza los requisitos y desarrollos de hardware y software para sistemas de visión artificial, con énfasis en imágenes monocromas, imágenes en color e imágenes multiespectrales, además, su desarrollo clasifica por color, por hematomas, forma y densidad; informa que su clasificador permite la detección de enfermedades, defectos y contaminación en frutas y verduras.

Kondo y Ting en [5] presentan un estudio del estado del arte para describir la configuración básica de la adquisición de datos, incluidos el color, la masa y el tamaño. En este trabajo los autores consideran que los componentes de una concepción simple deberían contener un software confiable para reenviar el producto a los canales adecuados de clasificación, y reportan que existen equipos que pueden clasificar hasta 10 frutas por segundo.

Los instrumentos de clasificación mejoran los tiempos de clasificación e inspección sanitaria. Bhargava en [2] afirma que la clasificación simultánea de frutas por tamaño y color ahorraría el tiempo de inspección sanitaria, reduciendo significativamente el manejo de la fruta. En el trabajo de Sun [12] se considera que un sistema de clasificación automática de manzanas debe involucrar los aspectos de: color, peso, dimensión y defectos.

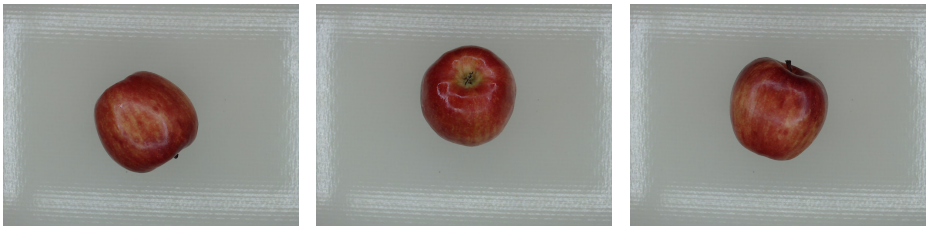


Fig. 2. Ejemplos del conjunto Gala.

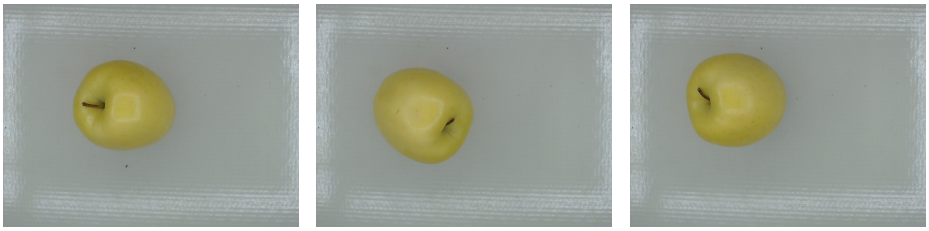


Fig. 3. Ejemplos del conjunto Golden Delicious.

El autor destaca que el brillo superficial de una manzana refleja la frescura y sus defectos, considerando a esta característica sensorial entre las más importantes [12], en la literatura especializada también se reporta el trabajo de Sofu Er [11], donde se reporta que encontraron que la característica que más afecta la calidad de la clasificación de la manzana es la mancha y la descomposición.

El trabajo de Duran Melo en [3] reporta que lograron clasificar manzanas tomando en cuenta características de su superficie, basándose en un preprocesamiento y segmentación de la imagen para extraer las características de interés para la clasificación. Reporta que, extrajeron 14 características de imágenes de manzanas Gala, tales como: niveles de rojo y verde, contornos internos en la manzana, perímetro del contorno principal, y posteriormente entrenaron una red Perceptrón Multicapa (MLP) para clasificar las manzanas en tres categorías con una exactitud del 88.33 %.

El trabajo de Moallem [8], es un buen referente en cuanto a técnicas de clasificación de manzanas, pues reportan que comparan diferentes modelos de aprendizaje automático, extrajeron características estadísticas, texturales y geométricas de imágenes de manzanas Golden Delicious, y usaron clasificadores tipo SVM (Maquinas de Soporte), MLP (Perceptrón Multicapa) y KNN (K-Vecinos Cercanos) las clasificaron las manzanas en saludables y desertadas, con una precisión del 92.5 %.

El trabajo de Valdez en [13] informa que utilizó dos arquitecturas de aprendizaje profundo para clasificar y detectar daños en manzanas, una de las arquitecturas fue YOLOV3 y la otra fue un método alternativo de detección de simple disparo (SSD) propuesto por Liu et al. [7].

Fig. 4. Ejemplos del conjunto Granny Smith.

Fig. 5. Ejemplos del conjunto Red Delicious.

3. Clasificación de manzanas

Para lograr la clasificación de manzanas con métodos no invasivos, en este trabajo se propone utilizar un sistema de visión en sitio, comparando la eficiencia de dos arquitecturas de redes neuronales convolucionales; la primera es la red Lenet5 que está basada en la arquitectura propuesta por Yann LeCun et al. [6] y la segunda es la red VGG16 [10] en la cual se utilizan técnicas de transferencia de aprendizaje, específicamente, extracción de características (feature-extraction) y entonación fina (fine-tuning).

Para la transferencia de aprendizaje se utilizó la arquitectura VGG16 que fue entrenada con Imagenet y que contiene 1.4 millones de imágenes con 1000 clases diferentes. Los entrenamientos se realizaron utilizando la plataforma Google Colab [14] en Python 3, con librerías de Keras y tensorflow versión 2.4.1.

3.1. Conjunto de datos

Una de las tareas fundamentales para el desarrollo de este trabajo, fue la creación de la base de datos de imágenes de manzanas de las categorías de nuestro interés, para lograr esta importante tarea, se diseñó y construyó una banda transportadora que tiene iluminación controlada y materiales adecuados para el manejo de alimentos, ya que son de fácil limpieza y no guardan microorganismos. Veáse Fig. 1.

El conjunto de datos está compuesto por cuatro categorías de manzanas que en total suman 2,400 imágenes con dimensión de 800×600 píxeles cada una, el 70 % de las imágenes de manzanas se utilizaron para entrenamiento, el 15 % para validación y 15 % para prueba, los conjuntos están organizados de la siguiente manera: 600 imágenes de manzana Gala (ver Fig. 2), 600 imágenes de manzana Golden Delicious (ver Fig. 3), 600 imágenes de manzana Granny Smith (ver Fig. 4) y 600 imágenes de manzana Red Delicious (ver Fig. 5).

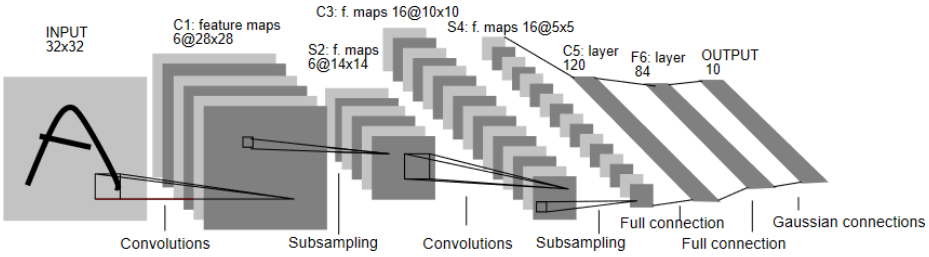


Fig. 6. Muestra los bloques que componen a la arquitectura Lenet5.

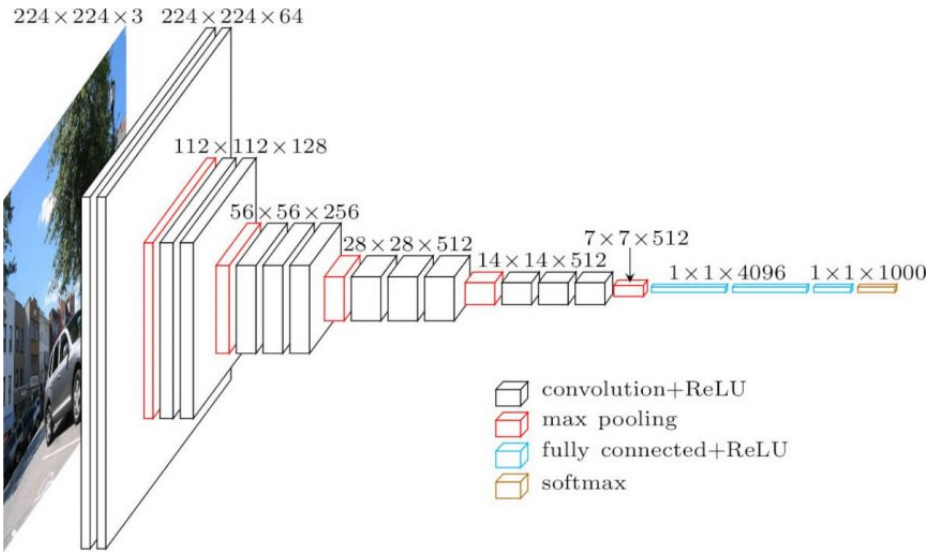
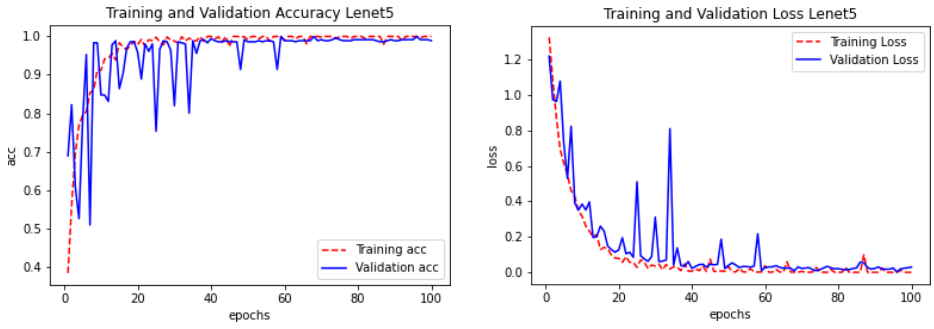


Fig. 7. Muestra la arquitectura VGG16 que participó en el ILSVRC-2014.

3.2. Arquitectura de aprendizaje profundo LeNet5 y VGG16

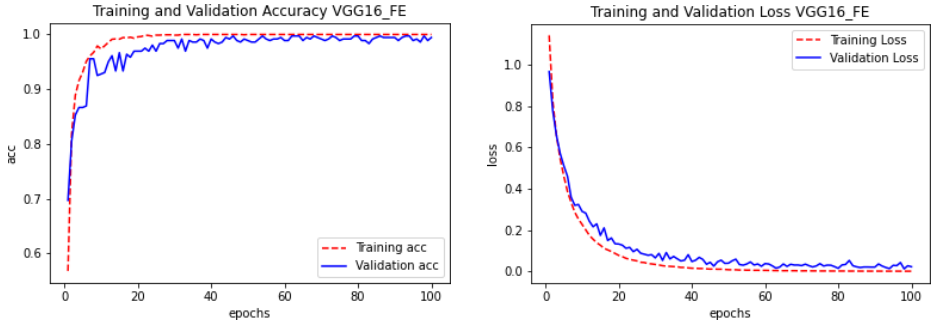
La arquitectura de red neuronal convolucional LeNet5 de Yann Lecun [6]. Se entrenó específicamente con el conjunto de datos MNIST para el problema de reconocimiento de caracteres a mano, logrando una precisión de clasificación de aproximadamente 99.2 %. Esta arquitectura consta de 2 capas convolucionales, 2 capas de agrupación máxima (max pooling) y 3 capas completamente conectadas, véase Fig. 6. Por otra parte, la red neuronal convolucional VGG16 fue propuesta por K. Simonyan y A. Zisserman [10].

Esta arquitectura participó en el ILSVRC-2014 (ImageNet Large-Scale Visual Recognition Challenge 2014) y alcanzó una precisión del 92.7 % quedando entre los 5 primeros en ImageNet, que es un conjunto de datos de más de 14 millones de imágenes que pertenecen a 1000 clases. La arquitectura VGG16 tiene cinco bloques convolucionales, cada bloque convolucional es seguido por una capa de agrupación y finalmente se tienen tres capas densas.



(a) Exactitud (b) Pérdida

Fig. 8. Resultados de la exactitud y la pédida usando la arquitectura LeNet5.



(a) Exactitud (b) Pérdida

Fig. 9. Resultados de la exactitud y la pédida usando la arquitectura VGG16 con extracción de características.

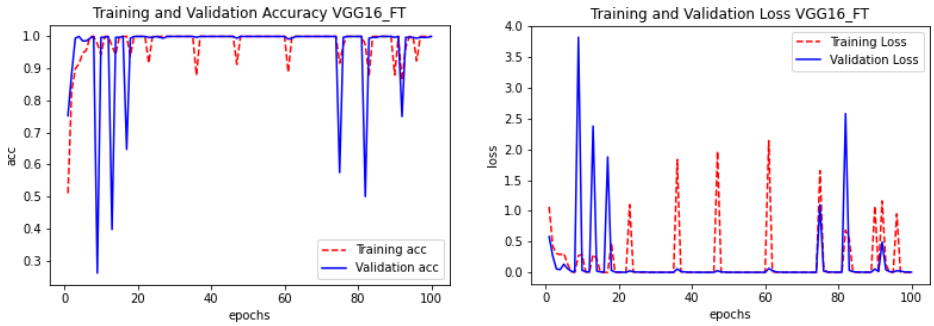
La arquitectura VGG16 que participó en el ILSVRC-2014 utilizó imágenes RGB de tamaño fijo de 224 x 224 pixeles. La imagen pasa a través de las capas convolucionales y los filtros que se usaron son de tamaño 3×3, en el caso de la agrupación máxima (max pooling) el tamaño es 2x2, véase Fig. 7.

4. Experimentación

En esta sección se describen los resultados de los entrenamientos de las arquitecturas utilizadas, la comparación de los resultados de Accuracy (exactitud) de las tres técnicas se muestra en la Fig. 11. A continuación, se presentan por separado cada una de las técnicas utilizadas.

4.1. Arquitectura de aprendizaje profundo Lenet5

Para este experimento se utiliza la arquitectura Lenet5, a la cual se le presenta el conjunto de imágenes reescaladas a 64×64 píxeles cada una. Los resultados del entrenamiento de la red Lenet5 en 100 épocas se muestran en Fig. 8.



(a) Exactitud (b) Pérdida

Fig. 10. Resultados de la exactitud y la pédida usando la arquitectura VGG16 con ajuste fino.

4.2. Transferencia de aprendizaje con extracción de características (feature extraction) en VGG16

Para este experimento se utiliza la arquitectura VGG16 que fue entrenada con Imagenet y que contiene 1.4 millones de imágenes con 1000 clases diferentes, se usa transferencia de aprendizaje con la técnica de extracción de características (feature extraction) para presentar el conjunto de imágenes de manzanas, esto significa, que sólo se reentrenó la parte final de la arquitectura. Los resultados del entrenamiento de la red VGG16 con el método de extracción de características, entrenando 100 épocas se muestran en Fig. 9.

4.3. Transferencia de aprendizaje con entonación fina (fine tuning) en VGG16

Para este experimento se continúa utilizando la arquitectura VGG16 que fue entrenada con Imagenet y que contiene 1.4 millones de imágenes con 1000 clases diferentes, se usa transferencia de aprendizaje con la técnica de entonación fina (fine tuning) para presentar el conjunto de imágenes de manzanas, esto significa que se reentrenó el bloque 5 de la arquitectura interna de VGG16 además de la parte final de la arquitectura. Los resultados del entrenamiento de la red VGG16 con el método de extracción de características, entrenando 100 épocas se muestran en Fig. 10.

4.4. Prueba del rendimiento en clasificación de las arquitecturas

Para evaluar el rendimiento en clasificación se usa la matriz de confusión, el conjunto de la prueba de validación está compuesto por un total de 360 manzanas de las 4 clases, es decir, 90 manzanas por clase. Los resultados de la matriz de confusión de la arquitectura Lenet5 se presentan en la Fig. 12(a).

Por otra parte, los resultados de la matriz de confusión de la arquitectura VGG16 con la técnica de extracción de características (FE) se presentan en la Fig. 12(b). Finalmente, los resultados de la matriz de confusión de la arquitectura VGG16 con la técnica de entonación fina (FT) se presentan en la Fig.12(c).

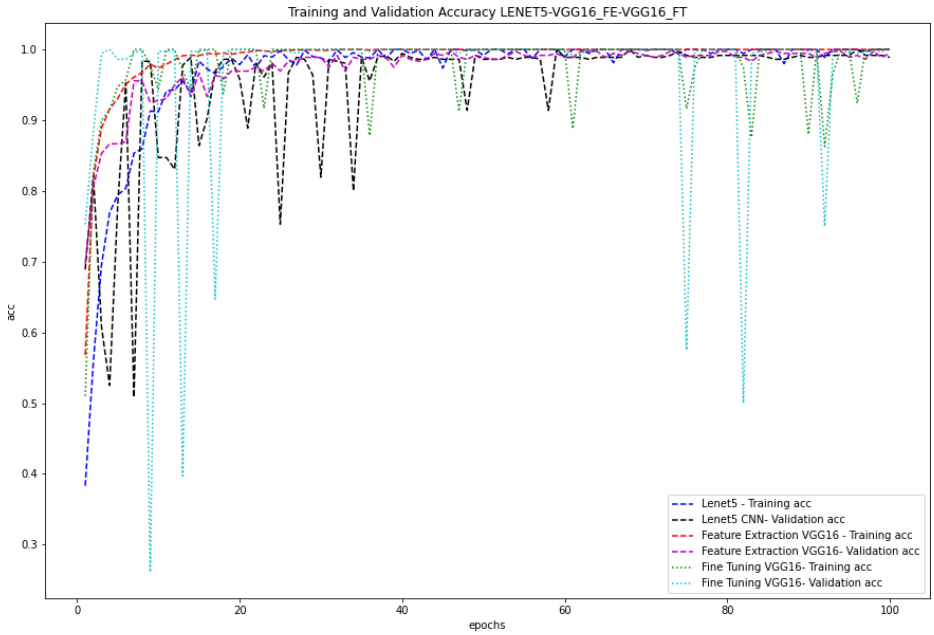


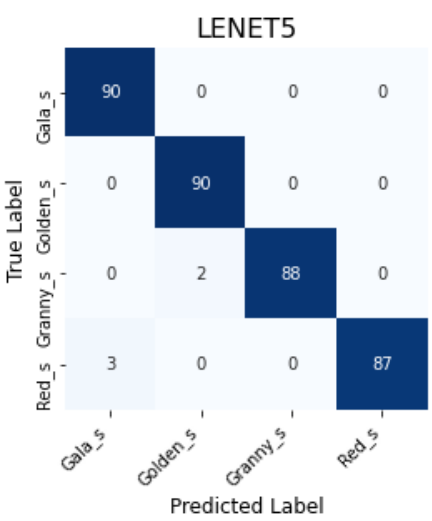
Fig. 11. Muestra los resultados de Accuracy (exactitud) en entrenamiento y en validación de las tres técnicas.

4.5. Análisis de resultados

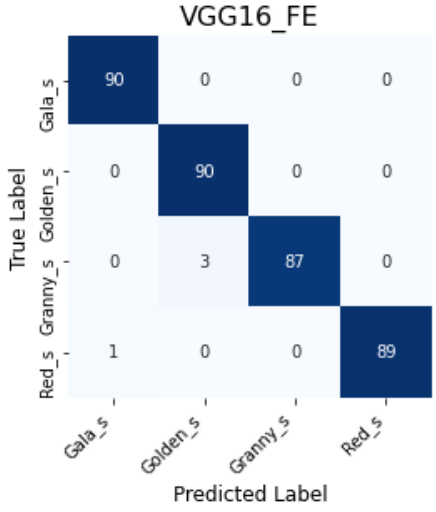
La Fig. 18, muestra que la arquitectura VGG16 con la técnica de entonación fina (fine-tuning) tiene el mejor resultado para el rendimiento en clasificación, en la matriz de confusión se obtuvo una exactitud del 99 %.

Se considera que este resultado es debido a que con esta técnica se hace un ajuste más fino y también se entrenan algunas de las capas finales de la base convolucional que se utiliza para la extracción de características, esto implica que con esta técnica se está realizando una entonación fina de las representaciones más abstractas del modelo que se está utilizando como base, esto porque las primeras capas aprenden características generales y gradualmente las capas sucesivas aprenden características más abstractas, por esta razón con esta técnica no se utiliza toda la base convolucional del entrenamiento previo, ya que, si el nuevo conjunto de datos (manzanas) difiere del conjunto de datos con el que se entrenó originalmente (Imagenet) la técnica solo utiliza las primeras capas, las cuales son congeladas para no ser entrenadas (freeze layers).

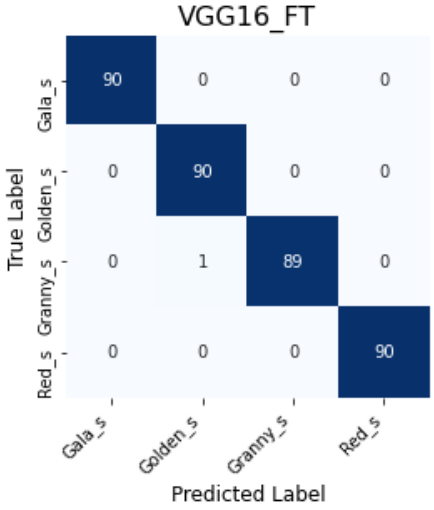
Aunque Lenet5 y VGG16 tienen la misma exactitud en las matrices de confusión (98 %), se debe considerar que el tamaño de las arquitecturas en cuestión de parámetros es muy distinto 2,471,512 y 15,765,828 respectivamente, esto quiere decir que si para la clasificación de manzanas, las condiciones de iluminación en el sistema de visión en sitio no cambian, entonces se recomienda utilizar la arquitectura Lenet5, ya que con esta arquitectura se podría tener un buen desempeño respecto al tiempo de clasificación incluso si se tratara de un sistema embebido con características de bajo costo computacional.



(a) Arquitectura LeNet5.



(b) Arquitectura VGG16 usando extracción de características.



(c) Arquitectura VGG16 usando ajuste fino.

Fig. 12. Matrices de confusión para cada una de las arquitecturas probadas.

5. Conclusiones y trabajo futuro

El rendimiento en clasificación fue de 98 % para Lenet5, 98 % para VGG16 con extracción de características y 99 % para VGG16 con entonación fina, sin embargo, se debe tomar en cuenta que, en la construcción del conjunto de datos de 2,400 manzanas, se utilizó condiciones de iluminación controlada, esto implica que para escenarios con iluminación variable los porcentajes de precisión alcanzados pueden variar.

Si las condiciones de iluminación en el sistema de visión en sitio no cambian, entonces se recomienda utilizar para la implementación la arquitectura Lenet5, esto porque la cantidad de parámetros de la red es mucho menor a la VGG16, pues se podría tener un buen desempeño respecto al tiempo de clasificación incluso si se tratara de un sistema embebido con características de bajo costo computacional.

Como trabajo a futuro, se analizarán daños y enfermedades propias de estas variedades de manzanas, además del análisis foliar de cultivares de manzanas para desarrollar propuestas de solución a los problemas que enfrenta el sector agrícola nacional.

Referencias

1. Baneh, N. M., Navid, H., Kafashan, J.: Mechatronic components in apple sorting machines with computer vision. Journal of Food Measurement and Characterization, vol. 12, no. 2, pp. 1135–1155 (2018) doi: 10.1007/s11694-018-9728-1
2. Bhargava, A., Bansal, A.: Fruits and vegetables quality evaluation using computer vision: A review. Journal of King Saud University-Computer and Information Sciences, vol. 33, no. 3, pp. 243–257 (2018) doi: 10.1016/j.jksuci.2018.06.002
3. Durán, J., González, F.: Sistema automático clasificador de manzanas (2012)
4. Gutiérrez Rico, V.: Cosecha, postcosecha y comercialización de la manzana. Technical report, Consorcio de Coordinación en Salud Integral y el Centro de Multiservicios Educativos (2012)
5. Kondo, N., Ting, K.: Robotics in bioproduction systems. The Society for Engineering in Agricultural, Food, and Biological Systems, vol. 33, no. 29, pp. 1–11 (2000) doi: 10.1016/s1474-6670(17)36744-7
6. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. vol. 86, pp. 2278–2324 (1998) doi: 10.1109/5.726791
7. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., Berg, A. C.: SSD: Single shot MultiBox detector. Computer Vision-ECCV, pp. 21–37 (2016) doi: 10.1007/978-3-319-46448-0_2
8. Moallem, P., Serajoddin, A., Pourghassem, H.: Computer vision-based apple grading for golden delicious apples based on surface features. Information Processing in Agriculture, vol. 4, no. 1, pp. 33–40 (2017) doi: 10.1016/j.inpa.2016.10.003
9. Servicio de Información Agroalimentaria y Pesquera: Panorama agroalimentario 2019. Technical report, SIAP (2018)
10. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014) doi: 10.48550/ARXIV.1409.1556
11. Sofu, M., Er, O., Kayacan, M., Cetişli, B.: Design of an automatic apple sorting system using machine vision. Computers and Electronics in Agriculture, vol. 127, pp. 395–405 (2016) doi: 10.1016/j.compag.2016.06.030
12. Sun, K., Li, Y., Peng, J., Tu, K., Pan, L.: Surface gloss evaluation of apples based on computer vision and support vector machine method. Food Analytical Methods, vol. 10, no. 8, pp. 2800–2806 (2017) doi: 10.1007/s12161-017-0849-7
13. Valdez, P.: Apple defect detection using deep learning based object detection for better post harvest handling (2020) doi: 10.48550/ARXIV.2005.06089
14. VVAA: Google colaboratory (2021) http://colab.research.google.com/ash

ISSN 1870-4069

Identificación de nubes de ceniza volcánica mediante redes neuronales en imágenes MODIS

Ivan Edmundo de la Rosa-Montero[1],
José Carlos Jimenez-Escalona[1],
Hind Taud[1], José Luis Poom-Medina[2]

[1] Instituto Politécnico Nacional,
Escuela Superior de Ingeniería Mecánica y Eléctrica Ticomán,
Sección de Estudios de Posgrado e Investigación,
México

[2] Universidad de Sonora,
División de Ciencias Exactas y Naturales,
México

ivan_edmundo@hotmail.com,
{jjimeneze, htaud}@ipn.mx

Resumen. La identificación de las nubes de ceniza volcánica representa un problema clave para la seguridad de la aviación debido a sus efectos negativos sobre las aeronaves. En este trabajo se han utilizado redes neuronales para identificar nubes de ceniza volcánicas a través de mediciones multiespectrales del Moderate Resolution Imaging Spectroradiometer (MODIS). Con el fin de identificar la presencia de nubes de ceniza volcánica después de un evento eruptivo, se ha entrenado una red neuronal perceptrón multicapa utilizando un conjunto de 14 imágenes MODIS, del volcán Popocatepetl, con presencia de ceniza obtenidas a partir del método de diferencia de temperatura de brillo. Además, se utilizaron métodos de balanceo de datos (undersampling, oversampling, SMOTE) para mejorar el rendimiento de la red. Se han comparado los resultados obtenidos por la red neuronal aplicando los métodos de balanceo de datos: a) Sin balanceo de datos, b) undersampling, c) oversampling y d) SMOTE. Los resultados muestran que el enfoque de red neuronal entrenado con el conjunto de datos no equilibrado presenta en sus gráficas de exactitud un sobreajuste y no es capaz de predecir la presencia de nubes de ceniza volcánica en las nuevas imágenes, esto puede ser debido a que hay un número mucho mayor de píxeles clasificados como ”sin ceniza”que los clasificados como çeniza”. Por otro lado, cuando se entrena la red con el conjunto de datos equilibrado, ésta obtiene un 93.11 % de exactitud y mejora a un 31 % el puntaje F1 en la predicción de la presencia de nubes de ceniza volcánica en las nuevas imágenes.

Palabras clave: Redes neuronales, perceptrón multicapa, ceniza volcánica, MODIS, undersampling, oversampling, SMOTE.

Ivan Edmundo de la Rosa-Montero, José Carlos Jimenez-Escalona, Hind Taud, et al.

Identification of Volcanic Ash Clouds Using Neural Networks in MODIS Imagery

Abstract. The identification of volcanic ash clouds represents a key problem for aviation safety due to their negative effects on aircraft. In this work, neural networks have been used to identify volcanic ash clouds through multispectral measurements from the Moderate Resolution Imaging Spectroradiometer (MODIS). In order to identify the presence of volcanic ash clouds after an eruptive event, a multilayer perceptron neural network has been trained using a set of 14 MODIS images of Popocatepetl volcano with the presence of ash obtained from the brightness temperature difference method. In addition, data balancing methods (undersampling, oversampling, SMOTE) were used to improve the performance of the network. The results obtained by the neural network applying the data balancing methods: a) No data balancing, b) undersampling, c) oversampling and d) SMOTE have been compared. The results show that the neural network approach trained with the unbalanced data set presents in its accuracy plots an overfitting and is not able to predict the presence of volcanic ash clouds in the new images, this may be because there is a much higher number of pixels classified as ”no ash” than those classified as ”ash”. On the other hand, when the network is trained with the balanced dataset, it obtains 93.11% accuracy and improves to a 31% F1 score in predicting the presence of volcanic ash clouds in the new images.

Keywords: Neural networks, Multilayer perceptron, volcanic ash, MODIS, undersampling, oversampling, SMOTE.

1. Introducción

La erupción de un volcán puede ocasionar presencia de diferentes tipos de productos volcánicos en la atmósfera, como son ceniza volcánica y dióxido de azufre (SO2) que pueden ser transportadas tanto verticalmente por efecto de erupción, como horizontalmente, por la acción de los vientos dominantes en la zona. En los casos en los que se tiene abundante emisión de ceniza y esta es transportada a zonas urbanas, esta se deposita en las calles, sobre casas, vehículos, etc.

En las zonas aeroportuarias la caída de ceniza representa una alerta que paraliza las operaciones aeronáuticas. Por otro lado, el tiempo de permanencia de estos materiales en la atmósfera representa un alto riesgo para la seguridad de las aeronaves que utilizan el espacio aéreo alrededor de volcanes activos, debido a los efectos negativos que se han identificado.

Actualmente el uso de imágenes satelitales para la detección de ceniza volcánica es de gran utilidad debido al amplio rango de visión que se tiene con estas. El uso de estas imágenes, a pesar de tener buenos resultados, aún presenta problemas al momento de identificar nubes de ceniza volcánica, por lo que algunos investigadores han optado por implementar herramientas del área de la inteligencia artificial para el estudio de las mismas [6, 4].

Tabla 1. Erupciones volcánicas.

Volcán	Fecha	Hora	Hora de erupción	Sensor
Popocatépetl	14/04/2012	05:10	04:37	Terra
Popocatépetl	14/04/2012	17:20	10:24	Terra
Popocatépetl	16/04/2012	20:11	19:46	Terra
Popocatépetl	18/04/2012	16:55	12:48	Terra
Popocatépetl	20/04/2012	16:45	14:10	Terra
Popocatépetl	25/04/2012	20:10	17:09	Terra
Popocatépetl	14/04/2012	07:55	04:37	Aqua
Popocatépetl	14/04/2012	20:30	10:24	Aqua
Popocatépetl	20/04/2012	19:50	19:09	Aqua
Popocatépetl	21/04/2012	08:05	06:57	Aqua
Popocatépetl	02/05/2012	17:10	16:14	Terra
Popocatépetl	03/05/2012	17:50	15:22	Terra
Popocatépetl	04/05/2012	04:45	04:34	Terra
Popocatépetl	04/05/2012	16:55	13:59	Terra

Debido a los efectos negativos que la parecencia de ceniza volcánica puede ocasionar, se requiere la obtención de datos de una forma muy rápida, desafortunadamente los métodos actuales toman mucho tiempo debido a la enorme cantidad de información que se requiere analizar, por lo cual ha surgido la necesidad de desarrollar nuevos métodos para la detección de nubes, que sean capaces de arrojar la información en el menor tiempo posible.

Por otro lado, aunque el aprendizaje profundo se ha estudiado desde hace muchos años, los resultados más relevantes se han obtenido en los últimos años, incluso se ha superado la capacidad del ser humano en algunas tareas. Una de las razones de ello, es que en la actualidad se cuenta con potentes equipos de cómputo, capaces de llevar a cabo el procesamiento requerido por los algoritmos.

En este trabajo se propone el uso de redes neuronales y de métodos de balanceo, con el fin de detectar de una forma más rápida y precisa nubes de ceniza volcánica y que ayude a la construcción de mapas de prevención y mitigación de riesgo en caso de una erupción que emita productos volcánicos a la atmósfera en la zona de volcanes ubicados en el territorio mexicano.

2. Redes neuronal perceptron multicapa

Un Perceptron multicapa es una red de tipo ”hacia delante” compuesta por varias capas de neuronas entre la entrada y la salida de la misma. Esta red permite establecer regiones de decisión mucho más complejas que las que permite la regresión logística.

Sea K el número de neuronas de la primera capa oculta recibe como entrada un vector de características $\bar{x} = (x_1,\ x_2,\ \ldots,\ x_n)$ y genera como salida un valor calculado en la ecuación 3.4:

$$q_k^{(1)} = w_{0k}^{(1)} + \sum_{i=1}^{n} w_{ik}^{(1)} x_i \ \forall k \in \{1, 2, ..., K\}\,, \tag{1}$$

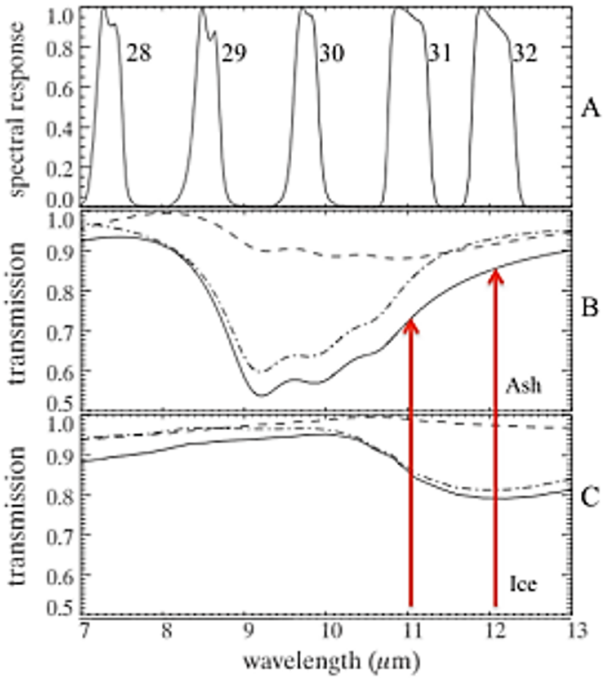


Fig. 1. (A) Respuesta espectral de la bandas MODIS 28,29,30,31 y 32;(B) Transmisión espectral de la ceniza; (C) Transmisión espectral hielo.

donde $q_k^{(1)}$es la salida de la k-ésima neurona de la primera capa, $w_{ik}^{(1)}$ es el peso que conecta la i-ésima componente del vector de características con la k-ésima neurona en la primera capa y $w_{0k}^{(1)}$es el peso asociado a lo que se conoce como sesgo o ”bias” de la red.

A esta salida se le suele aplicar lo que se conoce como una función de activación, la cual se encarga de transformar la salida $q_k^{(1)}$ a un valor acotado acorde a la aplicación. Este proceso puede repetirse para una siguiente capa oculta, utilizando las salidas de las neuronas de la capa anterior como entradas de las neuronas de la siguiente capa, en la cual se repite el proceso. Una red que posee múltiples capas ocultas se denomina red neuronal profunda (Deep Neural Network, DNN).

3. Desequilibrio de clases, consecuencias y métodos de solución

Un problema común al que se enfrenta la minería de datos es el desequilibrio de clases. Se dice que un conjunto de datos está desequilibrado si una clase supera ampliamente a la otra . El problema del desequilibrio de clases se produce cuando la clase que es superada en número es la clase de interés. Una complicación obvia que surge con este problemas es la efectividad de la exactitud para determinar el rendimiento de un clasificador.

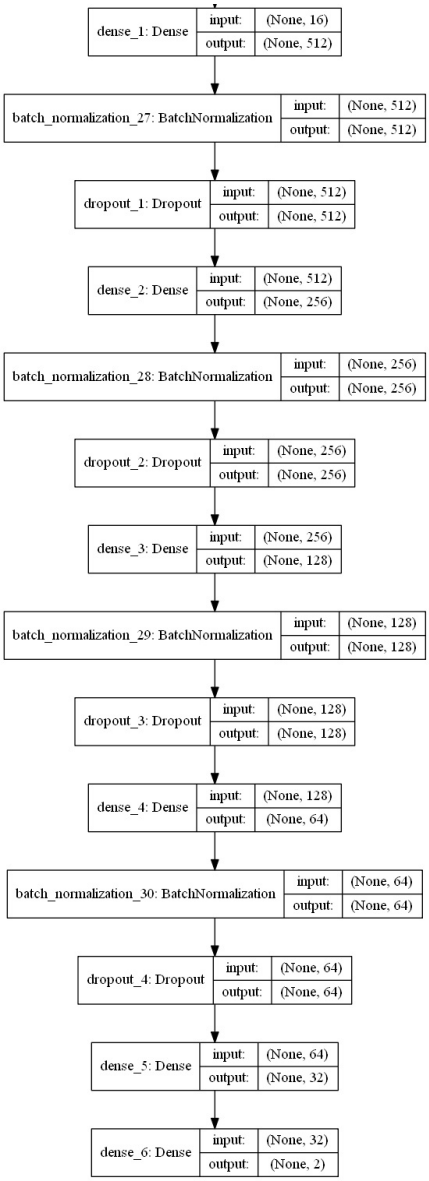


Fig. 2. Arquitectura perceptron multicapa.

Una consecuencia de esta limitación puede observarse en el rendimiento de la mayoría de los clasificadores tradicionales. Esto se debe al hecho de que la mayoría de los clasificadores optimizan la precisión. En consecuencia, se han desarrollado numerosos métodos que tratan de resolver el problema del desequilibrio de clases. Los métodos de muestreo (por ejemplo, sobremuestreo aleatorio y submuestreo aleatorio) se han convertido en enfoques estándar para mejorar el rendimiento de la clasificación [3].

Tabla 2. Matriz de confusión dos clases.

		Clasificador	
		Ceniza	No ceniza
Valor real	Ceniza	TP	FN
	No ceniza	FP	TN

En los métodos de muestreo, el conjunto de capacitación se modifica de manera que se cree una distribución de clases más equilibrada. El conjunto de datos muestreados resultante es entonces más susceptible a los algoritmos tradicionales de extracción de datos, que pueden utilizarse para clasificar los datos.

3.1. Métodos de muestreo

El muestreo es una de las metodologías más populares utilizada para afrontar el desequilibrio de clases. El objetivo de los métodos de muestreo es crear una base datos que tenga relativamente una distribución balanceada de sus clases, así los clasificadores tradicionales tendrán un mejor rendimiento.

Debido a que los métodos de muestreo son utilizados con el fin de realizar una correcta clasificación de clase minoría, el conjunto de datos resultante debe aproximarse al conjunto original. Específicamente, el conjunto resultante debe contener solamente instancias que son similares a las del conjunto original.

Submuestreo aleatorio. En el submuestreo aleatorio, las instancias de la clase mayoritaria se descartan aleatoriamente hasta que se alcanza una distribución de datos más equilibrada. Considérese, por ejemplo, un conjunto de datos compuesto por 10 instancias de clases minoritarias y 100 instancias de clases mayoritarias.

En el submuestreo aleatorio, se podría intentar crear una distribución de clases equilibrada seleccionando 90 instancias de clases mayoritarias al azar para ser eliminadas. El conjunto de datos resultante constará entonces de 20 instancias: 10 instancias de clase mayoritaria (que permanecen aleatoriamente) y 10 instancias de clase minoritaria (las originales).

Sobremuestreo aleatorio. En el sobremuestreo aleatorio, las instancias de las clases minoritarias se copian y se repiten en el conjunto de datos hasta que se alcanza una distribución más equilibrada. Así, si hay dos instancias de clases minoritarias y 100 de clases mayoritarias, el sobremuestreo tradicional copiaría las dos instancias de clases minoritarias 49 veces cada una.

El conjunto de datos resultante consistiría entonces en 200 instancias: las 100 instancias de la clase mayoritaria y las 100 instancias de la clase minoritaria (es decir, 50 copias de cada una de las dos instancias de la clase minoritaria).

SMOTE. Mientras que el submuestreo aleatorio sufre la pérdida de información potencialmente útil, el sobremuestreo aleatorio sufre el problema de overfitting. Específicamente , al replicar aleatoriamente instancias en el conjunto de datos, el modelo aprendido podría ajustar los datos de entrenamiento demasiado cerca y, como resultado, no generalizar bien a los casos no vistos.

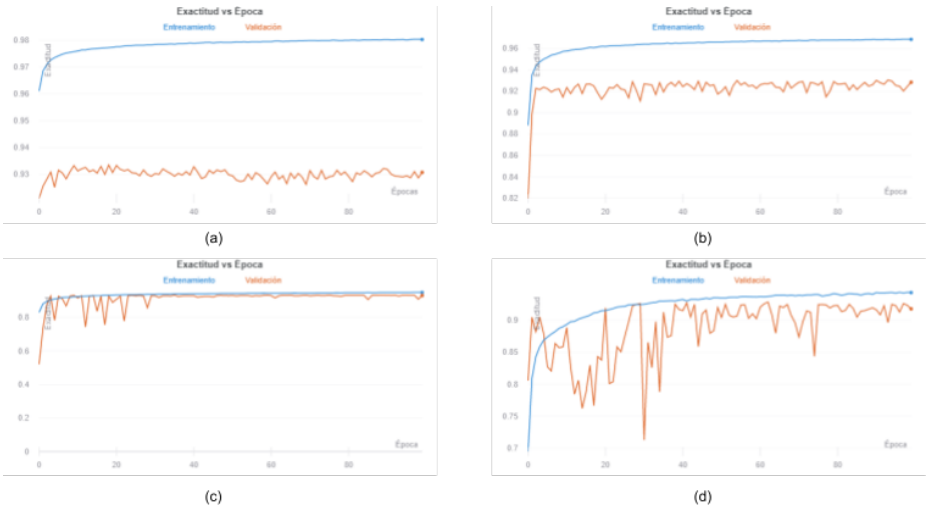


Fig. 3. Gráfica de exactitud vs época con el conjunto de datos. (a) Sin balancear; (b) Oversampling; (c) SMOTE y (d) Undersampling.

Para superar este problema, Chawla et al [3] desarrollaron un método para crear instancias sintéticas en lugar de simplemente copiar las instancias existentes en el conjunto de datos. Esta técnica se conoce como la técnica de sobremuestreo de minoría sintética (SMOTE). Como se ha mencionado, en SMOTE, el conjunto de formación se altera añadiendo instancias de clase minoritarias generadas sintéticamente, lo que hace que la distribución de clases sea más equilibrada.

Decimos que las instancias creadas son sintéticas, ya que son, en general, nuevas instancias minoritarias que han sido extrapoladas y creadas a partir de las instancias de clases minoritarias existentes. Para crear las nuevas instancias de clases minoritarias sintéticas, SMOTE primero selecciona una instancia de clase minoritaria a al azar y encuentra a sus vecinos de clase minoritaria más cercanos k.

La instancia sintética se crea entonces eligiendo al azar uno de los k vecinos más cercanos de b y conectando a a y b para formar un segmento de línea en el espacio de características. Las instancias sintéticas se generan como una combinación convexa de las dos instancias elegidas a y b.

4. Metodología

Con el fin de determinar si es posible reconocer automáticamente patrones de ceniza volcánica en imágenes satélites ópticas que detectan en el infrarrojo Térmico, se estudiaron imágenes MODIS de distintos eventos eruptivos en México. La factibilidad del uso de una red neuronal para detectar cenizas volcánicas se exploró entrenando una red neuronal perceptron multicapa, además del el uso de técnicas de balanceo de datos.

Tabla 3. Comparación de valores de exactitud.

Picchiani et al. [8]	Gray et al. [5]	MLP	Random Oversampling	Random Undersampling	SMOTE
97 %	93.2 %	93.05 %	93.15 %	93.09 %	93.11 %

4.1. Zona de estudio

En México existen amplias provincias volcánicas dentro de las cuales, según el Instituto de Geofísica de la UNAM, en la actualidad se cuenta con 12 volcanes activos. Un detalle importante es que en torno a varios de estos volcanes activos se encuentran localizadas importantes aglomeraciones urbanas y rurales, así como instalaciones aeronáuticas.

El Popocatépetl es uno de los volcanes más activos del país, con una elevación de 5,419.43 msnm, es la tercera cima más alta de México. Se encuentra localizado en la parte central del Cinturón Volcánico Transmexicano, en las coordenadas 19° 01' 23" N y 98° 37' 22" W. Los 25 millones de personas que habitan a menos de 100 km del cráter, lo convierten en uno de los volcanes más peligrosos del planeta.

Después de setenta años de inactividad, se notó un paulatino incremento en la actividad fumarólica del volcán, que reinició su actividad el 21 de diciembre de 1994 [5]. A partir de ese año ha tenido etapas efusivas y explosivas asociadas con el crecimiento y destrucción de domos de lava en el interior del cráter. Sus cenizas han alcanzado las ciudades de Puebla y de México y poblaciones incluso más distantes como Querétaro y Veracruz.

La emisión y dispersión de cenizas, uno de los fenómenos más frecuentes en la actividad de este volcán, ha afectado los estados mencionados, además de Tlaxcala y el Distrito Federal, convirtiéndose en una seria amenaza para la salud pública. Por otra parte, debido a la gran altitud del Popocatépetl (5,419 msnm), las emisiones de gas y ceniza volcánicas han afectado una región y espacio aéreo más amplios.

Una futura erupción del volcán puede poner en peligro la seguridad del tráfico aéreo y afectar varios de los principales aeropuertos de México y países vecinos [2]. Es por ello, que los efectos de la actividad volcánica pueden catalogarse como de índole de seguridad nacional. Para el desarrollo de este trabajo fueron utilizadas 14 imágenes (Tabla 1), las cuales fueron tomadas por alguno de los dos sensores MODIS (Terra o Aqua).

4.2. Percepción remota

Los aerosoles exhiben una transmítancia variable en los rangos de 8-10 y 10-12 μm [1]. Debido a que los canales de las imágenes MODIS cubren este rango espectral, se obtuvieron imágenes de este sensor y se utilizo el método de Split window para determinar áreas de ceniza volcánica en cada evento eruptivo.

En la figura 1, se ilustra la transmisión espectral de la ceniza y del hielo en las bandas 28, 29, 30 , 31 del sensor MODIS. Es posible observar su variación de la transmisión espectral o entre las bandas 30 y 31 (11 μm y 12 μm) del sensor. Esto nos ayuda a discernir entre nubes meteorológicas y nubes de ceniza volcánica.

Tabla 4. Métricas de evaluación para cada modelo.

Modelo	Exactitud	Precisión	Sensibilidad	Puntaje F1
MLP	93.05	98	15	26
R. Oversampling	93.15	89	18	30
R. Undersampling	93.09	49	21	29
SMOTE	93.11	75	19	31

El método de diferencia de temperatura de brillo fue utilizado para detectar las nubes de ceniza volcánica. El método consiste en obtener la diferencia de temperatura de brillo que existe entre dos imágenes adquiridas a dos longitudes de banda diferentes dentro de espectro infrarrojo (11 μm y 12 μm) con el fin de discriminar entre las nubes de ceniza volcánica y las nubes meteorológicas.

4.3. Entrenamiento perceptron multicapa

Durante el diseño de la arquitectura de la red neuronal se considero el uso de batch normalization debido a que nos permite entrenar nuestra red de una manera mas rápida mediante la estandarización de la salidas de cada una de nuestras capaz. Por otro lado, se opto por el uso de dropout como una estrategia para reducir las posibilidades de caer en overfitting. La arquitectura del modelo utilizado consiste en 4 capas ocultas con batch normalization y dropout en cada una de ellas (Figura 2).

Base de datos. Las imágenes ocupadas fueron modificadas de la siguiente manera. Cada imagen fue recortada píxel por píxel de modo que cada píxel formaba una nueva imagen. De esta manera tenemos ahora imágenes de dimensión 1×16 y la base de datos para el entrenamiento consiste en 1728000 imágenes y para la validación consiste en 691200 imágenes.

Métricas de evaluación. Un método común para determinar el rendimiento de un clasificador es a través del uso de una matriz de confusión. Una matriz de confusión de dos clases, en este caso ceniza y no ceniza, se vería de la siguiente manera (tabla 2). Las columnas de la matriz representan las predicciones realizadas por el clasificador para cada clase, y las filas los valores reales por cada clase. por lo cual la matriz queda dividida en 4 clases, TP, FN, FP y TN, que significan lo siguiente:

– **TP – True Positives:** Es el número verdaderos positivos, es decir, de predicciones correctas para la clase ceniza.

– **FN – False Negatives:** Es el número de falsos negativos, es decir, la predicción es negativa cuando el valor real es positivo, en nuestro caso la la prediccion es No ceniza cuando debería ser Ceniza. A estos casos también se les denomina errores de tipo II.

– **FP – False Positives:** Es el número de falsos positivos, es decir, la predicción es positiva cuando el valor real es negativo, en nuestro caso la predicción es Ceniza cuando debería ser No ceniza. A estos casos también se les denomina errores de tipo I.

– **TN – True Negatives:** Es el número de verdaderos negativos, es decir, de predicciones correctas para la clase co ceniza.

Tabla 5. Prueba de hipótesis.

Modelo	p̂	x	n
MLP	15	8379	56170
SMOTE	19	10857	56170

La matriz de confusión nos permite calcular las siguientes métricas.

– **Exactitud** se refiere a la dispersión del conjunto de valores obtenidos a partir de mediciones repetidas de una magnitud. Cuanto menor es la dispersión mayor la precisión.
Se representa por la proporción entre el número de predicciones correctas (tanto positivas como negativas) y el total de predicciones, y se calcula mediante la ecuación 4.1:

$$\frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{2}$$

– **Precisión** se refiere a lo cerca que está el resultado de una medición del valor verdadero. Es representada por la proporción entre los positivos reales predichos por el algoritmo y todos los casos positivos. Se calcula según la siguiente ecuación 4.2:

$$\frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{3}$$

– **Sensibilidad** es una valor que nos indica la capacidad de nuestro clasificador para discriminar los casos positivos de los negativos. También se conoce como Tasa de Verdaderos Positivos. Es la proporción de casos positivos que fueron correctamente identificados por el clasificador. Se calcula según la ecuación 4.3:

$$\frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{4}$$

– **Puntaje F1** es la combinación de las métricas de precisión y sensibilidad. La mejor puntuación es igual a 1 y la peor a 0 (Ecuación 4.4):

$$2 \times \frac{\text{Precision} \times \text{Sensibilidad}}{\text{Precision} + \text{Sensibilidad}}. \tag{5}$$

5. Resultados

Con el objetivo de mejorar el rendimiento de la red neuronal perceptron multicapa es necesario optimizar los hiperparametros utilizados . Los parámetros a optimizar fueron: Batch size, taza de aprendizaje, optimizador y el dropout. Por otra parte, con la finalidad de obtener mejores resultados en la clasificación, diferentes técnicas de balanceo de datos fueron aplicadas. A continuación se muestran los resultados obtenidos mediante la aplicación de estas técnicas. En la figura 3 se muestran las gráficas de exactitud vs época para cada uno de los experimentos.

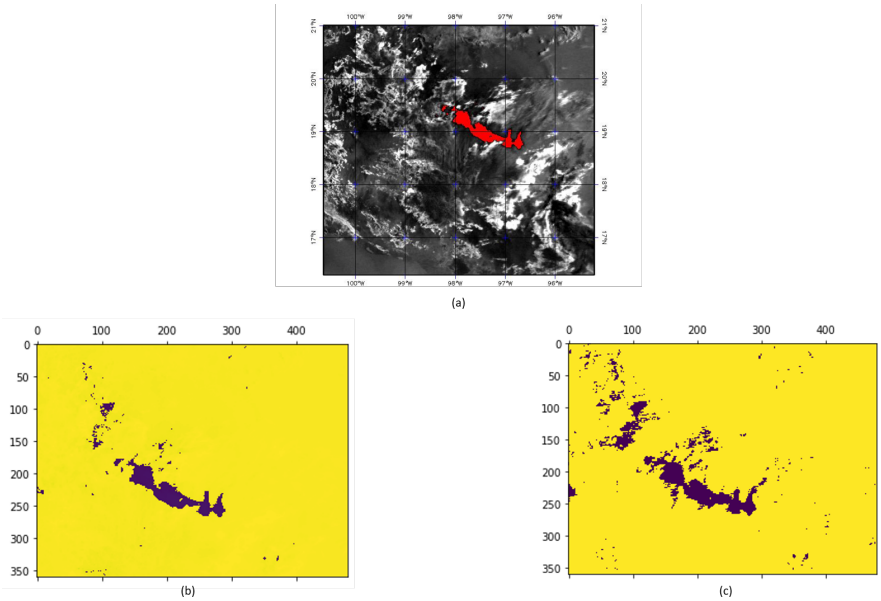


Fig. 4. (a)Imagen con ceniza volcánica; (b)Imagen etiquetada; (c)Predicción del modelo.

Como se puede observar, al entrenar la red con un conjunto de datos sin balancear se cae en el problema de overfitting. Las técnicas de balanceo de datos, en especial la técnica SMOTE, logran mejorar considerablemente las curvas de exactitud. En la Tabla 3 observamos los valores de exactitud obtenidos en cada uno de nuestros experimentos y además son comparados con resultados de trabajos previos.

Como fue explicado anteriormente, la exactitud no es la métrica mas adecuada a tomar en cuenta para este tipo de problemas, cabe destacar que en este trabajo fue considerada con el único fin de comparar resultados con trabajos previos. Es por esto que a continuación se muestran los resultados obtenidos en las demás métricas, (Tabla 4).

Con el fin de discernir si los resultados presentados son significativamente similares, se propone realizar una prueba de hipótesis, en la cual se considerara como referencia la sensibilidad , debido a que esta métrica solo considera las muestras con presencia de ceniza volcánica. Para la prueba consideraremos lo siguiente:

$$Z = \frac{\hat{p_1} - \hat{p_2}}{\sqrt{2\hat{p}(1-\hat{p})/n}}, \tag{6}$$

donde:

$$\hat{p} = (x_1 + x_2)/2n. \tag{7}$$

Se considera la siguiente hipótesis nula:

$$H_0 : p_1 = p_2. \tag{8}$$

Con el fin de rechazar nuestra hipótesis nula debemos probar lo siguiente:

$$Z < -z_\alpha, \tag{9}$$

$$z_\alpha = z_{0,5} = 1{,}645. \tag{10}$$

Lo cual nos indicaría que:

$$p_1 < p_2. \tag{11}$$

Al calcular Z se obtiene lo siguiente:

$$Z = -17{,}79 < -z_\alpha < -1{,}645. \tag{12}$$

Con lo cual podemos decir con un 95 % de confidencia que la mejora en el rendimiento de nuestro modelo al utilizar técnicas de balanceo de datos, como SMOTE, es significativa. En base a los resultados arrojados por los experimentos podemos observar que la técnica mas eficaz para mejorar el rendimiento de la red es el SMOTE, debido a que es el que mejor resultado arroja para la métrica puntaje F1. Los resultados obtenidos en este experimento se muestran en la figura 4.

6. Conclusiones

Se presentó la implementación y comparación de diferentes técnicas de balanceo de datos para su implementación en una red neuronal perceptron multicapa. Se observó que la implementación de la red perceptron multicapa arrojó valores mayores al 90 % en la exactitud, además se disminuyó el sobreentrenamiento mediante la implementación de técnicas de balanceo de datos. Esta red fue entrenada con una base de datos para esta red consistió en 2,419,200 ejemplos cuando no fue balanceada. Además, para la implementación de esta red, se necesito realizar el balanceo de datos con el objetivo de que existiera el mismo número de pixeles ”no ceniza” y ”ceniza”.

En trabajos de investigación previos [6] y [4], se observa que la red neuronal implementada nos arroja un valor para la exactitud mayor al 90 %, siendo este el único valor presentado. En este trabajo se presentan valores similares en la exactitud, cabe resaltar que a diferencia de los trabajos de investigación previos, donde las redes neuronales fueron entrenadas y validadas con imágenes que pertenecen al mismo evento eruptivo, la red neuronal presentada aquí, fue entrenada y validada con diferentes eventos eruptivos. Por lo tanto, la muestra utilizada representa de mejor manera al fenómeno estudiado.

En este trabajo se presentan otros valores importantes como lo son la precisión, sensibilidad y el F1, estos valores nos dan un mejor panorama del funcionamiento del modelo. Como se puede observar los resultados del puntaje F1 siguen siendo bajos a pesar de tener una buena precisión, esto debido a que la sensibilidad sigue siendo baja. Podemos observar que a pesar de que fueron utilizadas técnicas de balanceo de datos, el modelo sigue presentando problemas a la hora de detectar píxeles con ceniza volcánica. Se recomienda seguir recolectando imágenes de diferentes eventos eruptivos con el fin de tener mas ejemplos con píxeles de ceniza volcánica y así el modelo pueda aprender de mejor manera.

Referencias

1. Ackerman, S. A.: Remote sensing aerosols using satellite infrared observations. Journal of Geophysical Research: Atmospheres, vol. 102, no. D14, pp. 17069–17079 (1997), doi: 10.1029/96JD03066
2. Bonasia, R., Scaini, C., Capra, L., Nathenson, M., Siebe, C., Arana-Salinas, L., Folch, A.: Long-range hazard assessment of volcanic ash dispersal for a Plinian eruptive scenario at Popocatépetl volcano (Mexico): Implications for civil aviation safety. Bulletin of Volcanology, vol. 76, no. 1, pp. 1–16 (2013), doi: 10.1007/s00445-013-0789-z
3. Chawla, N. V., Cieslak, D. A., Hall, L. O., Joshi, A.: Automatically countering imbalance and its empirical relationship to cost. Data Mining and Knowledge Discovery, vol. 17, no. 2, pp. 225–252 (2008), doi: 10.1007/s10618-008-0087-0
4. Gray, T., Bennartz, R.: Automatic volcanic ash detection from modis observations using a back-propagation neural network. Atmospheric Measurement Techniques, vol. 8, no. 12, pp. 5089–5097 (2015), doi: 10.5194/amt-8-5089-2015
5. Pereña, R. E.: Historia de la actividad del volcán Popocatépetl (2012)
6. Picchiani, M., Chini, M., Corradini, S., Merucci, L., Sellitto, P., Del Frate, F., Stramondo, S.: Volcanic ash detection and retrievals using modis data by means of neural networks. Atmospheric Measurement Techniques, vol. 4, no. 12, pp. 2619–2631 (2011), doi: 10.5194/amt-4-2619-2011

ISSN 1870-4069

Caracterización de valores atípicos en nube de puntos en 3D para la reducción del tiempo de ejecución en memoria

Israel Sotelo Rodríguez[1], Jesús Carlos Pedraza Ortega[1], Luis Rogelio Román Rivera[2], Juan Manuel Ramos Arreguín[1], Efrén Gorrostieta Hurtado[1]

[1] Universidad Autónoma de Querétaro Campus Aeropuerto, Maestría en Ciencias en Inteligencia Artificial, Querétaro, México

[2] Universidad Autónoma de Querétaro Cerro de las campanas, Doctorado en Ingeniería, Querétaro, México

{isotelo17, lronman26}@alumnos.uaq.mx, caryoko@yahoo.com, jsistdig@yahoo.com.mx, efren.gorrostieta@gmail.com

Resumen. Para poder realizar la reconstrucción de objetos en 3D, puede recurrirse a un tipo de estructura geométrica llamada "Nube de puntos". Una nube de puntos es un conjunto de coordenadas que representan la forma o superficie de un objeto. Uno de los principales problemas al hacer el mapeo 3D de escenas u objetos es la presencia de ruido y valores atípicos en una nube de puntos, producido por distintos factores como la luz solar o artificial, la reflexividad de los objetos, la geometría o limitantes relacionadas con el sensor [1]. La caracterización de los valores atípicos se realizó utilizando nubes de puntos con ruido blanco en distintas magnitudes, y nubes de puntos sin ruido, tomadas de la base de datos de Rakotosaona [2]. Se redujo el porcentaje de coordenadas de puntos, con intervalos de 10% en un rango de 10% a 90%. Se realizaron 144 inferencias utilizando el modelo de PointCleanNet [3] y el resultado se comparó con la etiqueta de los valores atípicos de cada una de las nubes, obteniendo una matriz de confusión para medir la precisión, valor F, y el error cuadrático medio. La matriz de confusión para cada nube mostró que al remover 20% o menos de los puntos pueden alcanzarse inferencias hasta con un 91% de precisión contribuyendo a la disminución del tiempo de inferencia.

Palabras clave: Nube de puntos, inferencia, matriz de confusión, caracterización, valores típicos.

3D Point Cloud Outlier Characterization for Memory Runtime Reduction

Abstract. To reconstruct 3D objects, a type of geometric structure called "point cloud" can be used. A point cloud is a set of coordinates that represent the shape or surface of an object. One of the main problems when doing 3D mapping of scenes or objects is the presence of noise and outliers in the point cloud, produced by different factors such as sunlight or artificial light, object reflectivity,

geometry, or sensor-related constraints [1]. Outlier characterization was performed using point clouds with noise at different magnitudes, and point clouds without noise, taken from the Rakotosaona database [2]. The percentage of point coordinates was reduced, within 10% intervals in a range from 10% to 90%. One hundred forty-four inferences were performed using the PointCleanNet model [3]. Each inference was compared with the outlier label obtaining a confusion matrix to measure the precision, F1-score, and mean square error. The confusion matrix for each cloud showed that by removing 20% or less of the points, inferences with up to 91% accuracy can be achieved, contributing to a decrease in inference time.

Keywords: Point cloud, inference, confusion matrix.

1. Introducción

Cuando un objeto en tercera dimensión es escaneado para obtener una nube de puntos normalmente es contaminada por distintos tipos de ruido o magnitudes, esto provoca que las nubes de puntos tengan que someterse a un preprocesamiento para poder realizar tareas de segmentación o clasificación de forma eficiente.

Existen diversas metodologías para remover los valores atípicos de una nube de puntos, algunas están basadas en extraer características de puntos vecinos para poder clasificar los valores atípicos y reducirlos. La ecuación (1) describe una nube de puntos con presencia de valores atípicos. Dónde P_i, describe el conjunto de puntos de la superficie del objeto y O describe el conjunto de valores atípicos presentes [3]:

$$\boldsymbol{P} = \{\boldsymbol{p_i}, \dots, \boldsymbol{p_n}\} \cup \{\boldsymbol{O_j}\}_{\boldsymbol{O_j} \in \boldsymbol{O}}. \tag{1}$$

De manera general se establecen las siguientes metodologías utilizadas para la reducción de ruido y valores atípicos en nubes de puntos [4]. La reducción de valores atípicos por radio establece que los puntos que corresponden a los valores atípicos tienen menor densidad de puntos vecinos que los puntos que conforman al objeto [10].

La reducción de valores atípicos por esparcimiento utiliza la distancia entre puntos y el número de vecinos, asumiendo que los valores atípicos tienen una distribución normal y que los puntos que tienen en promedio dos sigmas como desviación estándar deben de ser clasificados como valores atípicos [11]. PointCleanNet es un algoritmo que estima los puntos vecinos, para cada punto de la nube de puntos, esto permite que nubes de puntos densas puedan ser procesadas sin perder características esenciales.

Los puntos vecinos se calculan utilizando un radio r, el radio es calculado en base a la diagonal descrita por los valores máximos y mínimos de los puntos frontera. Este tipo de algoritmos están pensados para ser ejecutados en GPU. Las nubes de puntos son cada vez más densas debido a la resolución de los sensores utilizados para el escaneo o reconstrucción de superficies u objetos, produciendo que cada vez más puntos tengan que ser cargados en memoria [8].

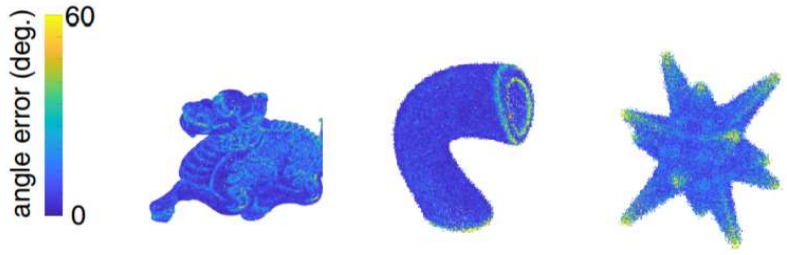


Fig. 1. Estimación de normales de PCPNet [6].

Fig. 2. Nube de puntos de armadillo con distintas magnitudes de ruido [3].

Existen distintas propuestas para evitar el consumo excesivo de memoria para el procesamiento de los puntos vecinos de una nube, por ejemplo, el algoritmo *"k nearest neighbors"* [5] que utiliza la localidad de puntos para intentar reducir el tiempo de ejecución. El tiempo de ejecución para el procesamiento de una nube de puntos está en función de la densidad de la nube de puntos y el algoritmo para la estimación de los puntos vecinos [8].

1.1. Trabajos relacionados

El aprendizaje profundo (en inglés, Deep Learning) es un subcampo del aprendizaje automático. Es una metodología en la que el enfoque principal es el aprendizaje en capas sucesivas de representaciones significativas [9].

Técnicas de aprendizaje profundo utilizadas en redes neuronales han demostrado tener buenos resultados en la reducción de ruido y la eliminación de valores atípicos para las nubes de puntos en 3D. Para atacar este problema han surgido distintos algoritmos como el algoritmo PCPNet [6], el cual mediante la estimación de propiedades locales logra reducir el nivel de ruido en una nube de puntos.

En la fig. 1 podemos observar el resultado cualitativo de PCPNet, contiene un error mínimo en la estimación de normales. Para el aprendizaje sugieren un método basado en parches, el enfoque principal es estimar propiedades de forma local para normales (orientadas y desorientadas) y la curvatura de nubes de puntos sin procesar con presencia alta de ruido [3]. PointNet [7] es un algoritmo enfocado en la clasificación de objetos, segmentación de partes y análisis semántico de la escena.

Una de las principales ventajas que tiene PointNet sobre otros algoritmos de clasificación y segmentación es que, al procesar una nube de puntos, dichos algoritmos requieren que las nubes de puntos sean regulares, por lo que estas tienen que ser sometidas a transformaciones. PointNet no necesita realizar transformaciones de este

tipo, ya que cada punto se trata de forma única e independiente debido a que utiliza un método llamado *"Max Pooling"* como método de discretización [7].

PointCleanNet es un algoritmo dedicado a la reducción de ruido y valores atípicos. Este algoritmo está basado en la reducción de valores atípicos por radio. El radio es utilizado para la estimación de vecinos el cual ayuda a manejar nubes de puntos con alta densidad. La base de datos que utiliza para el entrenamiento del modelo contiene 28 figuras con distintas magnitudes de ruido y su correspondiente nube de puntos sin ruido, de esta forma es posible conocer qué puntos de la nube son valores atípicos.

Para generar los valores atípicos de la nube se introdujo ruido gaussiano con una desviación estándar del 20% de acuerdo con la diagonal definida por los valores máximos y mínimos de cada nube [3].

2. Materiales y métodos

En esta sección se describe, el material utilizado, la caracterización de ruido, la red neuronal y métricas de validación.

2.1- Material utilizado

Para la creación y preprocesamiento de las nubes de puntos se utilizaron las siguientes herramientas.

- Laptop Hp Pavilion 4 Gb RAM, 500 Gb HDD, GTX1650, Rizen 7.
- MeshLab versión 2020.12.
- Pycharm versión 2020.1.
- Anaconda Python 3.7 64-Bit.
- Jupiter Notebook.

Los algoritmos para el preprocesamiento de la nube fueron desarrollados en Python 3.7 utilizando Jupiter Notebook, las nubes eran cargadas en la memoria de la GPU para el preprocesamiento, MeshLab fue utilizado como herramienta de visualización, una nube de puntos con una densidad de 140, 000 puntos al ser visualizada ocupaban 532 MB, por lo que las especificaciones de la Laptop HP Pavilion eran suficientes.

Para las inferencias el hardware y software utilizado fue el siguiente:

- Workstation NVIDIA GeForce RTX 3060 TI. 8GB GDDR6. 4864 CUDA Cores, AMD Ryzen 5600x, 6 Cores, 12 hilos, 3.7GHz 32MB L3 Cache 3MB L2 Cache, 16 GB RAM.
- Python 3.7 64-Bits.
- Docker container Linux Engine Versión 19.03.8.
- Ubuntu 20.04.2 LTS.

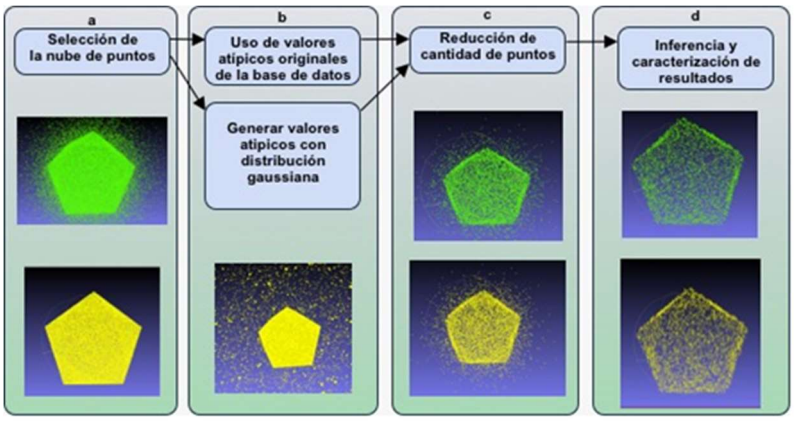


Fig. 3. Metodología propuesta para la caracterización del ruido en distintas magnitudes**.**

Las inferencias fueron realizadas en una estación de trabajo con las especificaciones antes mencionadas, esto debido a que la alta densidad de las nubes de puntos requiere un alto procesamiento en GPU. Se utilizó una tarjeta RTX 3060, con sistema operativo Ubuntu, y contenedores Docker para la manipulación del ambiente de programación.

2.2 Caracterización de ruido

La caracterización de ruido consta de 4 etapas principales como se muestra en la fig. 3., la selección de nubes de puntos a), generación de valores atípicos b), reducción de la densidad de la nube e inferencia c) y caracterización de resultados d).

La caracterización del ruido se realizó utilizando la metodología propuesta en la fig. 3. Se utilizaron nubes de puntos de la base de datos de PointCleanNet para reducción de valores atípicos (a).

De la base de datos se generaron 288 nubes, que corresponden a 4 figuras, de las cuales se obtuvo una nube de puntos con ruido gaussiano con una desviación estándar de 1%, 2.5% y 5% b). El modelo que describe la nube de puntos y el ruido está descrito por la ecuación (2). $\boldsymbol{P'}$ representa la nube de puntos con ruido, P son el conjunto de puntos que representan la superficie del objeto, n_i es el ruido aditivo, $\boldsymbol{O}$ es el conjunto de valores atípicos presentes en la nube [3]:

$$\boldsymbol{P'} = \{\boldsymbol{p_i'}\} = \{\boldsymbol{p_i} + \boldsymbol{n_i}\}_{\boldsymbol{p_i \in P}} \cup \{\boldsymbol{O_j}\}_{\boldsymbol{O_j \in O}}. \quad (2)$$

Se generó un archivo en el que se clasifican cada uno de los puntos. Además, la densidad de las nubes varía con intervalos de 10% en un rango de 10% a 90% c).

Posterior a esto se realizaron 144 inferencias d), con el modelo propuesto por PointCleanNet [3], utilizando como entrada las nubes con distintas magnitudes de ruido y distinta densidad. El objetivo de la inferencia es obtener el rendimiento de la red neuronal con distintos niveles de ruido y densidad para determinar el punto de inflección con el que la remoción de puntos no afecte el rendimiento y a la vez contribuya a reducir el tiempo de inferencia.

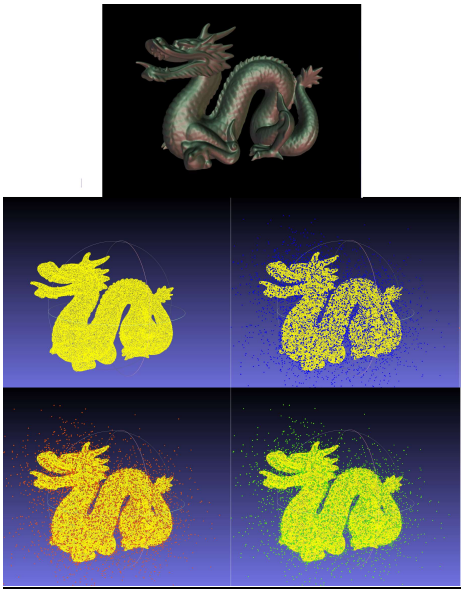


Fig. 4. Nube de puntos de dragón con distintas magnitudes de ruido con una desviación estándar de 1%, 2.5% y 5% y distinta densidad.

2.3 Red neuronal

La arquitectura de la red neuronal convolucional utilizada es la propuesta por PointCleanNet [3], tiene dos etapas principales, una etapa dedicada a reducir el número de valores atípicos y otra a reducción de ruido.

Para la reducción de valores atípicos la red neuronal está basada en las arquitecturas de PCPNet [6] y PointNet [7], debido a que utilizan Perceptrones de múltiples capas para la extracción de características significativas, además utilizan una metodología para saltar conexiones con el objetivo de promover la propagación del gradiente y mejorar el entrenamiento [3].

Está arquitectura de red, además utiliza reducción de valores atípicos por radio, está metodología es utilizada para estimar que tan probable es el punto central de ser un valor atípico.

El modelo de la nube de puntos es descrito por la ecuación (1), la nube de puntos está compuesta por los puntos que describen la superficie del objeto, y que además cuenta con presencia de ruido y valores atípicos. Para el entrenamiento se utiliza una función no lineal que es utilizada para remover los valores atípicos [3]:

$$\tilde{O}_i = g(P_i') > 0.5\,. \tag{3}$$

La ecuación (3) representa la probabilidad de valor atípico $\tilde{O}_i$, establece que un punto es añadido al conjunto de valores atípicos si la probabilidad de valor atípico es mayor al 0.5 [3].

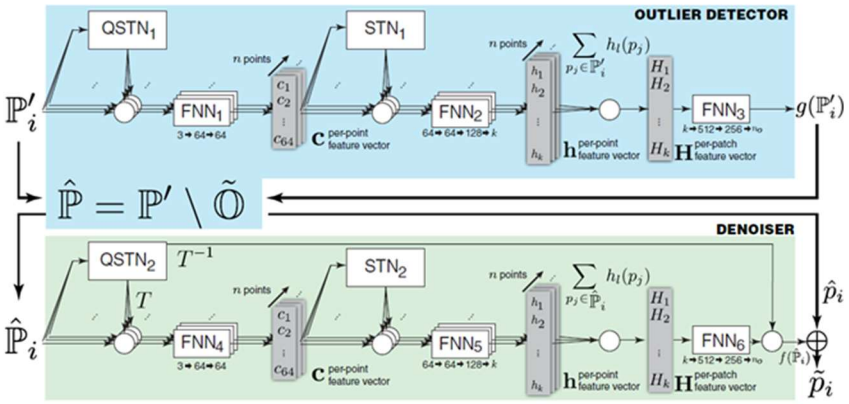


Fig. 5. Diagrama de red convolucional de PontCleanNet[3], la red recibe como entrada una nube de puntos con presencia de ruido Pi, ésta es procesada por una red neuronal utilizada para la clasificación de valores atípicos y obtener una nube de puntos con reducción de outliers y el conjunto de outliers removidos $\widehat{P_i}/\tilde{O}$ [3].

2.4 Criterios de evaluación

Cuando intentamos reducir el nivel de valores atípicos en una nube de puntos debemos de considerar que los puntos que se están removiendo con el modelo propuesto correspondan a los valores atípicos (True Positive), que la cantidad de puntos que se remuevan erróneamente sean cuantificados (False Positive). Estas dos métricas contribuyen a la Precisión (ecuación 4), Recall (ecuación 5), F1-Score (ecuación 6) y el Error Cuadrático Medio (ecuación 7) del modelo [3]:

$$Precisión = \frac{\sum true\ positive}{\sum Predicted\ Positive\ Condition}, \tag{4}$$

$$True\ Positive\ Rate, Recall = \frac{\sum true\ positive}{\sum Positive\ Condition}, \tag{5}$$

$$F1 - Score\ = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \tag{6}$$

$$\text{MSE}\ = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2. \tag{7}$$

3. Resultados

En esta sección se describen los resultados cuantitativos y cualitativos obtenidos después de realizar las inferencias para la etapa de reducción de valores atípicos utilizando el modelo propuesto por PointCleanNet.

3.1 Resultados cuantitativos

La tabla 1 muestra los resultados de las inferencias de las nubes de puntos con presencia de valores atípicos realizadas con el modelo propuesto por PontCleanNet [3], cada columna de la tabla representa las métricas de evaluación así como el tiempo de ejecución en GPU. Se realizaron inferencias para 144 nubes de puntos con distintas magnitudes de ruido con una desviación estándar de 1%, 2.5% y 5% y distinta densidad como se muestra en la tabla 1.

La densidad de la nube está en función del porcentaje de puntos que se removieron, siendo de menor densidad aquellas nubes a las que se removió mayor porcentaje de puntos. Las inferencias se realizaron utilizando la estación de trabajo definida en la sección de material utilizado. Se puede observar en la tabla 1. que para una densidad de 112, 000 puntos se obtiene una Precisión del 87.32% (ecuación 4), Recall del 84.08 % (ecuación 5), F1-Score del 91.48% (ecuación 6), MSE del 05.09% (ecuación 7), y tiempo de ejecución en GPU de 68 minutos.

Para esta densidad de puntos el error cuadrático indica que el rendimiento del modelo no se degrada con la reducción de la densidad de puntos. Por otra parte, los resultados obtenidos en nubes de puntos con una densidad de 84, 000 puntos indican una degradación en la clasificación de los valores atípicos, en comparación con los resultados obtenidos con una densidad de 112,000 puntos.

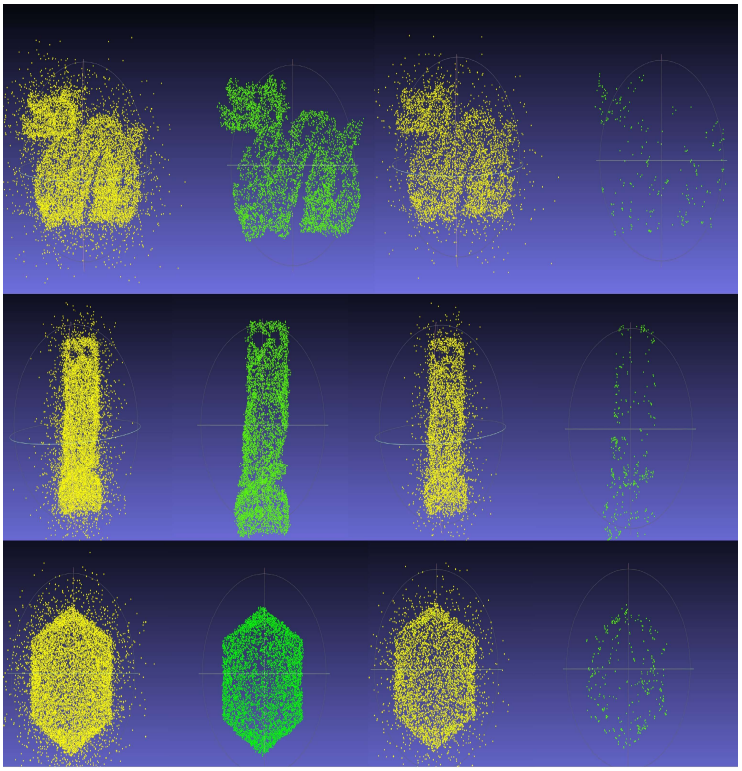
3.2 Resultados cualitativos

La siguiente imagen muestra tres figuras distintas, las figuras cuentan con distinto nivel de valores atípicos y distinta densidad. Cabe recalcar que las inferencias corresponden al análisis hecho para 126,000 y 84,000 puntos respectivamente, el propósito principal es mostrar el punto en el que el rendimiento del modelo conlleva a una degradación en la clasificación de valores atípicos.

Nubes de puntos con una densidad de 84,000 puntos o menor, tienen a degradar el rendimiento del modelo. Por otro lado el tiempo de inferencia disminuye cuando la densidad es mayor a 112,000 puntos, dónde el rendimiento de la red no se ve perjudicado obteniendo F1-Score de 91.48%.

4. Conclusiones y trabajo futuro

La inferencia para el procesamiento de una nube con 140,000 puntos toma en promedio 85.71 minutos. Para una nube con 112,000 puntos toma en promedio 68.57 minutos. Reducir el tamaño de una nube de puntos en un 20% contribuye a la reducción del tiempo de inferencia en un 24.99%. El f1-score muestra que al remover el 20% de los puntos se obtiene 91.48%, es aquí cuando la reducción del tiempo de ejecución alcanza su óptimo, ya que se preserva la clasificación de los valores atípicos. Por el contrario, analizando los resultados de la nube con una densidad de 84,000 puntos, esta toma en promedio 51.31 minutos, pero la reducción del rendimiento para la clasificación de nubes de puntos es evidente con un 77.05% de f1-score.



a) b) c) d)

Fig. 6. Comparativa de resultados obtenidos para nubes de puntos con presencia de valores atípicos. De izquierda a derecha y de arriba abajo: a) Nubes con 126,000 puntos y valores atípicos. b) Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 126, 000 puntos. c) Nubes con 84,000 puntos y valores atípicos., d) Inferencia utilizando el modelo propuesto por PointCleanNet para las nubes de 84, 000 puntos.

Las nubes de puntos procesadas permitieron analizar los resultados de las inferencias para verificar bajo qué circunstancias la red neuronal tiene resultados aceptables, es decir cuando la degradación de las características de la figura en si se ven perjudicadas, esto para determinar hasta qué punto es recomendable reducir la densidad de la nube, además permite determinar cuando la densidad de la nube afecta de manera significativa el rendimiento de la red.

Al hacer la inferencia en la nube de puntos podemos observar que uno de los factores que afectan a la reducción de ruido y valores atípicos es la forma o la figura de las nubes de puntos, así como el nivel de ruido presente. Nubes de puntos con formas complejas se ven afectadas dando como resultado a la pérdida de características en la nube.

Como trabajo futuro, se tiene considerado continuar con el desarrollo e implementación de una red neuronal convolucional que contribuya a mejorar el rendimiento de redes neuronales para la reducción de ruido y valores atípicos, así como mejorar el tiempo de ejecución de nubes de puntos densas. Continuar con la

Tabla 1. Comparación de resultados cualitativos.

Densidad de nube pts.	Precisión %	Recall %	F1-Score %	MSE %	GPU min
14, 000	29.78	100	45.89	72.71	8.58
28, 000	29.23	99.75	45.23	68.50	17.14
42, 000	32.84	99.26	49.39	58.14	25.71
56, 000	39.64	96.72	56.57	42.41	34.24
70, 000	50.19	94.68	66.40	27.92	42.78
84, 000	63.70	92.13	77.05	16.31	51.31
98, 000	77.93	87.43	86.33	08.59	59.85
112, 000	87.32	84.08	91.48	05.09	68.57
126, 000	95.38	80.75	95.03	02.84	77.14
140,000	99.61	90.78	94.99	02.75	85.71

implementación de pruebas para medir el desempeño de la red y determinar áreas de mejora.

Referencias

1. Sotoodeh, S.: Outlier detection in laser scanner point clouds. International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 36, no, 5, pp. 297–302 (2006) doi: 10.3929/ethz-b-000037220
2. Rakotosaona, M. J., La Barbera, V., Guerrero, P., Mitra, N. J., Ovsjanikov, M.: Index of /projects/2019/pointcleannet/data. Recuperado 6 de enero de 2021, de http://geometry.cs.ucl.ac.uk/projects/2019/pointcleannet/data/ (2021)
3. Rakotosaona, M. J., La Barbera, V., Guerrero, P., Mitra, N. J., Ovsjanikov, M.: PointCleanNet: Learning to denoise and remove outliers from dense point clouds. In Computer Graphics Forum, vol. 39, no. 1, pp. 185–203 (2019) doi: 10.1111/cgf.13753
4. Javaheri, A., Brites, C., Pereira, F., Ascenso, J.: Subjective and objective quality evaluation of 3D point cloud denoising algorithms. In: Proceedings of IEEE International Conference on Multimedia & Expo Workshops (ICMEW), pp. 1–6 (2017) doi: 10.1109/ICMEW.2017.8026263
5. Sankaranarayanan, J., Samet, H., Varshney, A.: A fast all nearest neighbor algorithm for applications involving large point-clouds. Computers & Graphics, vol. 3, no. 2, pp. 157–174 (2007) doi: 10.1016/j.cag.2006.11.011
6. Guerrero, P., Kleiman, Y., Ovsjanikov, M., Mitra, N. J.: PCPNet learning local shape properties from raw point clouds. Computer Graphics Forum, vol. 37, no. 2, pp. 75-85 (2018) doi: 10.1016/j.cag.2006.11.011
7. Qi, C. R., Su, H., Mo, K., Guibas, L. J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference On Computer Vision and Pattern Recognition, pp. 652–660 (2017)

8. Javaheri, A., Brites, C., Pereira, F., Ascenso, J.: Subjective and objective quality evaluation of 3D point cloud denoising algorithms. In: Proceedings of IEEE International Conference on Multimedia & Expo Workshops ICMEW´17. pp. 1–6 IEEE (2017) doi: 10.1109/ICMEW.2017.8026263.
9. Chollet, F.: Deep learning with python: Das Praxis-Handbuch vom Entwickler der Keras-Bibliothek. MITP-Verlags GmbH & Co. KG (2017)
10. Schoenenberger, Y., Paratte, J., Vandergheynst, P.: Graph-based denoising for time-varyingpoint clouds. In: Proceedings of 3DTV Conference: The True Vision-Capture, Transmission and Display of 3D Video, Lisbon, Portugal, pp. 1–4 (2015) doi: 10.1109/3DTV.2015.7169366
11. Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., Beetzet, M.: Towards 3D point cloud based object maps for household environments. Robotics and Autonomous Systems, vol. 56, no. 2, pp. 927–941 (2008) doi: 10.1016/j.robot.2008.08.005

ISSN 1870-4069

Pronóstico de series de tiempo de imágenes de sequías utilizando autocodificadores y redes neuronales

Manuel Medrano, Juan Flores, Héctor Rodríguez,
Rodrigo López, Carlos Lara

Tecnológico Nacional de México,
Instituto Tecnológico de Culiacán,
México

manuel_medrano@itculiacan.edu.mx

Resumen. Las sequías son uno de los peligros climáticos que más afectan a las poblaciones. Estas son causadas en su mayoría debido por el calentamiento global. De esta manera, grandes zonas de población pueden quedarse sin agua durante un periodo de tiempo indefinido. Un pronóstico correcto de este tipo de eventos puede permitir a grandes poblaciones a prepararse para contrarrestar estas situaciones. Tradicionalmente el pronóstico de series de tiempo se realiza mediante un conjunto de datos numéricos con el cual se puedan encontrar patrones de comportamiento de la información dentro de los siguientes n espacios de tiempo. Este trabajo propone una nueva metodología para predecir series de tiempo de imágenes basada en autoencoders (autocodificadores) aplicada al estudio de las sequías. Utilizando una serie de tiempo en formato de imagen, es posible la obtención de los siguientes pasos de tiempo, y retornar la información de pronóstico como una imagen que representa el futuro, mediante la combinación de modelos de tipo autoencoder y múltiples redes neuronales.

Palabras clave: Series de tiempo, autoencoder, redes neuronales, generador de imágenes.

Drought Imaging Time Series Forecasting Using Autoencoders and Neural Networks

Abstract. Droughts are one of the climatic hazards that most affect populations. These are mostly caused by global warming. In this way, large population areas can be left without water for an indefinite period of time. A correct forecast of this type of event can allow large populations to prepare to counteract these situations. Traditionally, time series forecasting is done using a set of numerical data with which patterns of behavior of the information can be found within the following n periods of time. This work proposes a new methodology to predict image time series based on autoencoders (autocoders) applied to the study of droughts. Using a time series in image format, it is possible to obtain the following time steps, and return the forecast information as an image that represents the future, by combining autoencoder-type models and multiple neural networks.

Keywords: Time series, autoencoder, neural networks, image generator.

1. Introducción

En años recientes las sequías se han visto aumentadas en el mundo, debido al cambio climático. Este evento climático afecta a más cantidad de personas que cualquier otro peligro climático [9]. En el trabajo de Kallis [9] describen a las sequías como una falta temporal de agua, que es necesariamente pero no explícitamente causada por una anomalía climática y que daña a una actividad, a grupos o al medio ambiente.

Algunos de los problemas causados por las sequías son: Reducción de cultivos, pastizales, productividad forestal, niveles de agua superficiales y subterráneas, así como aumento de incendios y tasa de mortalidad de ganado y vida silvestre. De igual manera este evento climático tiene repercusiones en cuanto al suministro de agua y alimentos en la actividad económica.

En este trabajo se aborda la estimación de las sequías mediante el pronóstico de series de tiempo de imágenes. El pronóstico de series de tiempo es una técnica estadística utilizada comúnmente para la estimación de valores en el futuro a partir de datos históricos [16]. Mediante la utilización de información estructurada, que representa información histórica tomada en periodos regulares de tiempo, y la utilización de técnicas de pronóstico, se puede obtener un conjunto de valores que representan la información futura.

Gracias al pronóstico de series de tiempo se pueden obtener diferentes beneficios. Uno de los usos que generalmente se le otorga a esta técnica es la toma de decisiones. Debido a que el pronóstico de series de tiempo muestra valores que aun no han sido presentados, se puede tomar una acción inmediata, en diferentes áreas y con diferente nivel de impacto, como: capacidad de producción, demanda de producción, pronóstico de clima, sequías, inundaciones, entre otros.

De esta manera se pueden tomar decisiones tanto para mejorar como para combatir amenazas. Por ejemplo, en el trabajo [17] muestran, mediante técnicas de pronóstico, una forma de prevenir inundaciones repentinas observando las descargas máximas en el pasado. Comúnmente el tipo de información de series de tiempo que es utilizada para el pronóstico es representada con un vector.

Otra representación alternativa es el formato de imagen o en forma de matriz. Siguiendo el mismo procedimiento que las series de tiempo, las imágenes son tomadas periódicamente en un intervalo constante de tiempo. De tal manera que, al igual que con la información numérica, sea posible hacer un pronóstico de una serie de imágenes, y dicho pronóstico representarán una estimación futura.

Los enfoques comunes pueden ser los siguientes: 1) Obtener información relevante de un conjunto de imágenes para representarlas en series de tiempo con formato numérico, y trabajar con esta información, como lo muestra en [18]; 2) Transformar un conjunto de series de tiempo con formato numérico a un conjunto de imágenes y que estas sean evaluadas por un proceso que pueda manejarlas en este formato, tal como se realiza en [10].

Este trabajo propone la compresión de la serie de tiempo de imágenes mediante el empleo de un autoencoder para la reducción del espacio de búsqueda. Con la información codificada generamos n redes neuronales (una red por cada píxel contenida en la imagen codificada).

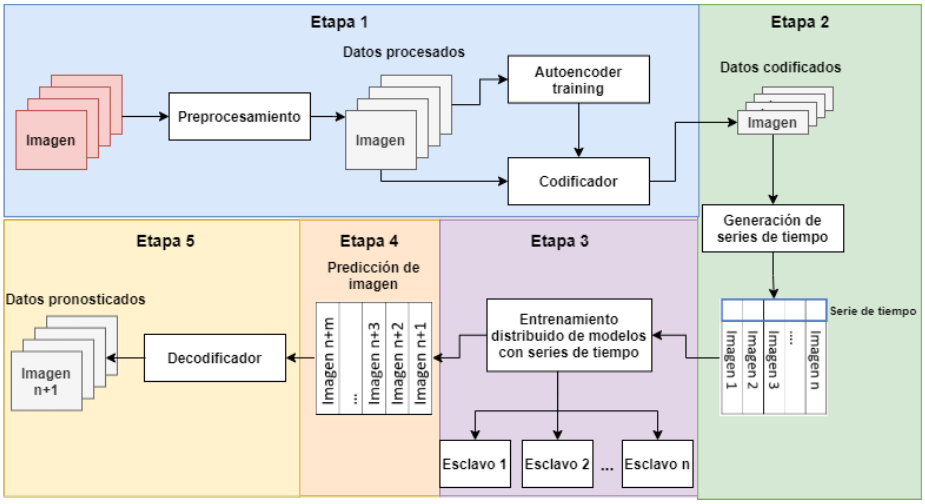


Fig. 1. Procedimiento de pronóstico de imágenes propuesto.

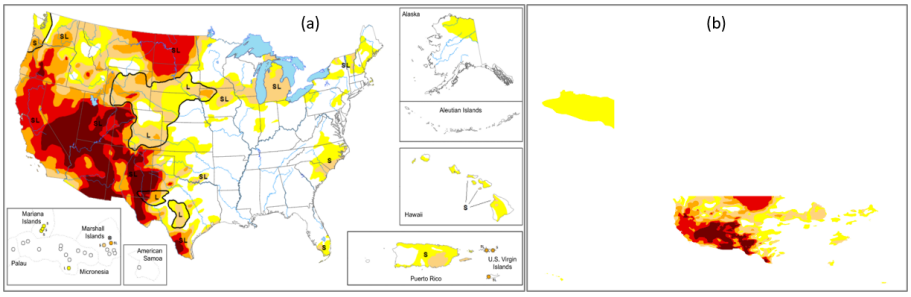


Fig. 2. a) Mapa original de sequías. b) Representación de la sequía de EU. Obtenidos del sitio [15].

Para posteriormente generar el siguiente instante en el tiempo (imagen codificada). Finalmente, decodificamos la imagen generada por las n redes neuronales para obtener la representación real. En este trabajo planteamos otra aproximación para solucionar el pronóstico de series de tiempo utilizando un conjunto de imágenes tomadas periódicamente que representan zonas de sequías, esta información es de tipo "Geographic Information System"(GIS) en formato "Web Map Service" (WMS) [15].

Considerando que este conjunto de imágenes puede ser tratado como una serie de tiempo, es válido afirmar que, utilizando directamente este conjunto de imágenes, se pueda generar la información histórica futura en el mismo formato de imagen. De tal manera que, la imagen de salida del proceso sea una imagen totalmente nueva muy similar a la imagen original y, que es pronosticada por un conjunto de modelos de aprendizaje máquina.

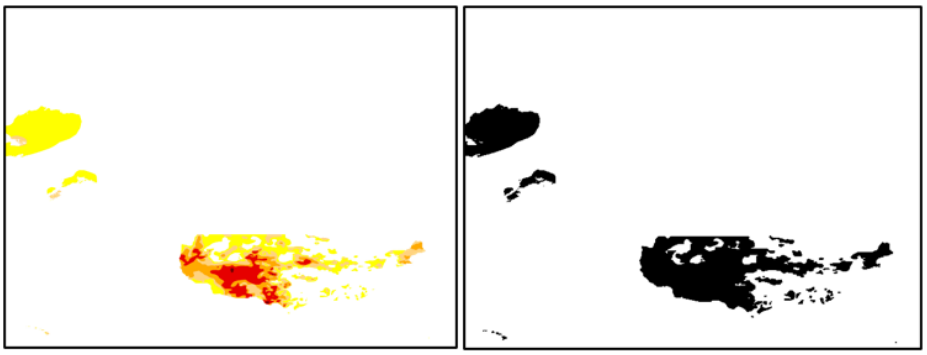


Fig. 3. Procesamiento de imágenes de color a imagen binaria.

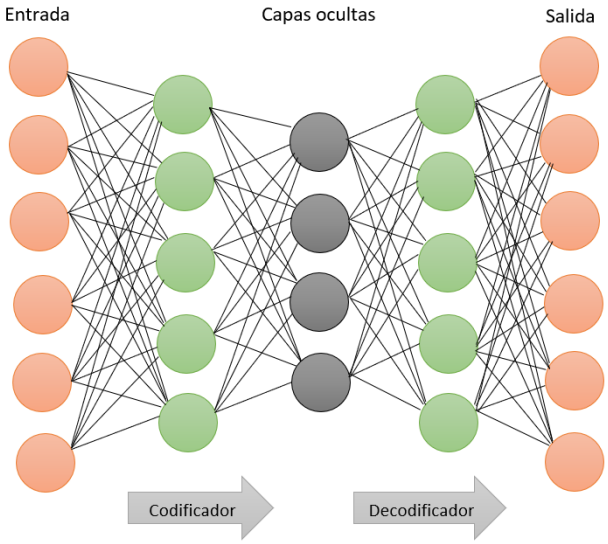


Fig. 4. Arquitectura de una red neuronal de autoencoder. Tomado de [8].

2. Trabajo relacionado

El pronóstico de series de tiempo se puede encontrar en diferentes áreas, como puede ser la Economía, Energía, Medicina o la Ingeniería [7]. Mediante la utilización de diversas técnicas, se pueden obtener estimaciones minimizando la incertidumbre lo más posible [1]. En el trabajo [5] se presentan, discuten y evalúan una serie de técnicas para pronósticos de series de tiempo, estos modelos son:

Redes neuronales artificiales (ANN), Media Móvil Integrada Auto-Regresiva (ARIMA), Maquinas de Soporte Vectorial (SVM), Razonamiento basado en casos (CBR), Series de tiempo difusas, Modelo de predicción gris, Media Móvil y Suavisado Exponencial (MA & ES), K-Vecinos cercanos (KNN) y Modelos híbridos.

Tabla 1. Niveles de codificación, muestra original 480×640.

Conjunto de capas	Dimensiones de codificación	Total de datos
2	(120, 160, 8)	153,600
3	(60, 80, 4)	19,200
4	(30, 40, 4)	4,800
5	(15, 20, 4)	1,200

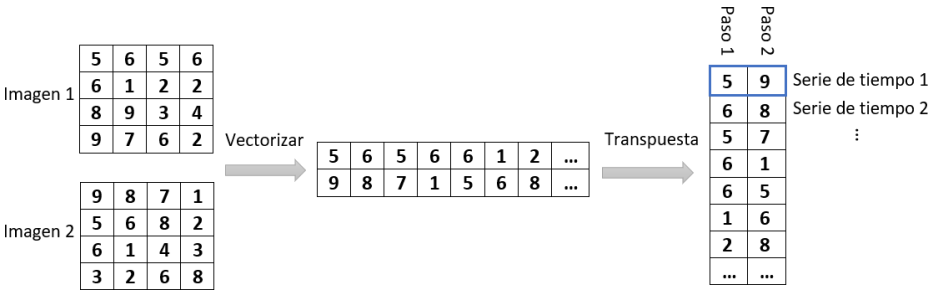


Fig. 5. Generación de series de tiempo por imágenes codificadas.

Concluyendo que la utilización de las técnicas antes mencionadas ofrecen una serie de ventajas y desventajas, y que hacen un énfasis en la utilización de los modelos híbridos, pues estos permiten tomar ventajas de las fortalezas y debilidades de cada uno de los modelos. Existen diferentes enfoques en cuanto al pronóstico de series de tiempo. Muchos de estos enfoques utilizan las características distintivas de las imágenes, definidas por la aplicación dada, para realizar un pronóstico basado en la secuencia de imágenes.

Por ejemplo, un enfoque básico de pronóstico es utilizar dos imágenes consecutivas para pronosticar un vector de características. Como se realiza en [12] donde se asume que todos los objetos de las imágenes se trasladan en la misma dirección, y de esta manera encuentra el mejor vector que representa dicha traslación. Otra suposición de este enfoque básico puede ser que las imágenes no son consecutivas, pero los píxeles se intensifican y permanecen constantes al pasar del tiempo.

En [3] utilizan un enfoque basado en la estimación de vectores de movimiento por el método de correlación cruzada, mediante la partición de las imágenes en subconjuntos de píxeles del mismo tamaño y asigna el vector que une los subconjuntos de imágenes consecutivas con el mayor coeficiente de correlación cruzada. En cambio, en [14] se maximiza la probabilidad del campo vectorial dadas dos imágenes consecutivas, mediante la utilización de la metaheurística del recocido simulado.

Otros enfoques utilizan los modelos de regresión. Como en [19] donde se realiza un análisis de sensibilidad de la respuesta de los parámetros de crecimiento de la altura del tallo, el índice de área foliar y la biomasa al número de días después de la siembra. También es posible utilizar Máquinas de Soporte Vectorial (SVM), enfocándose en determinar si se pueden separar los puntos de datos con un hiperplano $(k-1)$ dimensional.

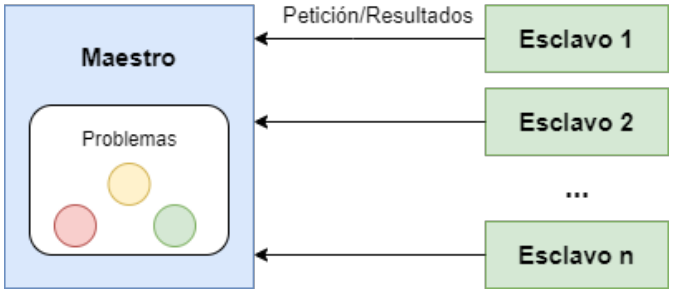


Fig. 6. Metodología maestro-esclavo.

Tabla 2. Niveles de codificación, muestra original 480×640.

Forma	Redes neuronales	Tiempo	Error autoencoder	Precisión paso 1	Error paso 1
(15,20,4)	1,200	0d-0h-52m-58s	0.0343	0.9561	0.0439
(30,40,4)	4,800	0d-3h-28m-51s	0.0187	0.9682	0.0318
(60,80,4)	19,200	0d-14h-10m-41s	0.0163	0.9805	0.0195
(120,160,8)	153,600	4d-17h-40m-18s	0.0059	0.9849	0.0151

En [4] utilizan este enfoque mediante el análisis de la evolución espacio-temporal de la superficie del mar, extrayendo las series de tiempo de imágenes de radar. Algunos enfoques utilizan técnicas de aprendizaje automático, como puede ser las Redes Neuronales Artificiales (ANNs). Las ANNs son una aproximación que puede ser utilizada en una amplia gama de problemas relacionados con las series de tiempo.

Por ejemplo, en [13] se utiliza una ANN para el pronóstico de la irradiancia global horizontal dadas dos entradas: campos vectoriales de movimiento de nubes y imágenes de índice de nubes. Otro enfoque que ha sido utilizado de manera exitosa es el aprendizaje profundo. En este campo los modelos comúnmente utilizados son las Redes Neuronales de Convolución (CNN) para el manejo de imágenes y las Redes de memoria a corto plazo (LSTM) para el manejo de secuencias correlacionadas.

En [11] se transforma una red de usuarios de metro en imágenes, obteniendo así con una CNN la información espacial de los datos. De manera separada, los datos de la serie temporal se introducen en un modelo LSTM que extraen de manera apropiada las características temporales, para luego combinar la información espacial y las características espaciales con una red ANN que realiza el proceso de pronóstico del número de pasajeros del metro. Con lo anterior se puede concluir que la utilización de arquitecturas híbridas presentan buenos resultados si se utilizan de manera adecuada en una parte específica de la arquitectura.

De esta manera, en este trabajo se realiza una implementación híbrida con múltiples arquitecturas en diferentes puntos. Utilizando un autoencoder, para la codificación de imágenes, se busca la reducción de la cantidad de características necesarias para su implementación. En conjunto con la utilización de una arquitectura que utiliza LSTMs, para la extracción de características temporales, junto con ANNs, para la extracción de características y pronósticos.

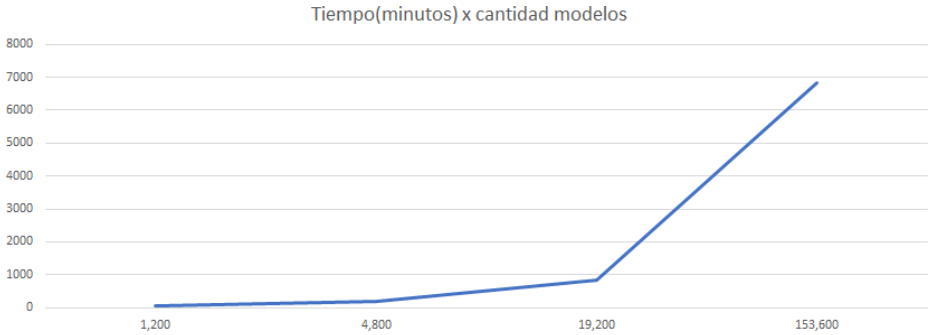


Fig. 7. Relación cantidad de modelos por tiempo de ejecución.

3. Materiales y métodos

Para realizar la estimación del siguiente instante de tiempo (imagen) dada una secuencia de imágenes consecutivas se realizan en 5 pasos o etapas. La primera etapa consiste en comprimir (codificar) la información mediante el empleo de autoencoders. La segunda etapa utiliza la información codificada (comprimida), la vectoriza y genera las n series de tiempo (donde n es igual al número de píxeles en cada imagen codificada).

La tercera etapa genera y entrena n redes neuronales (una por cada píxel contenido en la imagen codificada). La cuarta etapa utiliza las redes entrenadas y genera la siguiente imagen (codificada) en el tiempo. Para finalmente la quinta etapa utilizando en autoencoder previamente definido de codifica la imagen generada por las n redes neuronales. La Figura 1 muestra el flujo del proceso principal del pronóstico de imágenes.

3.1. Datos

Los datos utilizados en este trabajo fueron tomados de una pagina web que mantiene un monitor de las sequías en Estados Unidos semana con semana. La información fue tomada por un grupo de expertos que a partir de esta información la transforma en un mapa que muestra las regiones de Estados Unidos que se encuentran en una sequía.

El mapa utiliza cinco categorías, cada una de estas categorías muestra su grado de impacto en las diferentes áreas, estas categorías son: D0 anormalmente seco, esto es una sequedad a corto plazo que detiene o hace más lento el crecimiento de cultivos/pastizales; D1 sequía moderada, algunos cultivos/pastizales han sufrido daños, y las fuentes, corrientes, embalses o posos presentan signos de escasez; D2 Sequía severa, posibilidad de pérdidas en cultivos/pastizales y la escasez de agua es común.

D3 Sequía extrema, hay perdidas significantes de cultivos/pastizales y hay restricciones de agua extendidos; D4 Sequía excepcional, hay perdidas excepcionales de cultivos/pastizales y escasez de agua en fuentes, corrientes, embalses o posos [15]. En este trabajo se utiliza información desde el 4 de Enero del 2000 hasta el 1 de Agosto del 2020. Obteniendo así un conjunto de datos de 1,079 muestras para utilizar.

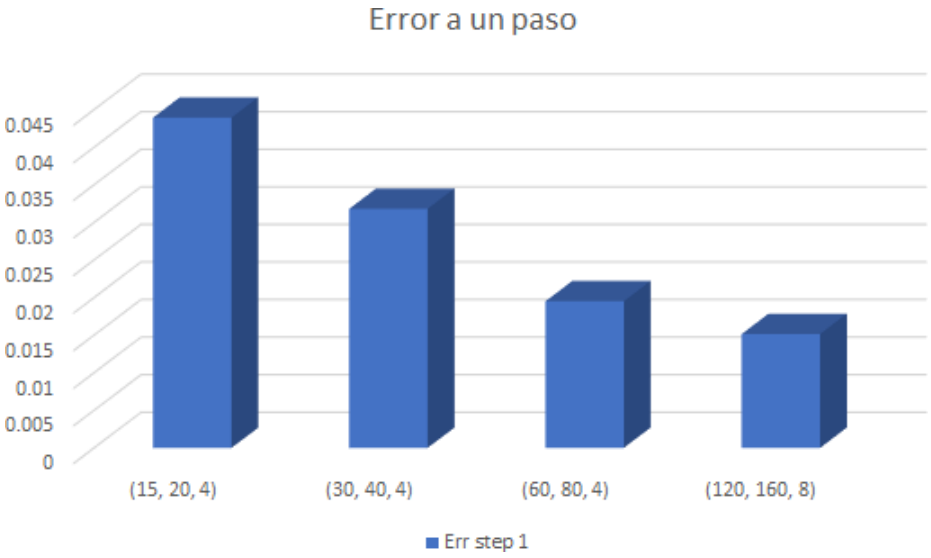


Fig. 8. Error en diferentes niveles de codificación.

La información utilizada es de tipo ”Geographic Information System”(GIS) mediante un servicio ”Web Map Service”(WMS). Este servicio provee imágenes de mapas georeferenciados que contiene la información limpia de delimitaciones. Cada archivo WMS contiene una capa del Monitor de Sequía basada en los shapefiles limpios y re-proyectados a la proyección WGS84 [15]. La Figura 2 muestra un ejemplo de la representación de la sequía en EU.

3.2. Preprocesamiento de las imágenes

Cuando se habla de pronósticos, se habla de reconocimiento de patrones. Estas técnicas engloban una serie de problemas que las hace tener algún grado de dificultad para su implementación. Muchas de las dificultades básicas del aprendizaje máquina se derivan principalmente del uso de grandes cantidades de entradas. De manera que, las tareas se vuelven exponencialmente más difíciles a medida de que el número de entradas aumenta.

A este problema se le conoce como la “Maldición de la dimensionalidad” [8]. En este caso, las imágenes originales son de 480×640 en RGB lo cual da un total de 921,600 valores por cada una de las imágenes del conjunto de datos. Esto representa un problema, pues el manejo de esta gran cantidad de valores requerirá tanto mayor poder de procesamiento como tiempo de ejecución. Para evitar este problema, se utilizan imágenes con un conjunto de colores diferente.

Al utilizar las imágenes con una gama de colores tipo escala de grises maneja un solo canal de color, lo cual reduce la cantidad de valores por imágenes de 307,200 valores. Sin embargo, la cantidad de colores que tiene que aprender a diferenciar los modelos de aprendizaje máquina sigue siendo grande.

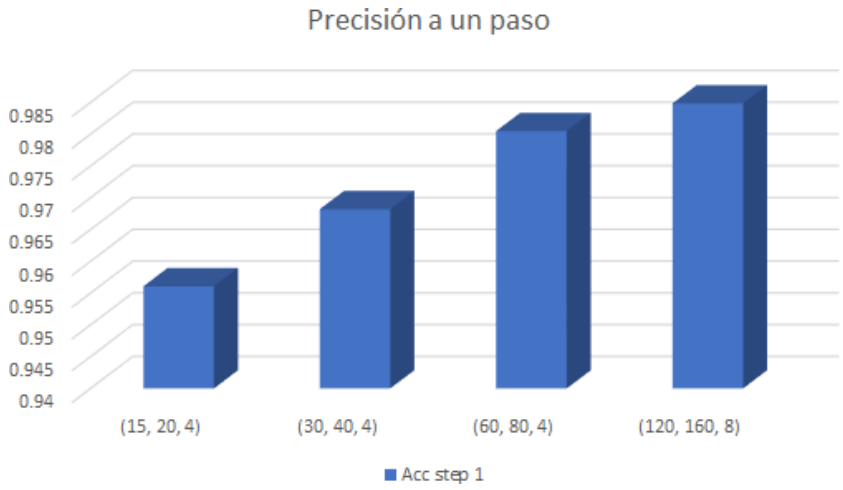


Fig. 9. Precisión en diferentes niveles de codificación.

Como la escala de grises maneja un total de 255 valores diferentes, el modelo tiene que ser capaz de identificar correctamente toda esta gama de colores. Para reducir esto, se utiliza una escala binaria de colores, que transforma las imágenes a solo 2 valores diferentes, blanco o negro. De esta manera, las imágenes respetan otro tipo de información, hay sequía o no hay sequía. La Figura 3 muestra una comparación con la imagen original y la imagen transformada en color monocromático.

3.3. Codificación de las imágenes

En [8] describen a un autoencoder como un tipo de red neuronal que trabaja sin etiquetas y que pertenece a la clase del aprendizaje no supervisado. Este tipo de red contiene una capa de entrada que coincide con las dimensiones de la imagen, o vector de entrada, junto con capas ocultas las cuales reducen la dimensionalidad, seguido de una capa de salida con la mismas dimensiones que la entrada.

El objetivo de esta arquitectura es codificar y decodificar la información introducida en la capa de entrada, la imagen original, en la capa de salida, donde en el proceso de codificación se reduce la dimensionalidad de la imagen, y en la decodificación esta se reconstruye. La Figura 4 muestra la arquitectura de un autoencoder. Tal como se muestra en la Figura 4 un autoencoder esta constituido por dos partes: el codificador, que se encarga de reducir la dimensionalidad de la información; y el decodificador, que se encarga de aumentar la dimensionalidad hasta el punto original.

En este trabajo se utilizan estas dos partes del autoencoder para reducir la dimensionalidad de la información y que sea mucho más fácil realizar un pronóstico de las sequías. Los autoencoders son un conjunto de capas de Convolución y Max pooling, donde las capas de Max pooling se encarga de reducir apropiadamente las primeras dos dimensiones, y las capas de convolución define la tercera dimensión de las muestras. Dependiendo del número de conjuntos de capas que se utilicen es la dimensión resultante en la capa intermedia.

Fig. 10. Imagen pronosticada con forma de codificación (15, 20, 4).

La muestra comprimida por este modelo de red neuronal es en realidad una representación de la muestra original pero con las dimensiones reducidas. De esta manera, en este trabajo se utiliza el conjunto de datos codificados para reducir la necesidad de poder y tiempo computacional. La Tabla 1 muestra el conjunto de capas Convolución-Maxpooling utilizadas, las dimensiones de las imágenes que se obtienen después de la codificación y el total de datos por imagen resultante de esta codificación.

3.4. Generación de series de tiempo

En esta sección se presenta la transformación de las imágenes codificadas a vectores de series de tiempo. Cada píxel de un imagen codificada representa una serie de tiempo independiente. De modo que, al juntar los píxeles de todas las imágenes se obtiene múltiples conjuntos de series de tiempo. Donde cada uno de estos conjuntos de datos pueda ser utilizados como información de entrada para un solo modelo.

De esta manera, cada uno de los conjuntos de datos es tratado como una serie de tiempo independiente, con el cual se realiza un proceso común de pronóstico de series de tiempo. Así cada uno de los modelos que hayan sido entrenados pronosticará una parte de una imagen codificada que represente el siguiente paso en el tiempo. De tal manera que, la imagen codificada pase por el decodificador y se genere una imagen del siguiente paso en la secuencia.

Para realizar este proceso el primer paso consiste en transformar todo el conjunto de datos de múltiples dimensiones a un solo vector de una dimensión. Posteriormente juntar cada uno de estos vectores de manera adyacente, y realizar una transpuesta a la información para una mejor manipulación. De esta manera cada una de las filas del nuevo conjunto de datos obtenido, representara una parte de cada una de las imágenes, osea una secuencia de series de tiempo. La Figura 5 muestra un ejemplo de los pasos a seguir de este proceso.

3.5. Entrenamiento distribuido

Debido principalmente a que cada una de las series de tiempo antes descritas va a ser utilizada por un modelo diferente, serán necesarios miles de modelos de aprendizaje máquina para realizar los pronósticos.

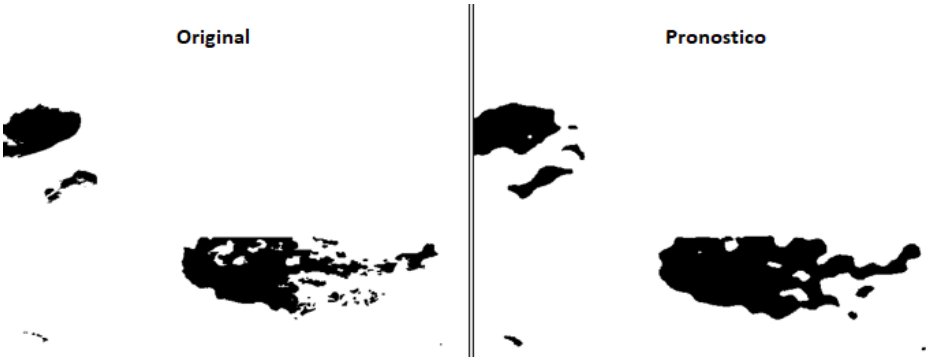


Fig. 11. Imagen pronosticada con forma de codificación (30, 40, 4).

Lo anterior representa una gran cantidad de tiempo invertido si fuera a ser realizado por una sola máquina. En este trabajo se opto por utilizar una técnica de sistemas distribuidos donde todos estos entrenamientos serán realizados en múltiples computadoras al mismo tiempo.

En [2] describen que las aplicaciones de sistemas distribuidos están compuestas por múltiples aplicaciones diferentes que se ejecutan en máquinas diferentes, o muchas replicas ejecutándose a través de diferentes máquinas, todas comunicándose juntas para implementar un sistema.

Utilizando un modelo maestro-esclavo, se distribuyen las responsabilidades entre estos nodos. En el trabajo [6] se define este método se como un modelo simple basado en una política de p esclavos a p conjuntos, donde los individuos son separados en p conjuntos iguales y enviados a p esclavos para su evaluación.

Según el trabajo [6] se asume que la arquitectura maestro-esclavo sigue un paradigma cliente-servidor donde las conexiones se cierran después de cada solicitud. De esta manera, un esclavo (cliente) se conecta a un maestro (servidor) para solicitar un trabajo, el maestro responde enviando un conjunto de tareas que son definidas por el maestro.

La Figura 6 muestra una representación de esta metodología. Durante la ejecución, las tareas del maestro corresponden a la generación, entrenamiento y validación de un autoencoder, codificación del conjunto de datos, creación del conjunto completo de series de tiempo, partición equitativa de cantidad de entrenamiento por esclavo, generación de pronósticos y decodificación del pronóstico.

Las tareas del esclavo corresponden al manejo de lectura de parámetros de entrenamiento, que son: tamaño de ventana de series de tiempo, índice inicial y final de los conjuntos de datos, y la carpeta de almacenamiento de modelos. Generación de cada una de las series de tiempo, creación y entrenamiento de los modelos de pronóstico, modelo de dos capas LSTM y dos ANN y una capa de salida, y guardado del modelo entrenado.

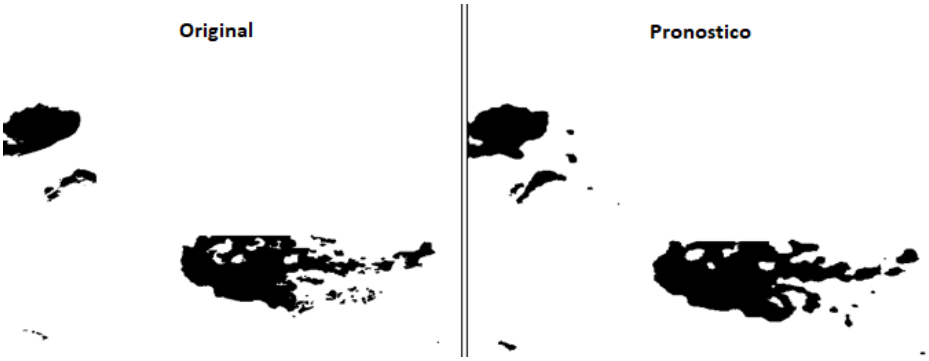


Fig. 12. Imagen pronosticada con forma de codificación (60, 80, 4).

4. Resultados

Para probar la efectividad de la metodología propuesta, fueron realizados diferentes experimentos. Los experimentos fueron realizados con dos esclavos en un solo equipo de computadora, en el lenguaje de programación python, con las siguientes características: CPU Intel Core i7-8700, 16 GB de RAM y dos NVIDIA GeForce RTX 2080 TI. Cada esclavo utiliza exclusivamente una sola GPU.

Ambos esclavos utilizan el mismo modelo de red neuronal, y ese mismo modelo es utilizado para cada una de las series de tiempo, el modelo consiste en: Una capa de entrada de tipo LSTM, una capa oculta de tipo LSTM, dos capas ocultas de tipo ANN y una capa de salida.

Todas las capas utilizan la función de activación “elu”, el modelo utiliza la función de perdida Error Cuadrático Medio (MSE) y función de optimización rmsprop”. Cada una de las redes son entrenadas con un Batch Size de 64 por 150 épocas con un Early Stopping con un valor de paciencia de 15.

La Tabla 2 muestra una comparativa con las diferentes características de los experimentos realizados. Donde la primera columna muestra la forma de compresión del experimentos, la segunda columna muestra el total de redes neuronales implementadas y la tercera columna el tiempo total que llevo realizar el experimento.

A partir de la cuarta columna se muestran los valores del entrenamiento, estos son el valor de perdida del autoencoder, y los valores de precisión y error de las imágenes pronosticadas del paso uno y paso dos. Los valores de precisión y perdida de las imágenes fueron obtenidos de la Ecuación 1 y 2 correspondientemente:

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{m} 1 - |\,\text{original}_{ij} - \text{pronostico}_{ij}\,|}{nm}, \tag{1}$$

$$\frac{\sum_{i=1}^{n}\sum_{j=1}^{m} |\,\text{original}_{ij} - \text{pronostico}_{ij}\,|}{nm}, \tag{2}$$

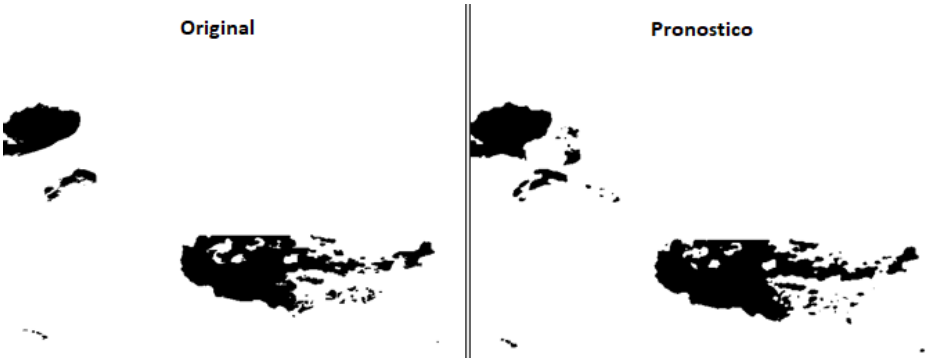


Fig. 13. Imagen pronosticada con forma de codificación (120, 160, 8).

donde n es la anchura de la imagen y m es la altura de la imagen. Con las Ecuaciones 1 y 2 se obtiene el número de coincidencias entre la imagen original y la pronosticada, es decir la comparación de que tantos píxeles son iguales. Para complementar los valores obtenidos en la Tabla 1, la Figura 7 muestra una gráfica del tiempo de entrenamiento. En donde se compara la cantidad de tiempo invertido con la cantidad de modelos a implementar.

En la Tabla 7 se puede observar que al tener un menor nivel de codificación, es decir, una mayor cantidad de modelos a implementar, se requiere una inversión de tiempo mayor a la hora de implementar esta metodología. Por ejemplo, al utilizar un nivel bajo de codificación con el autoencoder, se obtienen dimensiones de codificación muy grandes, como lo muestra la primera fila de la Tabla 1 que tiene las dimensiones (120, 160, 8), dando un total de 153,600 valores.

Como muestra la Figura 7 el tiempo necesario para la implementación de estas dimensiones de codificación es muy alto y no hay una ganancia significativa en cuanto a precisión y error que justifique utilizar niveles tan bajos de codificación, tal como lo muestran la Figura 9 y 8.

Después de pasar por todo el proceso de entrenamiento, cada uno de los modelos puede pronosticar *n* pasos hacia el futuro, es decir un píxel de una imagen. Para después pasar por el autoencoder entrenado y decodificar esa imagen pronosticada. La Figura 10, 11, 12 y 13 muestran una comparación con los pasos futuros originales y los pronosticados.

5. Conclusiones

Para obtener una imagen que represente los siguientes pasos en el tiempo de las sequías, es necesario un proceso extenso. Este proceso requiere diversos pasos que son necesarios para su implementación. El paso que requiere más tiempo de implementación es el entrenamiento de la gran cantidad de modelos necesarios para llevar a cabo el pronóstico. Pero aunque se requiera mucho tiempo, este puede ser reducido si se implementan una cantidad mayor de trabajadores que puedan realizar los pronósticos de manera simultanea en paralelo.

En los resultados mostrados en la Figura 9 y 8, se puede observar que a partir de la codificación con dimensiones (60,80,4) no se obtiene una ganancia significativa de precisión y error que valga la pena para implementación a niveles de codificación más bajos. Mediante el uso del valor de error del entrenamiento del autoencoder, se podría encontrar un punto mínimo de codificación el cual ayudaría a obtener buenos resultados en un tiempo menor.

Los experimentos realizados fueron un gran reto al momento de producir los pronósticos, principalmente debido a la gran cantidad de recursos necesarios para su implementación. Los experimentos más extensos utilizaban alrededor de los 80 Gb de espacio para almacenar todos los modelos entrenados necesarios. Igualmente el tiempo necesario para producir un pronóstico adecuado de las imágenes puede llegar a tomar varios días de ejecución dependiendo de los recursos disponibles.

Si bien aplicar esta metodología puede ser un poco difícil, debido a la necesidad de recursos y la complejidad de la misma, los resultados obtenidos fueron positivos conforme al pronóstico de las sequías. Después de todo el proceso los pronósticos alcanzaron un 98 % de precisión en algunos casos, obteniendo imágenes muy fieles a las originales.

Referencias

1. Brownlee, J.: Deep learning for time series forecasting: Predict the future with MLPs, CNNs and LSTMs in Python. Machine Learning Mastery (2018)
2. Burns, B.: Designing distributed systems: Patterns and paradigms for scalable, reliable services. O'Reilly Media, Inc. (2018)
3. Chow, C.W., Urquhart, B., Lave, M., Dominguez, A., Kleissl, J., Shields, J., Washom, B.: Intra-hour forecasting with a total sky imager at the UC San Diego solar energy testbed. Solar Energy, vol. 85, no. 11, pp. 2881–2893 (2011) doi: 10.1016/j.solener.2011.08.025
4. Cornejo-Bueno, L., Nieto Borge, J. C., Alexandre, E., Hessner, K., Salcedo-Sanz, S.: Accurate estimation of significant wave height with support vector regression algorithms and marine radar images. Coastal Engineering, vol. 114, pp. 233–243 (2016) doi: 10.1016/j.coastaleng.2016.04.007
5. Deb, C., Zhang, F., Yang, J., Lee, S. E., Shah, K. W.: A review on time series forecasting techniques for building energy consumption. Renewable and Sustainable Energy Reviews, vol. 74, pp. 902–924 (2017) doi: 10.1016/j.rser.2017.02.085
6. Dubreuil, M., Gagne, C., Parizeau, M.: Analysis of a master-slave architecture for distributed evolutionary computations. IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), vol. 36, no. 1, pp. 229–235 (2006) doi: 10.1109/tsmcb.2005.856724
7. Gamboa, J. C.: Deep Learning for Time-Series Analysis (2017) doi: 10.48550/ARXIV.1701.01887
8. Joshi, A. V.: Machine learning and artificial intelligence. Springer International Publishing (2020) doi: 10.1007/978-3-030-26622-6
9. Kallis, G.: Droughts: Annual review of environment and resources. vol. 33, no. 1, pp. 85–118 (2008) doi: 10.1146/annurev.environ.33.081307.123117
10. Li, X., Kang, Y., Li, F.: Forecasting with time series imaging. Expert Systems with Applications, vol. 160, pp. 113680 (2020) doi: 10.1016/j.eswa.2020.113680
11. Ma, X., Zhang, J., Du, B., Ding, C., Sun, L.: Parallel architecture of convolutional bi-directional LSTM neural networks for network-wide metro ridership prediction. In: IEEE

Transactions on Intelligent Transportation Systems, vol. 20, no. 6, pp. 2278–2288 (2019) doi: 10.1109/tits.2018.2867042
12. Magnone, L., Sossan, F., Scolari, E., Paolone, M.: Cloud motion identification algorithms based on all-sky images to support solar irradiance forecast. In: Proceedings of the IEEE 44th Photovoltaic Specialist Conference (2017) doi: 10.1109/pvsc.2017.8366102
13. Marquez, R., Pedro, H. T. C., Coimbra, C. F. M.: Hybrid solar forecasting method uses satellite imaging and ground telemetry as inputs to ANNs. Solar Energy, vol. 92, pp. 176–188 (2013) doi: 10.1016/j.solener.2013.02.023
14. Mohamed Sallah, A. H., Tychon, B., Piccard, I., Gobin, A., Van Hoolst, R., Djaby, B., Wellens, J.: Batch-processing of AquaCrop plug-in for rainfed maize using satellite derived fractional vegetation cover data. Agricultural Water Management, vol. 217, pp. 346–355 (2019) doi: 10.1016/j.agwat.2019.03.016
15. United States Drought Monitor: El U.S. Drought Monitor es un producto de la colaboración entre el National Drought Mitigation Center de University of Nebraska-Lincoln, el United States Department of Agriculture y el National Oceanic and Atmospheric Administration (2021) droughtmonitor.unl.edu/Data/GISData.aspx
16. Wei, W. W.: Time series analysis. Oxford University Press (2013) doi: 10.1093/oxfordhb/9780199934898.013.0022
17. Yan, W., Liu, J., Zhang, M., Hu, L., Chen, J.: Outburst flood forecasting by monitoring glacier-dammed lake using satellite images of Karakoram Mountains. Quaternary International, vol. 453, pp. 24–36 (2017) doi: 10.1016/j.quaint. 2017.03.019
18. Yang, L., Gao, X., Li, Z., Jia, D., Jiang, J.: Nowcasting of surface solar irradiance using FengYun-4 satellite observations over China. Remote Sensing, vol. 11, no. 17, pp. 1984 (2019) doi: 10.3390/rs11171984
19. Zhang, W., Chen, E., Li, Z., Zhao, L., Ji, Y., Zhang, Y., Liu, Z.: Rape (Brassica napus L.) growth monitoring and mapping based on radarsat-2 time-series data. Remote Sensing, vol. 10, no. 2. pp. 206 (2018) doi: 10.3390/rs10020206

ISSN 1870-4069

Técnicas de selección de características y su aplicación en el análisis de polen a través de su textura

Arely Guadalupe Sánchez Méndez, Pedro Arguijo,
José Antonio Hiram Vázquez López, Roberto Ángel Melendez Armenta

Tecnológico Nacional de México, campus Misantla,
División de Ingeniería en Sistemas Computacionales, Misantla, Veracruz,
México

aregpesanmen@gmail.com, pedro_arguijo@excite.com, {jahvazquezl,
ramelendeza}@misantla.tecnm.mx

Resumen. Se propone la implementación de métodos de selección de características para la clasificación de conjuntos de datos de polen basándose en un estudio comparativo entre métodos de selección de características de tipo filtro, envoltura y embebido. Se aplicaron diversos algoritmos de selección de cada método en una base de datos de polen, compuesta por 12 clases de polen que se identifican por 244 características; 34 extraídas mediante el análisis de texturas por medio de la matriz de coocurrencia de niveles de gris (GLCM); y el resto, extraídas mediante patrones locales binarios (LBP). Al aplicar selectores de características en cada conjunto de datos e implementar diferentes clasificadores, se determinó el mejor subconjunto de características de acuerdo a los valores de las métricas de clasificación empleadas. Se obtuvieron resultados competitivos al reducir de 244 a 90 características, aumentando la precisión de clasificación de 75.55% a 77.56%.

Palabras clave: Selección de características, polen, clasificación, GLCM, LBP.

Feature Selection Techniques and their Application in Pollen Analysis through their Texture

Abstract. Feature selection for pollen grain classification is proposed, we consider the selection methods of filter, wrapper, and embedded methods. These selection methods were applied to a dataset containing 12 pollen classes and 244 features per grain; 34 extracted by texture analysis using the gray level co-occurrence matrix (GLCM); and the rest extracted using local binary patterns (LBP). The best subset of features was determined according to the values of the classification metrics used in different classifiers. The results obtained indicate that it is feasible to reduce from 244 to 90 features, increasing the classification accuracy from 75.55% to 77.56%.

Keywords: Feature selection, pollen, classification, GLCM, LBP.

1. Introducción

La palinología es una herramienta importante en el estudio de los granos de polen y otras esporas, permite determinar la familia y género a la cual pertenece la prueba bajo estudio [1]. Entre las aplicaciones de ésta se tiene, por ejemplo, el análisis del polen fósil encontrado en el suelo extraído del fondo de antiguos lagos, el mapeo del clima de miles de años atrás, en el área forense permite determinar la geolocalización de sospechosos o el lugar en el cual se llevó a cabo un crimen, en la agronomía ayuda en la predicción y estudio de contaminantes biológicos.

La clasificación del grano de polen es un proceso cualitativo costoso, que implica la observación e identificación de características por parte de expertos altamente calificados llamados palinólogos. Aunque sigue siendo el método más preciso y eficaz, requiere mucho tiempo y recursos, lo que limita el avance de la investigación.

Los sistemas automáticos que permiten el conteo y clasificación de granos de polen suelen extraer diversas características del polen, ya sean de forma, textura, color o una combinación de estas características [2].

Cuando el número de características aumenta, es factible que algunos atributos sean irrelevantes y/o redundantes, lo cual pueden reducir el desempeño de clasificación. Por lo cual es necesario reducir la dimensionalidad del conjunto de características.

La selección de características es parte del proceso de descubrimiento de conocimientos en conjuntos de datos en el que se selecciona un subconjunto de atributos que son más relevantes para la construcción del modelo predictivo sobre el que se esté trabajando [3].

Existen diversos métodos de selección de atributos, entre los más importantes se encuentran: los de tipo filtro, utilizan un criterio de evaluación basándose únicamente en las propiedades de los datos por lo que son independientes del clasificador; los de tipo envoltura, utilizan el desempeño de algún clasificador para evaluar a los subconjuntos de características; los embebidos que interactúan con el algoritmo de aprendizaje a un costo computacional menor que los de envoltura [4].

En este contexto, el presente trabajo de investigación aborda la selección de características en un conjunto de datos de polen que contiene características concatenadas extraídas por matrices de coocurrencia de niveles de gris (GLCM) y patrones locales binarios (LBP), con la finalidad de encontrar el subconjunto de características que mejore el desempeño de clasificación de los tipos de polen del conjunto de datos que se maneja y, además, identificar las características más relevantes, realizando una comparación entre los resultados de clasificación según el selector de características implementado y el tipo de características seleccionadas.

2. Trabajos relacionados

La investigación se centra en trabajos que apliquen aprendizaje automático en el análisis de conjuntos de datos de polen, tomando como tópico principal la selección de características y clasificación, dando a conocer el estado actual de trabajos realizados relacionados con el tema central de esta investigación.

Tabla 1. Tabla de información del conjunto de datos seleccionado.

	ancho	alto	perimeter	...	energiab	homogeneidadb	clase
0	209	189	770.74935	...	0.003718	0.435142	1
1	207	235	790.991991	...	0.003515	0.434229	1
2	156	211	686.239682	...	0.006662	0.513888	1
3	202	191	713.15137	...	0.004909	0.497538	1
4	222	173	796.991991	...	0.004499	0.461106	1
5	195	178	696.381818	...	0.004001	0.464029	1
6	205	232	901.175757	...	0.004772	0.467111	1
7	268	255	1147.1829	...	0.004171	0.517268	1
8	204	224	846.506709	...	0.004	0.432065	1
9	212	183	719.10974	...	0.005408	0.486413	1

Dell et al. realizaron la discriminación y clasificación de polen utilizando microspectroscopía de infrarrojo por transformada de Fourier de infrarrojo medio junto con métodos estadísticos multivariados supervisados y no supervisados [5].

Mitsumoto et al. desarrollaron un método de clasificación y conteo automático de partículas de polen; por sus resultados, sugieren que un sistema de flujo capaz de medir tanto la luz dispersa como la autoflorescencia de partículas podría clasificar y contar los granos de polen en tiempo real [6].

Chica, M., propone un método basado en el procesamiento de imágenes y la clasificación de una clase para rechazar objetos de granos de polen desconocidos. Aplica algoritmos de selección de características. Como resultado obtuvo una precisión general de clasificación del 92.3% [7].

Hernández y Salamanca, redujeron de 11 variables a cuatro componentes principales mostrando correlación entre variables, además lograron la correcta clasificación del 72.4% de las muestras de polen corbícula en función de los tres factores de preponderancia de los componentes principales y su origen geográfico [8].

Hernández et al. utilizaron el método de PCA y sumaron las salidas de 30 redes neuronales, obteniendo una tasa de éxito del 91.67% ± 3.13% en su clasificación [9].

Popescu y Sasu, automatizaron la clasificación de plantas mediante imágenes de granos de polen. Investigaron la efectividad de 11 algoritmos de extracción/selección de características y de 12 clasificadores basados en aprendizaje automático. Encontraron que algunos de los algoritmos de extracción/selección de características especificados y de clasificación exhibieron un comportamiento consistente [1].

Marcos, et al. realizaron una evaluación exhaustiva sobre la utilidad del análisis de textura para la caracterización automatizada de muestras de polen. Aplicaron cuatro métodos de extracción de características de textura: GLCM, filtros log-Gabor (LGF), LBP y momentos discretos de Tchebichef (DTM). Posteriormente, realizaron reducción de dimensionalidad mediante el análisis discriminante de Fisher y realizaron la clasificación con el algoritmo de K vecinos más cercanos. Sus resultados revelaron que LGF y DTM fueron superiores a GLCM y LBP en la clasificación, descubriendo que la combinación de todas las características de textura dio como resultado el rendimiento más alto, con una precisión del 95% [10].

Del Pozo et al. realizaron preprocesamiento, extracción y clasificación de un conjunto de datos de imágenes de polen. Extrajeron un total de 50 características (24 parámetros geométricos y 26 parámetros de textura extraídos mediante GLCM). Sus

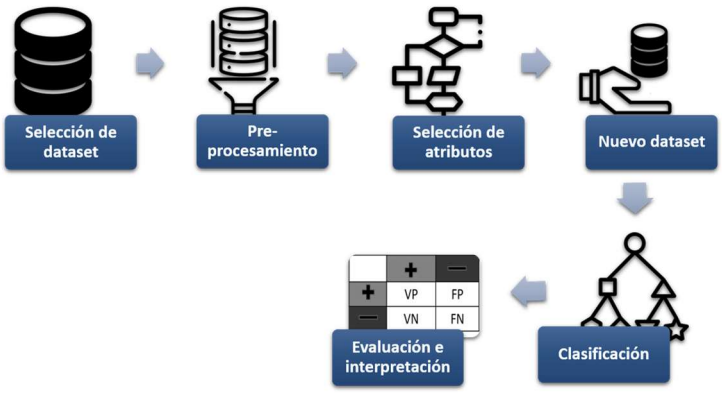


Fig. 1. Diagrama de la metodología implementada.

resultados mostraron tasas de éxito entre 90.54% y 92.81%, destacando la calidad de los parámetros presentados para la clasificación de granos de polen [11].

Goncalves et al. implementaron, ajustaron y probaron una combinación de tres extractores de características: bolsa de palabras visuales (BOW), color, forma y textura (CST) y una combinación de BOW y CST que se llama CST + BOW; cuatro técnicas de aprendizaje automático: dos variaciones de SVM (SMO y C-SVC), un clasificador basado en árbol de decisión (J48) y KNN. Sus resultados mostraron que la tasa de clasificación correcta (CCR) más alta fue del 64% y se logró utilizando CST + BOW y C-SVC [12].

Sevillano y Aznarte, aplicaron tres métodos de clasificación que hacen uso de redes neuronales convolucionales: uno se basa en el aprendizaje por transferencia, otro en extracción de características y un tercero, con un enfoque híbrido, que combina el aprendizaje por transferencia y la extracción de características. Como resultados obtuvieron tasas de clasificación correcta superiores al 97% [13].

3. Materiales y métodos

3.1. Materiales

3.1.1. Selección del conjunto de datos

Se utiliza una base de conocimientos generada como parte del proyecto *Análisis y clasificación de polen*; aprobado por el Tecnológico Nacional de México, el cual consta de 12 clases y cada observación está representada por un vector de 244 características extraídas, de una región de interés de 100×100 pixeles, mediante los métodos de extracción de características de textura: Patrones locales binarios (en inglés Local Binary Patterns, LBP) y Matriz de coocurrencia de escala de grises (en inglés Grey Level Co-ocurrence Matrix, GLCM).

El conjunto de imágenes de las cuales se extrajeron las características de textura indicadas son las reportadas por Tello-Mijares [14]. En total, el conjunto de datos

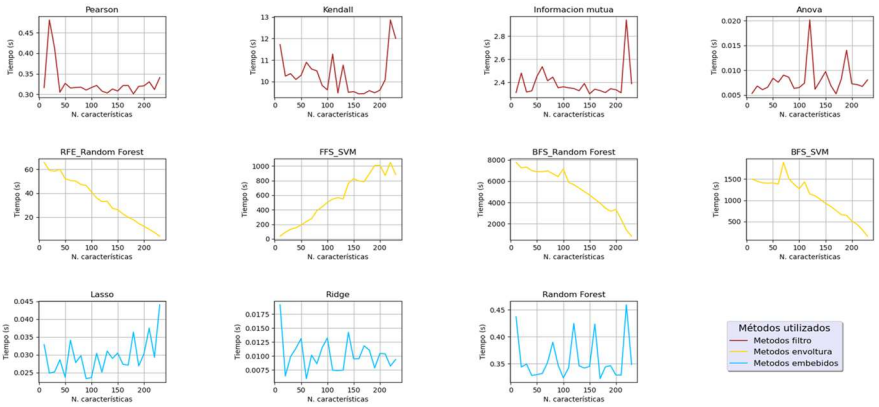


Fig. 2. Tiempos de ejecución de cada selector de características implementado, de acuerdo con el número de características seleccionadas con datos sin normalizar.

utilizado en este trabajo consta de 390 registros, con un promedio de 33 registros por clase. En la Tabla 1 se puede observar una parte del desglose de la información del conjunto de datos, mostrando únicamente 10 registros y seis de las 244 características que lo conforman.

3.1.2. Herramientas

Para el diseño, desarrollo y experimentos del proyecto se utilizaron los siguientes materiales:

- Equipo de cómputo: El análisis de datos, programación y experimentos se realizaron en una computadora con Procesador Intel(R) Core(TM) i7-7500U CPU 2.70 GHz, Microsoft Windows 10 Home Single Language 64-bits, RAM 16GB DDR4 2400 MHz, SSD 128 GB y HDD 1 TB.
- Software: Para el manejo de archivos que contienen a los conjuntos de datos se utiliza Excel. Para la programación y experimentos se utiliza Spyder con Python 3.7.6.
- Librerías de Python:
  - scikit-learn: Cuenta con algoritmos de clasificación, regresión, agrupamiento y reducción de dimensionalidad. Además, es compatible con otras librerías de Python como NumPy, SciPy y matplotlib.
  - Mlxtend: Implementa una variedad de algoritmos y utilidades centrales para máquinas aprendizaje y minería de datos.

3.2. Métodos

La metodología propuesta para el presente trabajo consta de las fases del proceso de descubrimiento de conocimientos en bases de datos, representadas en el esquema de la Fig. 1.

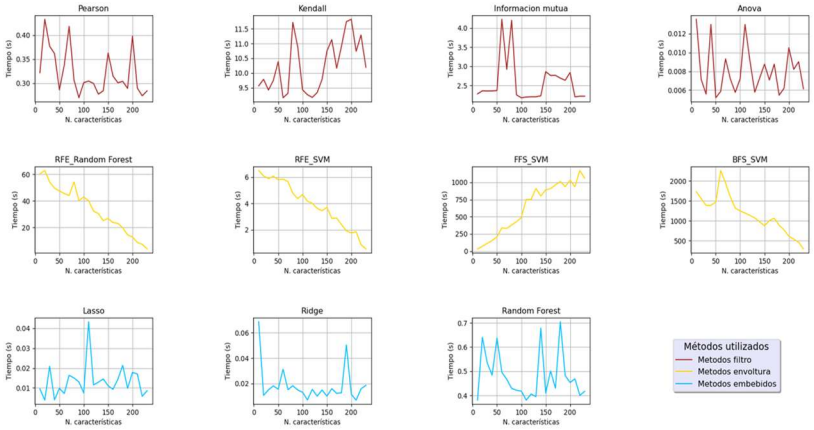


Fig. 3. Tiempos de ejecución de cada selector de características implementado, de acuerdo con el número de características seleccionadas con datos normalizados.

3.2.1 Preprocesamiento

Con el fin de asegurar la utilidad de los datos, se realiza la limpieza de estos. Para experimentos de la investigación se normalizan los datos como técnica de transformación, ya que se realizará una comparación entre resultados utilizando datos sin normalizar y normalizados.

El resultado de esta etapa son dos archivos con extensión .CSV que contienen los datos adecuados para la etapa de selección de atributos relevantes.

3.2.2 Conjuntos de entrenamiento y prueba

Para determinar qué elementos pertenecen a cada uno de los conjuntos se estableció que para el conjunto de entrenamiento se utilice 70% del conjunto original y el 30% restante representa el conjunto de prueba.

3.2.3 Selección de características

Se aplican los algoritmos de selección de características, elegidos de acuerdo con el análisis bibliográfico realizado [15, 16], los cuales se enlistan a continuación.

- De tipo filtro: Coeficiente de correlación (Pearson y Kendall), información mutua y análisis de varianza.
- De tipo envoltura: RFE con RF, FFS con RF, FFS con SVM, BFS con RF y BFS con SVM.
- De tipo embebido: LASSO, Ridge y RF.

Con esto, los experimentos se realizan con los algoritmos de selección de atributos aplicados a cada conjunto de datos.

Tabla 2. Valores de las medidas de clasificación para el conjunto de datos con 90 características seleccionadas del conjunto con datos sin normalizar, de acuerdo al algoritmo de selección implementado.

		SELECTORES DE CARACTERÍSTICAS										
		FILTRO				WRAPPER				EMBEDDED		
		Correlación		IM	Anova	RFE	FFS	BFS		Lasso	Ridge	RF
		Pearson	Kendall			RF	SVM	RF	SVM			
Tiempo seleccionador		0.3104	9.8201	2.3520	0.0063	46.4293	439.0289	6431.0504	1372.8778	0.0233	0.0115	0.3478
CLASIFICADOR	MEDIDAS											
Random Forest	Exactitud	70.94%	75.21%	70.09%	69.23%	75.21%	70.94%	71.80%	70.94%	64.96%	71.80%	76.07%
	Precisión	70.13%	77.24%	74.94%	70.81%	75.37%	73.33%	71.69%	73.63%	63.64%	73.95%	79.22%
	Exhaustividad	67.53%	75.05%	73.19%	72.40%	77.24%	74.35%	74.28%	73.87%	67.97%	75.03%	79.26%
	Tiempo (s)	0.027	0.022	0.024	0.024	0.026	0.030	0.023	0.022	0.022	0.033	0.026
Adaboost SVM	Exactitud	64.53%	68.80%	63.68%	67.95%	64.53%	67.09%	64.53%	67.09%	64.53%	72.22%	64.53%
	Precisión	54.63%	56.25%	53.83%	55.45%	54.63%	53.49%	54.63%	53.49%	54.63%	55.19%	54.63%
	Exhaustividad	58.41%	61.61%	57.76%	61.33%	58.41%	60.86%	58.41%	60.86%	58.41%	63.82%	58.41%
	Tiempo (s)	2170.615	2821.619	2195.823	2202.106	2376.360	2378.517	2611.610	2840.510	2378.508	2384.821	2210.515
Adaboost Perceptron	Exactitud	41.97%	58.21%	64.19%	77.01%	70.17%	52.22%	60.77%	60.77%	57.35%	74.44%	62.48%
	Precisión	31.82%	50.92%	63.78%	83.15%	75.28%	66.80%	60.84%	60.84%	67.64%	76.97%	61.62%
	Exhaustividad	38.80%	53.78%	64.36%	77.72%	69.55%	55.48%	57.81%	57.81%	58.98%	73.85%	63.44%
	Tiempo (s)	11.662	10.486	20.972	22.442	22.344	24.402	18.718	24.990	19.502	22.736	24.304
SVM	Exactitud	44.80%	47.37%	42.24%	61.04%	75.57%	42.24%	44.80%	42.24%	44.80%	49.08%	44.80%
	Precisión	35.22%	45.49%	33.87%	50.37%	78.45%	31.43%	35.22%	31.43%	35.22%	31.89%	35.22%
	Exhaustividad	45.94%	49.08%	39.46%	58.32%	78.52%	42.24%	45.94%	42.24%	45.94%	46.48%	45.94%
	Tiempo (s)	1033.626	1343.628	1045.630	1048.622	1131.600	1132.627	1243.624	1352.624	1132.623	1135.629	1052.626
KNN	Exactitud	65.90%	65.90%	51.37%	60.77%	65.90%	50.51%	65.04%	50.51%	65.04%	64.19%	65.90%
	Precisión	62.96%	60.27%	50.66%	51.13%	62.96%	45.21%	64.31%	45.21%	64.31%	63.16%	62.96%
	Exhaustividad	68.97%	65.33%	52.80%	57.66%	68.97%	53.09%	66.13%	53.09%	66.13%	64.08%	68.97%
	Tiempo (s)	2.046	2.232	2.232	2.418	2.790	2.232	2.418	2.232	2.046	2.418	2.604
Extra Tree	Exactitud	69.23%	69.23%	68.38%	68.38%	**77.78%**	69.23%	70.94%	69.23%	60.68%	74.36%	74.36%
	Precisión	68.29%	72.39%	68.05%	70.09%	**77.56%**	70.69%	72.52%	69.03%	58.51%	73.87%	74.76%
	Exhaustividad	68.87%	66.34%	68.10%	71.47%	**79.79%**	69.77%	74.40%	72.77%	60.87%	78.22%	76.61%
	Tiempo (s)	1.514	1.422	1.439	1.506	**1.642**	1.680	1.437	1.664	1.417	1.754	1.637
Multi-Layer Perceptron	Exactitud	23.93%	29.06%	35.95%	22.22%	66.46%	32.48%	47.66%	36.55%	33.33%	63.25%	29.92%
	Precisión	15.18%	26.26%	39.02%	20.42%	68.18%	22.06%	37.74%	31.58%	26.40%	62.07%	25.54%
	Exhaustividad	20.08%	25.78%	42.59%	22.97%	61.07%	27.81%	43.31%	34.09%	26.38%	64.15%	27.86%
	Tiempo (s)	0.096	0.373	0.227	0.216	0.318	0.237	0.061	0.069	0.153	3.012	0.137
Naive Bayes	Exactitud	38.46%	45.30%	38.46%	52.99%	69.03%	41.03%	39.32%	41.03%	39.32%	70.09%	38.46%
	Precisión	33.98%	40.33%	31.06%	46.41%	63.40%	35.87%	33.58%	35.87%	33.71%	67.29%	33.41%
	Exhaustividad	36.91%	42.56%	34.11%	51.71%	65.64%	38.93%	36.57%	38.93%	36.62%	72.11%	34.44%
	Tiempo (s)	0.008	0.008	0.008	0.008	0.007	0.010	0.008	0.009	0.009	0.009	0.013

3.2.4 Nuevo conjunto de datos

Se utilizan diferentes números de características seleccionadas por cada algoritmo, de manera que se generarán nuevos conjuntos de datos, de acuerdo con el algoritmo y número de atributos seleccionados.

Cada conjunto de datos nuevo debe contener los atributos mejor rankeados y/o seleccionados que resulten de la implementación de los algoritmos de selección de atributos. Por ejemplo, para el conjunto de datos empleado, aplicando el algoritmo de información mutua (método de selección de características), se seleccionan 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20 y 21 características en diferentes ejecuciones del algoritmo, por lo que se obtendrán 12 nuevos conjuntos de datos, cada uno con las características seleccionadas según la cantidad que se indicó antes de ejecutar el algoritmo de selección de características.

3.2.5 Clasificación

Se implementa un modelo de clasificación aplicando los algoritmos que a continuación se enlistan.

Tabla 3. Valores de las medidas de clasificación para el conjunto de datos con 90 características seleccionadas del conjunto con datos normalizados, de acuerdo con el algoritmo de selección implementado.

		SELECTORES DE CARACTERÍSTICAS										
		FILTRO				WRAPPER				EMBEDDED		
		Correlación		IM	Anova	RFE	FFS	BFS		Lasso	Ridge	RF
		Pearson	Kendall			RF	SVM	RF	SVM			
Tiempo seleccionador		0.2697	10.8965	2.2500	0.0058	40.1889	4.3601	427.8767	1318.8640	0.0130	0.0149	0.4211
CLASIFICADOR	MEDIDAS											
Random Forest	Exactitud	60.43%	61.28%	45.90%	55.30%	57.86%	57.01%	55.30%	55.30%	54.44%	57.86%	57.86%
	Precisión	58.10%	58.59%	44.69%	55.50%	57.31%	55.60%	56.99%	56.07%	53.47%	57.02%	56.45%
	Exhaustividad	60.40%	61.56%	48.79%	58.23%	60.16%	58.74%	57.77%	56.77%	53.87%	59.93%	59.93%
	Tiempo (s)	1.739	1.696	1.770	2.089	2.260	2.379	2.904	1.915	3.267	3.436	3.632
Adaboost SVM	Exactitud	63.68%	64.53%	62.82%	62.82%	63.68%	63.68%	61.11%	57.69%	62.82%	63.68%	63.68%
	Precisión	53.19%	55.28%	52.00%	52.06%	53.19%	53.19%	50.96%	51.99%	52.06%	53.19%	53.19%
	Exhaustividad	60.18%	61.57%	60.71%	59.26%	60.18%	60.18%	58.33%	61.61%	59.26%	60.18%	60.18%
	Tiempo (s)	1941.884	1946.567	1935.640	2230.669	2374.281	2240.035	2127.643	1996.519	3101.707	4110.113	3246.880
Adaboost Perceptron	Exactitud	41.37%	38.80%	31.11%	34.53%	48.21%	43.08%	31.97%	32.82%	37.09%	40.51%	39.66%
	Precisión	28.98%	30.15%	20.98%	28.31%	47.75%	39.86%	25.35%	28.79%	24.46%	40.19%	28.19%
	Exhaustividad	38.64%	43.26%	28.33%	37.21%	43.12%	41.36%	32.29%	32.56%	35.90%	38.44%	36.09%
	Tiempo (s)	8.585	6.715	2.040	9.690	29.240	62.050	12.240	2.890	4.675	17.765	22.950
SVM	Exactitud	64.79%	62.22%	51.11%	63.08%	64.79%	64.79%	51.11%	51.11%	63.08%	64.79%	64.79%
	Precisión	54.82%	52.64%	41.01%	48.28%	54.82%	54.82%	40.96%	40.96%	48.28%	54.82%	54.82%
	Exhaustividad	60.42%	59.35%	48.33%	58.49%	60.42%	60.42%	48.33%	48.33%	58.49%	60.42%	60.42%
	Tiempo (s)	32.032	29.120	34.944	37.856	59.696	40.768	36.400	33.488	43.680	52.416	45.136
KNN	Exactitud	53.46%	50.04%	38.93%	48.33%	53.46%	53.46%	51.75%	52.61%	48.33%	53.46%	53.46%
	Precisión	52.75%	49.41%	34.16%	51.87%	56.62%	56.62%	48.49%	47.90%	52.16%	56.62%	56.62%
	Exhaustividad	48.11%	44.28%	36.01%	45.72%	48.17%	48.17%	49.62%	51.86%	45.72%	48.17%	48.17%
	Tiempo (s)	0.938	0.871	1.072	0.871	1.072	1.206	0.871	0.938	1.206	1.206	1.273
Extra Tree	Exactitud	**69.83%**	68.97%	47.61%	57.86%	57.86%	54.44%	55.30%	56.15%	58.72%	53.59%	54.44%
	Precisión	**70.06%**	70.06%	48.19%	59.62%	58.95%	55.08%	57.47%	57.45%	57.09%	52.57%	53.78%
	Exhaustividad	**71.83%**	72.52%	47.58%	59.03%	58.17%	56.02%	57.08%	58.13%	56.94%	55.38%	55.41%
	Tiempo (s)	**1.362**	1.403	1.711	1.833	2.520	1.875	1.570	1.447	2.270	2.114	2.561
Multi-Layer Perceptron	Exactitud	42.22%	43.93%	35.38%	43.08%	43.08%	43.08%	33.68%	35.38%	43.08%	42.22%	44.79%
	Precisión	28.08%	30.51%	21.28%	28.88%	28.81%	30.10%	22.66%	23.47%	31.21%	28.57%	29.75%
	Exhaustividad	37.95%	40.05%	28.33%	38.79%	38.92%	38.73%	31.40%	30.47%	38.79%	38.28%	40.21%
	Tiempo (s)	0.528	0.478	0.177	0.620	0.620	0.733	0.562	0.224	0.858	0.789	0.996
Naive Bayes	Exactitud	60.88%	62.59%	49.77%	50.62%	56.61%	56.61%	55.75%	49.77%	55.75%	56.61%	56.61%
	Precisión	53.92%	56.43%	43.61%	40.75%	46.85%	46.85%	41.96%	39.20%	46.11%	46.85%	46.85%
	Exhaustividad	61.04%	63.21%	49.41%	51.53%	56.37%	56.37%	55.92%	49.33%	55.99%	56.37%	56.37%
	Tiempo (s)	0.006	0.008	0.009	0.006	0.008	0.012	0.010	0.008	0.016	0.007	0.014

- RF.
- AdaBoost.
- MLP.
- NB.
- KNN.
- Extra trees classifier.
- SVM.

3.2.6 Evaluación

Se analizan los resultados que se obtuvieron utilizando las medidas de precisión de los algoritmos, ofreciendo argumentos y valoraciones que demuestren la factibilidad y precisión de la clasificación.

4. Experimentos y resultados

4.1. Selección de características

Se utiliza el conjunto de datos sin ser normalizados y con datos normalizados. Al implementar cada uno de los algoritmos seleccionados para realizar la selección de

variables, uno de los argumentos de entrada en el que todos coinciden es el número de características a ser seleccionadas.

Para poder identificar un número óptimo de características seleccionadas, se han generado conjuntos de datos de 10 hasta 230 características, considerando que el conjunto de datos está conformado por 244 características. Posteriormente se realiza la clasificación de cada uno de los conjuntos de datos generados y la comparativa de los valores métricos para cada modelo.

De acuerdo con el tiempo de ejecución de los diferentes algoritmos de selección de características implementados en el conjunto de datos sin normalizar, se presentan las gráficas de la Fig. 2 donde se muestra el tiempo de ejecución (en segundos) en función de la cantidad de características seleccionadas (de 10,20,30,40, ..., 230). Las gráficas se analizan de acuerdo con el tipo de algoritmo: de color marrón los de tipo filtro, de amarillo los de envoltura y de azul los embebidos.

Los tiempos de ejecución para los métodos filtro van de 0.0052 a 12.8686 segundos; los de envoltura van de 4.0287 a 7774.5830 segundos; los de los embebidos su mayor tiempo de ejecución fue de 0.4594 y el menor fue de 0.0059. De los datos anteriores, de manera general, se puede identificar que los algoritmos con mayor tiempo de ejecución son los de tipo envoltura.

El algoritmo de selección de características que requirió más tiempo de ejecución, con 7774.583 segundos fue el de BFS implementando RF con 10 características. El algoritmo de selección de características que empleó menos tiempo de ejecución, con 0.0052 segundos, fue el de ANOVA con 170 características seleccionadas.

En la Fig. 3, el gráfico representa los tiempos de ejecución de los algoritmos de selección de características implementados en el conjunto de datos normalizado. Los tiempos de ejecución para los métodos filtro van de 0.0051 a 11.8254 segundos; los de envoltura van de 0.5302 a 2266.8091 segundos; los de los embebidos su mayor tiempo de ejecución fue 0.7050 y el menor fue de 0.0038 segundos.

De los datos anteriores, de manera general, se puede identificar que los algoritmos con mayor tiempo de ejecución son los de tipo envoltura. El algoritmo de selección de características que requirió más tiempo de ejecución, con 2266.8091 segundos fue el de BFS implementando SVM con 60 características. El algoritmo de selección de características que utilizó menos tiempo de ejecución, con 0.0038 segundos, fue el de LASSO con 20 características seleccionadas (con BFS-SVM seleccionando 20 características le llevó un tiempo de ejecución de 1560.9076 segundos).

Los resultados de tiempo de ejecución serán mejor valorados al momento de comparar los resultados de las medidas obtenidas en la clasificación de los datos considerando el número de características seleccionadas e implementando diversos algoritmos de clasificación.

4.2. Clasificación

Los resultados correspondientes a la clasificación de los datos sin normalizar y sin realizar la selección de características. El mayor porcentaje de exactitud en la clasificación de los datos ha sido del 75.21% obtenido por el clasificador RF.

Al implementar cada algoritmo de selección de características elegido, se crearon 23 nuevos conjuntos de datos, de acuerdo con el número de características seleccionadas

(desde 10 hasta 230 características). Seguido de esto, se realizó la clasificación de los datos, utilizando los algoritmos de clasificación seleccionados.

De cada nuevo conjunto de datos creado según el número de características seleccionadas con datos no normalizados, se procedió a realizar la clasificación de estos, aplicando los ocho clasificadores seleccionados para este fin. Se eligió al clasificador que diera mejores resultados en sus medidas de clasificación definidas, incluso el tiempo de ejecución al realizar la selección de características.

Al realizar la clasificación de los datos, el mayor porcentaje en exactitud, precisión y exhaustividad con 77.78%, 77.56% y 79.79%, respectivamente, se obtuvo cuando se seleccionaron 90 características.

En la Tabla 2 se presentan los valores de las medidas de clasificación para el conjunto de datos con 90 características seleccionadas del de datos no normalizados; el clasificador con mejor desempeño fue Extra Tree Classifier cuando se realizó la selección de características mediante el método de envoltura RFE implementado con Random Forest, el clasificador tuvo un tiempo de ejecución de 1.64 segundos y el del seleccionador de características fue de 46.4293 segundos (un tiempo de ejecución menor a un minuto, lo cual no es tan lento en comparación con los otros métodos de envoltura que llegaron a tardar más de dos horas).

Cuando se realizó la clasificación del conjunto de datos normalizado y sin realizar la selección de características. El mayor porcentaje de exactitud en la clasificación de los datos ha sido del 75.72% obtenido por el clasificador Extra Tree Classifier.

Al implementar los algoritmos de clasificación a los 23 nuevos conjuntos con datos normalizados, después de realizar la selección de características, se eligió al clasificador que diera mejores resultados en sus medidas de clasificación definidas, incluso el tiempo de ejecución al realizar la selección de características. Al realizar la clasificación de los datos, el mayor porcentaje en exactitud, precisión y exhaustividad con 69.83%, 70.06% y 71.83%, respectivamente, se obtuvo cuando se seleccionaron 90 características.

En la Tabla 3 se presentan los valores de las medidas de clasificación para el conjunto de datos con 90 características seleccionadas del conjunto con datos normalizados. El clasificador con mejor desempeño fue Extra Tree Classifier cuando se realizó la selección de características mediante el método filtro de coeficiente de correlación de Pearson, el clasificador tuvo un tiempo de ejecución de 1.362 segundos y el del seleccionador de características fue de 0.2697 segundos.

5. Conclusiones y trabajo a futuro

Se presenta la implementación de tres métodos de selección de características (filtro, envoltura y embebido) aplicando diferentes algoritmos de cada tipo. Con el objetivo de demostrar que para un conjunto completo de características iniciales existe un subconjunto de características que mejore o mantenga el desempeño de clasificación de los tipos de polen que se manejan en el conjunto de datos con características extraídas mediante el GLCM y LBP.

Se aplicaron cuatro algoritmos de selección de características de tipo filtro, tres de tipo envoltura y tres de tipo embebido. La mejor clasificación con los 244 atributos se obtuvo con el clasificador Random Forest con 75.21%, 75.55% y 77.45% de exactitud,

precisión y exhaustividad, respectivamente. Cuando se aplicaron los selectores de características, la mejor clasificación se obtuvo cuando se manejó un subconjunto de 90 características extraídas mediante el algoritmo RFE con Random Forest de tipo envoltura y se utilizó el clasificador Extra Tree; los valores de clasificación fueron de 77.78% en exactitud, 77.56% en precisión y 79.79% en exhaustividad.

Se puede concluir aquí que la selección de características mejoró los valores de las métricas de clasificación implementadas. Es importante mencionar que se obtuvieron valores muy cercanos e incluso mayores a la clasificación obtenida al aplicar la selección de características en este conjunto de datos cuando se seleccionaron las características mediante el algoritmo de correlación con el coeficiente de Kendall (método filtro) y los algoritmos Ridge y Random Forest que son de tipo embebido, además de que también funcionó como buen clasificador el Random Forest.

Finalmente se puede decir que el resultado de clasificación mejora si se utilizan métodos de selección que proporcionen buenos subconjuntos de características. Los resultados de los experimentos muestran que los algoritmos de tipo filtro, embebido y de envoltura funcionan bien con este tipo de datos, aunque los de tipo embebido puedan mejorar un poco los porcentajes de correcta clasificación, tienen el punto en contra de que llevan más tiempo de ejecución.

Por ejemplo, cuando se seleccionaron 90 características con valores de clasificación similares, el algoritmo RFE-RF (de tipo envoltura) le tomó 46.42 segundos, mientras que con el algoritmo de correlación con el coeficiente Kendall (de tipo filtro) fue de 9.82 segundos y con el de Random Forest(embebido) le tomó 0.34 segundos.

Con la finalidad de analizar la posibilidad de obtener mejores resultados de clasificación se realizó la normalización de los datos, pero al comparar los resultados de clasificación con los datos sin normalizar, se concluye que, el realizar la normalización de los datos disminuyó el porcentaje de clasificación.

Como trabajo futuro se propone realizar un sistema automatizado que permita el conteo de polen implementando el proceso de selección de características y considerando cuáles selectores y clasificadores funcionaron mejor en este trabajo de investigación, con la finalidad de mejorar la clasificación y correcta identificación de polen.

Referencias

1. Popescu, M. C., Sasu, L. M.: Feature extraction, feature selection and machine learning for image classification: A case study. In: Proceedings of International Conference on Optimization of Electrical and Electronic Equipment (OPTIM), IEEE, pp. 968–973 (2014) doi: 10.1109/OPTIM.2014.6850925
2. Damián, M. R.: Clasificación automática y recuento de granos de polen a partir de imágenes digitales de microscopía óptica. Tesis Doctoral, Universidade de Vigo (2005)
3. Kotsiantis, S.: Feature selection for machine learning classification problems: A recent overview. Artificial Intelligence Review, vol. 42, no 1, pp. 157–176 (2011) doi: 10.1007/s10462-011-9230-1
4. Hall, M. A.: Correlation-based feature selection for discrete and numeric class machine learning. In: Proceedings of the Seventeenth International Conference on Machine Learning, pp. 359–366 (2000)
5. Dell'Anna, R., Lazzeri, P., Frisanco, M., Monti, F., Campeggi, F. M., Gottardini, E., Bersani, M.: Pollen discrimination and classification by Fourier transform infrared (FT-IR)

microspectroscopy and machine learning. Analytical and bioanalytical chemistry, vol. 394, pp. 1443–1452 (2009) doi: 10.1007/s00216-009-2794-9
6. Mitsumoto, K., Yabusaki, K., Aoyagi, H.: Classification of pollen species using autofluorescence image analysis. Journal of bioscience and bioengineering, vol. 107, no 1, pp. 90–94 (2009) doi: 10.1016/j.jbiosc.2008.10.001
7. Chica, M.: Authentication of bee pollen grains in bright-field microscopy by combining one-class classification techniques and image processing. Microscopy research and technique, vol. 75, no. 11, pp. 1475–1485 (2012) doi: 10.1002/jemt.22091
8. Hernández, J., Salamanca, G.: Métodos estadísticos para la clasificación fisicoquímica de polen corbicular de la zona altoandina de Boyacá, Colombia (2012)
9. Hernández, J. A., González, C. M. T., Rivas, J. T., Mora, F. M., Huertas, O. S., Bogantes, M. R., Chavez, L. S.: Sistema de detección y clasificación automática de granos de polen mediante técnicas de procesado digital de imágenes. Uniciencia, vol. 27, no. 1, pp.59–73 (2013)
10. Marcos, J. V., Nava, R., Cristóbal, G., Redondo, R., Escalante-Ramírez, B., Bueno, G., Déniz, O., González-Porto, A., Pardo, C., Chung, F., Rodriguez, T.: Automated pollen identification using microscopic imaging and texture analysis. Micron, vol. 68, pp. 36–46 (2015) doi: 10.1016/j.micron.2014.09.002
11. del Pozo-Banos, M., Ticay-Rivas, J. R., Alonso, J. B., Travieso, C. M.: Features extraction techniques for pollen grain classification. Neurocomputing, vol. 150, pp. 377–391 (2015) doi: 10.1016/j.neucom.2014.05.085
12. Barbosa, A., Silva, J., Goncalves, G., Pascoli, M., Pott, A., Hirosi, M., Pistori, H.: Feature extraction and machine learning for the classification of Brazilian savannah pollen grains. PlOS ONE, vol. 11, no. 6, pp. 1–20 (2016) doi: 10.1371/journal.pone.0157044
13. Sevillano, V., Aznarte, J. L.: Improving classification of pollen grain images of the POLEN23E dataset through three different applications of deep learning convolutional neural networks. PLOS ONE, vol. 13, no. 9 (2018) doi: 10.1371/journal.pone.0201807
14. Tello-Mijares, S., Flores, F.: A novel method for the separation of overlapping pollen species for automated detection and classification. Computational and mathematical methods in medicine (2016) doi: 10.1155/2016/5689346
15. Bolón-Canedo, V., Sánchez-Maroño, N., Alonso-Betanzos, A.: Feature selection for high-dimensional data. Cham, Springer International Publishing (2015)
16. Bolón-Canedo, V., Alonso-Betanzos, A.: Recent advances in ensembles for feature selection. Berlin, Heidelberg, Springer, vol. 147 (2018)

ISSN 1870-4069

Sistema de control de acceso mediante identificación facial usando aprendizaje profundo

José Misael Burruel Zazueta[1], Hector Rodríguez Rangel[1], Gloria Ekaterine Peralta Peñuñuri[1], Víctor Alejandro González Huitrón[1], Luis Alberto Morales Rosales[2]

[1] Tecnológico Nacional de México, División de Posgrado, Maestría en Ciencias de la Computación, México

[2] Universidad Michoacana de San Nicolás de Hidalgo, Consejo Nacional de Ciencia y Tecnología, México

{jose_burruel, hrodriguez, gperalta, victor.gonzalez}@itculiacan.edu.mx, lamorales@conacyt.mx

Resumen. Existen diversas tecnologías empleadas para el desbloqueo automático de puertas, algunas de estas técnicas utilizan sistemas biométricos para analizar corporalmente al usuario y de esa forma asegurar que es una persona deseable. En este articulo, se propone un método de identificación facial para su implementación en sistemas de cerraduras biométricos. Este método esta basado principalmente en la utilización de una red neuronal convolucional desarrollada por Google llamada Facenet. Así mismo, se propone un sistema embebido de cerradura electrónica para implementarse mediante el identificador facial. A diferencia de otros proyectos documentados, el sistema de identificación facial logra más del 99 % de tasa de reconocimiento correcto y un alto rendimiento.

Palabras clave: Cerradura biométrica, aprendizaje profundo, reconocimiento facial, red neuronal convolucional, facenet.

Access Control System through Facial Identification Using Deep Learning

Abstract. There are various technologies used for the automatic unlocking of doors, some of these techniques use biometric systems to analyze the user's body and thus ensure that he is a desirable person. In this article, a facial identification method is proposed for its implementation in biometric lock systems. This method is mainly based on the use of a convolutional neural network developed by Google called Facenet. Likewise, an embedded electronic lock system is

proposed to be implemented through the facial identifier. Different from other documented projects, the facial identification system achieves more than 99% correct recognition rate and high performance.

Keywords: Biometric lock, deep learning, facial recognition, convolutional neural network, facenet.

1. Introducción

El robo a casa habitación ha estado en la quinta posición de incidencia delictiva en México por muchos años, incluso por encima de robo total de automóvil, según cifras del INEGI [6]. Poca vigilancia, sistemas de seguridad deficientes y otros factores, han contribuido a que los robos en propiedades privadas tengan tasas de incidencia alarmantes. La seguridad representa la protección de nuestra vida y activos. Garantizar la seguridad de las personas y sus cosas de valor es muy importante para prevenir la manipulación ilegal.

Por lo tanto, centrarse principalmente en la seguridad de la cerradura de la puerta o la seguridad de la puerta es muy importante para evitar problemas adicionales en el área monitorizada [10]. Las cerraduras de puertas son sistemas que contribuyen al resguardo de la seguridad de cualquier edificación, estos sistemas pretenden evitar el ingreso de personas no autorizadas a zonas especificas. En [5] se mencionan algunas tecnologías empleadas en la seguridad de las cerraduras de puertas (e.g. mecánica, biométrica, por contraseña, entre otras).

El aprendizaje profundo permite que los modelos computacionales que se componen de múltiples capas de procesamiento, aprendan representaciones de datos con múltiples niveles de abstracción. Estos métodos han mejorado de forma drástica el estado de la técnica en reconocimiento de voz, reconocimiento de objetos visuales, detección de objetos y muchos otros dominios [16].

El uso del aprendizaje profundo ha generado múltiples técnicas que se han popularizado exponencialmente en la ultima década debido a los altos estándares de calidad alcanzados. La seguridad habitacional no es la excepción, ya que se han realizado múltiples esfuerzos por mejorar los sistemas de seguridad basados en lecturas biométricas mediante aprendizaje profundo.

La principal motivación del presente trabajo es desarrollar un sistema de cerradura con altos estándares de calidad y con una mínima interacción con el usuario. Se busca reducir la relación del ser humano con el sistema para evitar errores por parte de los usuarios, el ejemplo mas común es extraviar objetos de manipulación de las cerraduras como llaves o tarjetas electrónicas, esto también supondría un problema de seguridad mayor si una persona no autorizada obtiene estos objetos.

El proyecto en general propone un sistema de cerradura biométrico basado en identificación facial. La realización del sistema de cerradura biométrico comprende diferentes procesos como son:

1. La identificación facial, donde el sistema identifica al sujeto a partir de su rostro.

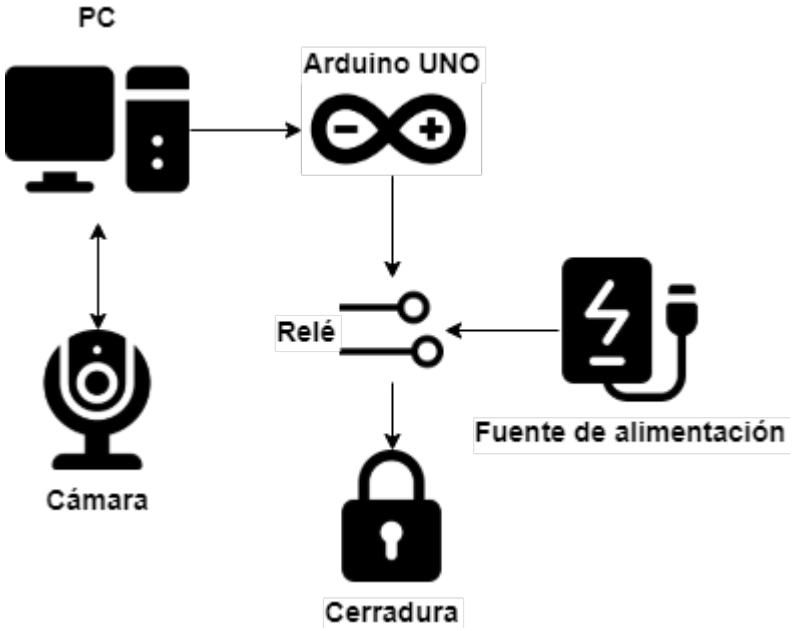


Fig. 1. Diagrama de interacción de los dispositivos de hardware en el sistema identificador facial.

2. La autenticación de la persona, donde se comprueba que no es una fotografía.
3. La apertura de la cerradura utilizando sistemas embebidos.
4. Finalmente el administrador del sistema, donde se gestionan usuarios, roles, etc.

En este trabajo en particular se hizo un enfoque en el proceso de identificación facial utilizando herramientas a partir de imágenes de intensidad (holístico) mediante la utilización de una red neuronal convolucional (CNN). En los resultados se muestran algunas métricas numéricas que evalúan el rendimiento del reconocimiento facial, el cual fue sometido a varias pruebas.

2. Reconocimiento facial

El reconocimiento facial es el proceso de identificar el rostro de una persona relevante mediante un sistema de visión. Ha sido una herramienta crucial de interacción persona-computadora debido a su uso en sistemas de seguridad, control de acceso, videovigilancia, áreas comerciales e incluso se usa en redes sociales como Facebook [4]. El reconocimiento facial parece ofrecer varias ventajas sobre otros métodos biométricos, algunos de los cuales se describen aquí:

Casi todas estas tecnologías requieren alguna acción voluntaria por parte del usuario, es decir, el usuario debe colocar su mano en un escáner para la toma de huellas dactilares o la detección de la geometría de la mano y debe pararse en una posición fija frente a una cámara para la identificación del iris o la retina.

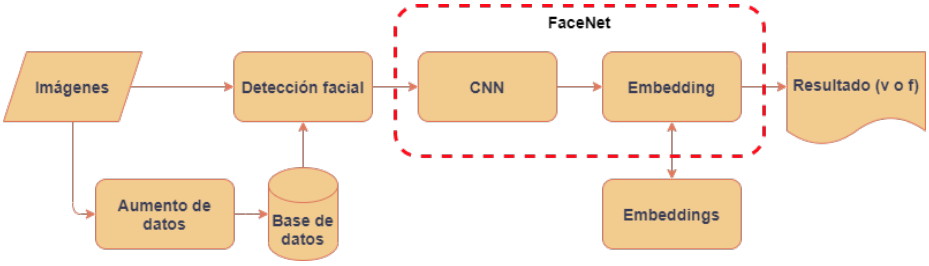


Fig. 2. Diagrama de flujo de información a través del software.

Sin embargo, el reconocimiento facial se puede realizar de forma pasiva sin ninguna acción o participación explícita por parte del usuario, ya que una cámara puede adquirir imágenes faciales a distancia [7].

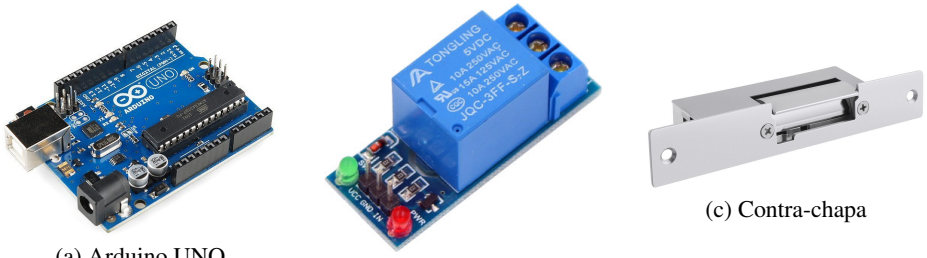
Existen diferentes técnicas para realizar la tarea de reconocimiento facial que se pueden dividir en tres grupos principalmente: 1) métodos que operan con imágenes de intensidad, 2) aquellos que tratan con secuencias de vídeo y 3) aquellos que requieren otros datos sensoriales como información 3D o imágenes infrarrojas [7].

– **Reconocimiento facial a partir imágenes de intensidad**: Los métodos de reconocimiento facial de imágenes de intensidad son dos, el método basado en características y el método holístico:

  • **Basado en características.** En este método, las características locales como ojos, nariz y boca se extraen en primer lugar y sus ubicaciones y estadísticas locales (geométricas y/o de apariencia) se introducen en un clasificador estructural. Un gran desafío para los métodos de este tipo es la restauración"de características, esto es cuando el sistema intenta recuperar características que son invisibles debido a grandes variaciones, p. Ej. Pose de cabeza cuando estamos haciendo coincidir una imagen frontal con una imagen de perfil [12].

  • **Holístico**: A diferencia del método basado en características, este método realiza la tarea de reconocimiento facial utilizando representaciones totales de la imagen, este método se divide en dos grupos diferentes: aproximaciones estadísticas e inteligencia artificial (IA). Las aproximaciones estadísticas realiza el reconocimiento comparaciones de correlación directa entre la cara de entrada y todas las demás caras de la base de datos [7]. Los métodos de IA e basan en la utilización de redes neuronales y aprendizaje maquina para lograr alcanzar altos estándares de calidad en el reconocimiento facial.

– **Reconocimiento facial a partir de secuencias de vídeo**: Un sistema de reconocimiento facial basado en vídeo generalmente consta de tres módulos: uno para detectar el rostro; un segundo para rastrearlo; y un tercero para reconocerlo. La mayoría de estos sistemas eligen unos pocos fotogramas buenos y luego aplican una de las técnicas de reconocimiento de imágenes de intensidad a esos fotogramas para identificar al individuo [7].



(a) Arduino UNO

(b) Relevador

(c) Contra-chapa

Fig. 3. Componentes de hardware utilizados en la cerradura biométrica.

– **Reconocimiento facial a partir de datos sensoriales**: Los métodos de reconocimiento facial a partir de datos sensoriales se empezaron a desarrollar debido a factores de posicionamiento del rostro en la cámara de vídeo, para que los métodos descritos anteriormente funcionen el rostro del usuario debe mostrarse de manera frontal, es decir, en 2D para poder extraer las características del mismo. Con los datos sensoriales es posible obtener imágenes del rostro del usuario en 3D, esto lo podemos dividir en dos técnicas distintas: modelo basado en 3D e infrarrojo. Ambas técnicas utilizan procedimientos bastante parecidos, se diferencian de las tecnologías empleadas para la obtención de la información para procesar.

3. Trabajos relacionados

En el área de cerraduras biométricas, donde características corporales de una persona son la llave de la cerradura, encontramos el trabajo de Baidya et al. [1], el cual propone un sistema biométrico embebido de desbloqueo de cerradura de puerta. El trabajo se basa en Arduino y un lector de huellas dactilares. La placa programable realiza la tarea de procesar la información recibida a través de los sensores, además envía señales de salida para desbloquear la puerta o bien emitir señales de indicación auditiva (buzzer) y visual (matriz de leds 4x4).

En los trabajo de Radzi et al. [13] y Balla et al. [2] desarrollaron sistemas de cerradura biométrica basados en reconocimiento facial, en ambos casos implementado en Raspberry Pi, para el primer sistema utilizaron una red neuronal convolucional llamada AlexNet y en el otro caso se utilizó un sistema de IoT que se comunica con el administrador para autorizar los accesos mediante una aplicación web y/o móvil.

Por otro lado, en el trabajo de Lwin et al. [17], desarrollaron una sistema de cerradura biométrica basado en reconocimiento facial utilizando el método Viola-Jones, realizando la clasificación utilizando la ecuación de distancia euclidiana para identificar al la persona. A diferencia de los trabajos descritos anteriormente, este sistema esta implementado en una PC convencional y en MatLab, si bien los niveles de seguridad son mucho mas deficientes en comparación con otras opciones, este trabajo presume de practicidad y una buena implementación a lo pretendido por los autores.

En el área de reconocimiento facial se han documentado diferentes trabajos en el tema. Como por ejemplo los trabajos de Parkhi et al. [11], Taigman et al.

(a) Original (b) Copia 1 (c) Copia 2

Fig. 4. Resultados del proceso de aumento de datos.

[15], Cao et al. [3] donde utilizan técnicas de aprendizaje profundo para realizar el reconocimiento facial. En [11] realizan el reconocimiento facial de personas importantes del mundo haciendo uso de servicios web gratis.

Probando diferentes arquitecturas como Fisher Vector Faces, DeepFace, Fusion, DeepID-2,3, FaceNet y FaceNet + Alignment; donde la bases de datos utilizada es YouTube Faces Dataset [16]. En [15] dividen el proceso en tres partes: alineación facial, representación, y la tercera parte corresponde a las métricas de verificación. La alineación facial se encarga de detectar las caras en las imágenes, basándose en puntos de referencia que detectan características correspondientes a la forma de la cara.

En la etapa de representación, se encuentran todos los procesos correspondientes a la extracción de las características que diferencian una cara de otra. Para finalmente, en la representación se realizan los entrenamientos a las redes neuronales profundas con el fin de que estas aprendan a distinguir cada una de las características extraídas de las imágenes previamente. En [3] crea una base de datos de más de 3 millones de imágenes faciales clasificadas en más de 9000 identidades diferentes.

Después, utiliza una línea de tiempo acerca de cómo crear una base de datos con características capaces de clasificar imágenes faciales. Posteriormente se continúa creando una plantilla para el conjunto de prueba enfocada en explorar el rendimiento del reconocimiento de pose y edad. Para finalmente, demostrar que el entrenamiento de las CNN con la nueva base de datos mejora significativamente el rendimiento en comparación con el estado del arte. La base de datos utilizada selecciona nombres de personalidades como artistas, políticos, deportistas, etc. disponibles en Google Image Search.

4. Materiales y métodos

Para explicar el desarrollo del sistema de identificación facial propuesto en este documento primero describiremos el hardware utilizado y como se lleva a cabo la comunicación entre los diferentes dispositivos utilizados para el desarrollo de esta propuesta. Continuando con la descripción del software o los procesos que involucran el identificador facial.

4.1. Diseño de hardware

Un sistema de cerradura biométrica otorga acceso a los usuarios autorizados mediante la verificación de sus características físicas o de comportamiento únicas, como

Fig. 5. Comportamiento de las imágenes con el proceso Triplet Loss.

huellas dactilares, reconocimiento facial, reconocimiento de voz, detector de venas, escáner de iris, etc.

Estos sistemas de bloqueo funcionan escaneando los datos biométricos y luego convirtiéndolos en una plantilla numérica que se guardará por primera vez. Luego, la próxima vez que alguien intente acceder a la puerta utilizando sus datos biométricos, se comparará con el valor guardado previamente [10].

El sistema propuesto en este trabajo, utiliza el reconocimiento facial como llave para obtener acceso a un área deseada. En la Figura 1 se observa el diagrama correspondiente de la relación existente entre los componentes físicos que conforman al sistema de cerradura biométrico propuesto.

El proceso comienza en la cámara que se utiliza para dos propósitos elementales en el sistema, el primero es la toma de fotografías para agregar a los usuarios a la base de datos y el segundo para captar las imágenes de los usuarios que serán identificados facialmente. La cámara utilizada en este proyecto esta interna en la PC y es de 1440p (alta definición).

El software del sistema es ejecutado en una PC, para este proyecto se utilizó una computadora con procesador Intel Core i7, 2.10 GHz de velocidad, 6 GB de RAM y sistema operativo Windows 7 Professional de 64 bits. Este software envía ordenes a un dispositivo Arduino UNO (Figura 2a) que se conecta a la PC vía USB para diferir las tareas de bloqueo y desbloqueo de la cerradura electrónica.

El Arduino UNO es una placa de desarrollo con terminales para entrada o salida de datos, que se utilizan para recibir información de sensores o bien, para enviar señales de encendido/apagado. La cerradura electrónica consiste en una contra-chapa (Figura 2b) que se desbloquea al recibir una señal de 12V, esta señal la recibe desde una fuente de alimentación de 12V, ya que la placa Arduino solo es capaz de enviar voltajes máximos de 5V.

Debido a lo explicado anteriormente, es necesaria la utilización de un relé eléctrico (Figura 2c), este dispositivo hace la función de ”switch”, es decir, cuando el relé recibe la señal (5V) del Arduino UNO, se activa y permite el flujo de voltaje (12V) desde la fuente de alimentación hasta la cerradura electrónica.

4.2. Sistema identificador facial

El sistema de identificación facial propuesto tiene como entrada las imágenes recibidas y como respuesta un resultado verdadero o falso, lo que significaría el

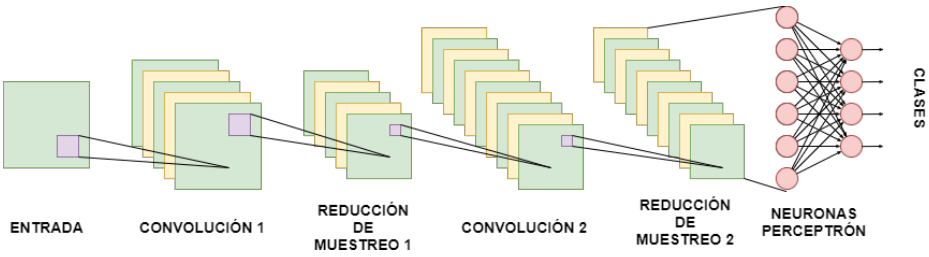


Fig. 6. Arquitectura clásica de una CNN.

desbloqueo o bloqueo de la cerradura biométrica respectivamente. En la Figura 2 se muestra el diagrama de flujo del tratamiento de las imágenes el cual lo podemos dividir en dos bloques.

El primer bloque corresponde a la creación de la base de datos de usuarios que el sistema puede reconocer facialmente. El segundo bloque realiza la tarea de Reconocimiento Facial en tiempo real, este boque se puede subdividir en tres etapas: (i) Detección Facial, (ii) Identificación Facial (usando una CNN) y (iii) Embedding.

4.3. Creación de la base de datos

Para la creación de la base de datos es necesario capturar fotografías frontales de los rostros de los usuarios que se desea tener disponibles en el sistema para su reconocimiento. Previo al cálculo de los Embeddings de estas imágenes, se realiza un aumento de datos (Figura 4).

El aumento de datos consiste en generar dos copias modificadas adicionales a partir de la original (Figura 4a), a la primera se le aplica un ligero cambio de ángulo (Figura 4b) y a la segunda un efecto de acercamiento (Figura 4c). Esto con el fin de fortalecer la base de datos y mejorar el cálculo de las distancias entre las personas.

Como se menciono anteriormente, estas imágenes son procesadas por el detector facial y la FaceNet. El resultado final de este tratamiento a las imágenes disponibles en la base de datos es un archivo llamado Embeddings, que son las distancias euclidianas de todas las identidades en un espacio de 128 dimensiones.

4.4. Reconocimiento facial

Como se menciono anteriormente, este bloque se subdivide en tres etapas. La etapa de detección facial se encarga de detectar y extraer los rostros de una imagen, esto se logra a través de modelos o moldes en formato XML llamados Haar Cascades, los cuales contienen características de un rostro frontal en lenguaje computacional, si una o varias regiones de la imagen corresponden con estos moldes, se extraen estas regiones en forma de coordenadas, si una imagen contiene rostros y son detectados se envían directamente a la siguiente etapa.

Las redes neuronales convolucionales (Figura 6) están compuestas por la capa de extracción de características y la capa de aprendizaje. La capa de extracción de

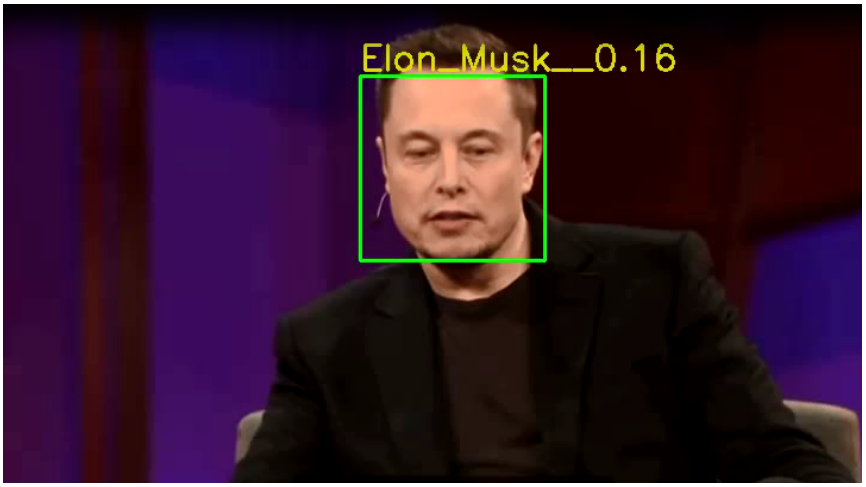


Fig. 7. Ejemplo de un reconocimiento facial exitoso.

características esta formada por capas de convolución y de pooling (reducción) y la capa de aprendizaje regularmente es una red fully conected.

Las CNN estan diseñadas para procesar datos que vienen en forma de múltiples matrices, por ejemplo, una imagen en color compuesta por tres matrices de dos dimensiones que contienen intensidades de píxeles en los tres canales de color [16].

Las etapas de Identificación Facial y Embedding se realizan con una herramienta desarrolla por Google llamada FaceNet [14]. FaceNet es una CNN pre-entrenada que extrae las características principales de los rostros y las transforma en un vector de 128 dimensiones conocido como ”embedding”.

Las CNNs convencionales no logran resolver un problema llamado ”One-Shot-Learning”, el cual se le atribuye a la incapacidad de hacer un entrenamiento óptimo con una sola imagen por usuario en la base de datos. En cambio, la FaceNet al ser una CNN pre-entrenada con más de 500 millones de imágenes (en su documentación menciona que) logra una efectividad del 99.63 %, siendo una herramienta muy útil y fácil de usar.

El entrenamiento de la FaceNet se realizó utilizando un proceso llamado ”Triplet Loss”(Figura 5). El proceso de Triplet Loss minimiza la distancia entre la base y la muestra para un rostro contenido en la base de datos. Y maximiza la distancia si la muestra no corresponde al rostro a identificar, lo que significa una identidad diferente [9].

Por lo que, una vez generados los embeddings de la base de datos, las imágenes recibidas en tiempo real son procesadas de la misma manera, pero para los fines de reconocimiento, el Embedding generado por la imagen no se almacena, sino que se compara la distancia con cada una de las identidades disponibles en los Embeddings y si la distancia está por debajo de un umbral, se le asigna esa identidad a la imagen (Figura 7).

Tabla 1. Matriz de confusión de los resultados obtenidos.

Real/Predicción	Positivo	Negativo
Positivo	160	0
Negativo	0	40

5. Análisis de resultados

El sistema de reconocimiento facial fue construido, compilado y ejecutado en lenguaje Python, únicamente una pequeña parte que se asigna a la programación del Arduino UNO fue desarrollada en lenguaje C++.

Las pruebas realizadas al sistema consistieron en la identificación de 50 personas, de las cuales 40 se encontraban registradas en el sistema para evaluar su correcta identificación y la correcta no identificación de las 10 personas restantes.

Para agilizar las pruebas, se realizó un vídeo con segmentos extraídos de otros vídeos de las 50 personas mencionadas anteriormente, para someterlo al identificador y obtener los resultados. La Tabla 1 se observa la matriz de confusión de los resultados obtenidos a las pruebas realizadas al sistema. En ella se observa 4 cuadrantes principales.

En el cuadrante superior izquierdo se determinan los resultados verdaderos-positivos (VP = 160) los cuales corresponden a la correcta identificación de una persona (asignar la identidad correcta al rostro de una persona). En el cuadrante superior derecho se determinan los resultados falsos-negativos (FN = 0) los cuales corresponden a la incorrecta no identificación de una persona (no asignar una identidad a un rostro disponible en la base de datos).

En el cuadrante inferior izquierdo se determinan los resultados falsos-positivos (FP = 0) los cuales corresponden la incorrecta identificación de una persona (asignar una identidad a un rostro incorrecto). Finalmente, en el cuadrante inferior derecho se determinan los resultados verdaderos-negativos (VN = 40) los cuales corresponden a la correcta no identificación de una persona (no asignar identidad a un rostro no disponible en la base de datos).

La sensibilidad y la especificidad son dos medidas diferentes de un modelo de clasificación binaria. La tasa de verdaderos positivos mide la frecuencia con la que clasificamos un registro de entrada como la clase positiva y su clasificación correcta [16]. La sensibilidad cuantifica qué tan bien el modelo evita los falsos negativos (Ecuación 1). La especificidad cuantifica qué tan bien el modelo evita los falsos positivos (Ecuación 2):

$$\text{Sensibilidad} = \frac{\text{VP}}{\text{VP} + \text{FN}}, \tag{1}$$

$$\text{Especificidad} = \frac{\text{VN}}{\text{VN} + \text{FP}}. \tag{2}$$

Tabla 2. Resultados obtenidos de las mediciones realizadas.

Medida	**Resultado**
Sensibilidad	1.0
Especificidad	0.0
Exactitud	1.0
Precisión	1.0
F1	1.0

La exactitud es el grado de cercanía de las mediciones de una cantidad al valor real de esa cantidad (Ecuación 3) [16]:

$$\text{Exactitud} = \frac{\text{VP} + \text{VN}}{\text{VP} + \text{FP} + \text{FN} + \text{VN}}. \tag{3}$$

El grado en que las mediciones repetidas en las mismas condiciones nos dan los mismos resultados se llama precisión en el contexto de la ciencia y la estadística. La precisión también se conoce como valor de predicción positivo (Ecuación 4) [16]:

$$\text{Precision} = \frac{\text{VP}}{\text{VP} + \text{FP}}. \tag{4}$$

En la clasificación binaria, consideramos que la puntuación F1 (o puntuación F, medida F) es una medida de la precisión de un modelo. La puntuación F1 es la media armónica de las medidas de precisión y sensibilidad (descritas anteriormente) en una única puntuación [16], como se define aquí:

$$\text{F1} = \frac{2\,\text{VP}}{2\,\text{VP} + \text{FP} + \text{FN}}. \tag{5}$$

Vemos puntajes para F1 entre 0.0 y 1.0, donde 0.0 es el peor puntaje y 1.0 es el mejor puntaje que nos gustaría ver. La puntuación F1 se utiliza normalmente en la recuperación de información para ver qué tan bien un modelo recupera resultados relevantes. En el aprendizaje profundo, vemos que la puntuación F1 se utiliza como una puntuación general sobre el rendimiento de nuestro modelo [16].

En la Tabla 2 se encuentran los resultados alcanzados de las diferentes medidas descritas anteriormente con base en la matriz de confusión obtenida al principio de este capítulo. Si bien los resultados son ideales, esto hace referencia a los resultados obtenidos por los autores que realizaron el entrenamiento de la Facenet.

Como se menciono anteriormente la red neuronal utilizada para este proyecto alcanza un 99.63 % de rostros reconocidos exitosamente, es decir, de cada 10,000 rostros, 37 serán reconocidos de forma incorrecta, por lo que su uso para los fines establecidos en este trabajo es de extrema seguridad, ya que, por lo general la cantidad de personas que son admitidas en ciertos lugares es relativamente pequeña.

Además, es necesario mencionar que las pruebas realizadas al sistema, constaron de fragmentos de vídeos que favorecieran al reconocimiento, es decir, aseguramos que en algún instante del vídeo el rostro de la persona se presentara de forma frontal, clara y libre de obstáculos que pudieran perjudicar el reconocimiento, esto debido a que el sistema supone de un consentimiento por parte del usuario a mostrar su rostro ante la cámara para su identificación.

6. Conclusiones

El diseño de la cerradura biométrica es sencilla y funcional para los fines propuestos en el trabajo. El desempeño del hardware utilizado fue el esperado y se lograron realizar las pruebas correspondientes al sistema de software de una manera excelente, obteniendo resultados extraordinarios a razón de costo/beneficio. Nuestro sistema de reconocimiento facial posee características ideales para su implementación en sistemas de acceso biométricos, ya que brinda altos estándares de fiabilidad, incluso si se utiliza un alto numero de usuarios registrados.

En su conjunto (hardware y software) el sistema propuesto, ofrece una notable opción para quienes requieren de un sistema de cerradura. El interés en sistemas automáticos ha crecido a lo largo de la historia y actualmente ese crecimiento sigue en auge, el ser humano ha desarrollado tecnologías que disminuyen el esfuerzo humano o bien, que provean mayor seguridad en tareas difíciles, peligrosas o que requieran de altos niveles de precisión.

Es por eso que el desarrollo de técnicas diferentes promete el avance tecnológico y por ende, mejores productos y a menor precio de mercado. La instalación de un sistema como el propuesto en el trabajo presente, podría reducir en más de un 50 % del costo respecto a los sistemas presentes en el mercado actual. Este beneficio de precio se puede ver aumentado significativamente si toman en cuenta que es posible administrar múltiples puertas con un solo dispositivo y solamente colocando una cámara por puerta.

6.1. Trabajo futuro

Existen diferentes áreas de oportunidad derivadas de nuestro sistema que seguramente aumentaran las capacidades de nuestra propuesta.

La primera es migrar a un dispositivo de IoT capaz de soportar sistemas basados en redes neuronales, p. ej., una Jetson Nano de NVIDIA. Esta extensión del proyecto proveería de un aumento de algunas capacidades, p. ej.: implementaciones de IoT, portabilidad, velocidad de respuesta, menor tamaño de prototipo, entre otras.

La segunda es incrustar una etapa de detección de realidad previa a la etapa de reconocimiento facial, esto es debido a que el sistema actual reconoce de igual forma a personas reales y fotografías. Una etapa de detección de realidad, filtraría únicamente rostros de personas reales y desecharía rostros de fotografías impresas o digitales.

Referencias

1. Baidya, J., Saha, T., Moyashir, R., Palit, R.: Design and implementation of a fingerprint based lock system for shared access. In: Proceedings of the IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC). IEEE (2017) doi: 10.1109/ccwc.2017.7868448
2. Balla, P. B., Jadhao, K. T.: IoT based facial recognition security system. In: Proceedings of the International Conference on Smart City and Emerging Technology (2018) doi: 10.1109/icscet.2018.8537344

3. Cao, Q., Shen, L., Xie, W., Parkhi, O.M., Zisserman, A.: VGGFace2: A dataset for recognising faces across pose and age. In: Proceedings of the 13th IEEE international conference on automatic face and gesture recognition (2018) doi: 10.48550/arXiv.1710.08092
4. Coskun, M., Ucar, A., Yildirim, O., Demir, Y.: Face recognition based on convolutional neural network. In: Proceedings of the International Conference on Modern Electrical and Energy Systems (2017) doi: 10.1109/mees.2017.8248937
5. Divya, R. S., Mathew, M.: Survey on various door lock access control mechanisms. In: Proceedings of the International Conference on Circuit, Power and Computing Technologies (2017) doi: 10.1109/iccpct.2017.8074187
6. INEGI. Incidencia delictiva (2021) www.inegi.org.mx/temas/incidencia/
7. Jafri, R., Arabnia, H. R.: A survey of face recognition techniques. Journal of Information Processing Systems, vol. 5, no. 2, pp. 41–68 (2009) doi: 10.3745/JIPS.2009.5.2.041
8. LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature, vol. 521, no. 7553, pp. 436–444 (2015) doi: 10.1038/nature14539
9. Ming, Z., Chazalon, J., Luqman, M. M., Visani, M., Burie, J. C.: Simple triplet loss based on intra/inter-class metric learning for face verification. In: IEEE International Conference on Computer Vision Workshops (2017) doi: 10.1109/iccvw.2017.194
10. Nehete, P. R., Chaudhari, J., Pachpande, S. R., Rane, K. P.: Literature survey on door lock security systems. International Journal of Computer Applications, vol. 153, no. 2, pp. 13–18 (2016) doi: 10.5120/ijca2016911971
11. Parkhi, O. M., Vedaldi, A., Zisserman, A.: Deep face recognition. In: Procedings of the British Machine Vision Conference, British Machine Vision Association (2015) doi: 10.5244/c.29.41
12. Parmar, D. N., Mehta, B .B.: Face recognition methods and applications. Journal of Information Processing Systems, vol. 5, no. 2, pp. 41–68 (2009) doi: 10.3745/JIPS.2009.5.2.041
13. Radzi, S. A., Alif, M. K., Athirah, Y. N., Jaafar, A. S., Norihan, A. H., Saleha, M. S.: IoT based facial recognition door access control home security system using raspberry pi. International Journal of Power Electronics and Drive Systems, vol. 11, no. 1, pp. 417 (2020) doi: 10.11591/ijpeds.v11.i1.pp417-424
14. Schroff, F., Kalenichenko, D., Philbin, J.: FaceNet: A unified embedding for face recognition and clustering. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (2015) doi: 10.1109/cvpr.2015.7298682
15. Taigman, Y., Yang, M., Ranzato, M., Wolf, L.: DeepFace: Closing the gap to human-level performance in face verification. IEEE Conference on Computer Vision and Pattern Recognition (2014) doi: 10.1109/cvpr.2014.220
16. Wolf, L., Hassner, T., Maoz, I.: Face recognition in unconstrained videos with matched background similarity. In: Proceedings of the Computer Vision and Pattern Recognition Conference (2011) doi: 10.1109/CVPR.2011.5995566
17. Yedulapuram, S., Arabelli, R., Mahender, K., Sidhardha, C.: Automatic door lock system by face recognition. IOP Conference Series: Materials Science and Engineering, vol. 981, no. 3 (2020) doi: 10.1088/1757-899x/981/3/032036

ISSN 1870-4069

Sistema de visión inteligente para monitorizar polinizadores (MonPo) usando aprendizaje profundo

Daniela Bolaños-Flores[1], Tania A. Ramírez-del Real[2], Magali Arellano-Vázquez[3], Dagoberto Armenta-Medina[3], Guadalupe O. Gutierrez-Esparza[4], Hamurabi Gamboa-Rosales[3]

[1] Universidad Tecnológica del Centro de Veracruz, México

[2] Consejo Nacional de Ciencia y Tecnología, Centro de Investigación en Ciencias de Información Geoespacial, México

[3] Consejo Nacional de Ciencia y Tecnología, Centro de Investigación e Innovación en Tecnologías de la Información, México

[4] Consejo Nacional de Ciencia y Tecnología, Instituto Nacional de Cardiología Ignacio Chávez, México

14484@utcv.edu.mx, tramirez@centrogeo.edu.mx, {magali.arellano, dagoberto.armenta, hamurabi.gamboa}@infotec.mx, ggutierreze@conacyt.mx

Resumen. Los ecosistemas están en constante amenaza, uno de los factores relevantes es, la crisis mundial de polinización. Adquirir el entendimiento sobre polinizadores y su comportamiento se ha convertido en un tema emergente para la sociedad. Actualmente, algunos de los métodos para registrar la actividad de los polinizadores en el campo son de forma manual, lo anterior implica tiempo y alto costo de mano de obra. El avance tecnológico permite la utilización de sistemas de videovigilancia en el sector agrícola, posibilitando la captura de datos sobre la interacción entre insectos y plantas. En el presente trabajo es propuesto un sistema de videovigilancia para monitorizar polinizadores (abejas y mariposas), denominado MonPo. El sistema consiste en una cámara Web, un sensor para detectar el movimiento y una raspberry Pi 3 como procesador digital. El cual logra identificar abejas y mariposas con una exactitud mayor al 90 %, empleando una red de aprendizaje profundo (VGG-16). El sistema MonPo almacena los datos cuando se detecta el movimiento, específicamente la fecha, hora y la clase del polinizador identificado (abeja o mariposa). Lo anterior permitirá realizar un análisis referente a la presencia de polinizadores en las diferentes estaciones del año, así como las plantas que favorecen la polinización.

Palabras clave: Polinizadores, procesamiento de imágenes, videovigilancia, aprendizaje profundo.

Daniela Bolaños-Flores, Tania A. Ramírez-del Real, Magali Arellano-Vázquez, et al.

Intelligent Vision System to Monitor Pollinators (MonPo) Using Deep Learning

Abstract. Ecosystems are in constant threat; one of the relevant factors is the global pollination crisis. Gaining an understanding of pollinators and their behavior has become an emerging issue for society. Some of the methods to record pollinator activity in the field are manual, which implies time and high labor costs. Technological advance allows the use of video surveillance systems in the agricultural sector, making it possible to capture data on the interaction between insects and plants. In the present work, a video surveillance system is proposed to monitor pollinators (bees and butterflies), called MonPo. The system consists of a webcam, a sensor to detect motion, and a Raspberry Pi 3 as a digital processor. Which manages to identify bees and butterflies with an accuracy greater than 90 %, using a deep learning network (VGG-16). The MonPo system stores data when movement is detected, precisely the date, time, and class of the identified pollinator (bee or butterfly); the above will allow an analysis regarding the presence of pollinators in the different seasons of the year, as well as the plants that favor pollination.

Keywords: Pollinators, image processing, video surveillance, deep learning.

1. Introducción

Actualmente, el uso de la tecnología se ha expandido en diversas áreas, una de éstas, es el sector agrícola, particularmente en la polinización, donde se requiere la acción de insectos con la capacidad de transportar el polen entre las flores para así lograr la producción de frutos y semillas. La polinización es esencial para la producción de alimentos, la viabilidad y diversidad genética de las especies vegetales con flor.

Los polinizadores son un grupo diverso de animales que incluyen algunas variedades de insectos, como lo son los coleópteros, lepidópteros, dípteros, además de muchas aves, murciélagos, roedores e incluso lagartijas. Los polinizadores son responsables del 80 % de reproducción de las especies vegetales, la variedad de éstos dificulta el estudio en extenso del declive de la población; la principal causa identificada es la pérdida del hábitat [1].

La modificación del entorno natural repercute en que los polinizadores no logran adaptarse, al no encontrar suficiente comida ni lugares de anidamiento. En el caso de las superficies de cultivo, los polinizadores aunque pueden encontrar los recursos suficientes, se enfrentan a agroquímicos y no solo los insecticidas, sino fungicidas que afectan la microbiota de la flora y a herbicidas que reducen los recursos florales disponibles [1].

Últimamente el uso excesivo de plaguicidas, herbicidas e insecticidas en el aire libre, así como también el cambio en el uso del suelo, la invasión de plantas y animales no nativos en los ecosistemas naturales, han sido un factor negativo para los principales polinizadores [5].

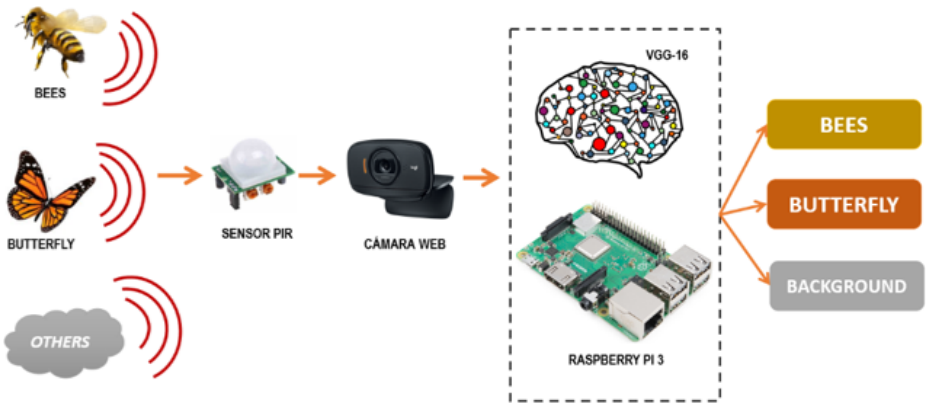


Fig. 1. Propuesta del sistema de visión inteligente para la detección de polinizadores.

Los plaguicidas se consideran una de las mayores amenazas para la conservación de la biodiversidad, reduciendo las fuentes de abastecimiento alimenticio de dichos insectos y disminuyendo su ciclo de vida, lo anterior, puede generar una perturbación en la humanidad debido a una disminución del 35 % [7] de la producción mundial de cultivos, además de afectar la salud, cambiando así la dieta de la humanidad mientras se incrementa la tasa bruta de mortalidad, a consecuencia de la falta de nutrientes que se obtienen de las frutas y verduras.

El cambio climático afecta a los polinizadores, como en el caso de la llegada de especies exóticas que compiten con las especies nativas, llevando a estas poblaciones a su extinción [1]. La monitorización de los polinizadores es una herramienta básica para la conservación, ya que pueden detectar de manera temprana señales de alerta en su comportamiento, además provee información para su estudio.

La disponibilidad de datos ecológicos abre la puerta al uso de minería de datos basados en la predicción, como el aprendizaje automático para la identificación de patrones. La proliferación de dispositivos para monitorización de polinizadores y sensado de variables climáticas aunado al apoyo de la ciencia ciudadana es una oportunidad para la creación conjunta de fuentes de datos [1].

Hoy en día, existe una gran cantidad de sistemas para detección de animales, pero suelen estar limitados en su funcionamiento, ya que algunos requieren una caja trampa, o cuentan con una limitada precisión, es por ello que en este trabajo se propone realizar un sistema encargado de detectar y clasificar a distintos polinizadores para la monitorización del comportamiento, mediante la aplicación de herramientas de aprendizaje automático.

El sistema hace uso de un algoritmo basado en Inteligencia Artificial para la clasificación de las distintas especies de insectos voladores, y en un futuro extraer la información de los patrones de comportamiento, así como tener propuestas para su conservación y cuidado. Este trabajo presenta la implementación de una arquitectura de una red neuronal convolucional para la detección y clasificación de algunos polinizadores, en particular abejas y mariposas, lo anterior permite estudiar su comportamiento, en diferentes estaciones del año.

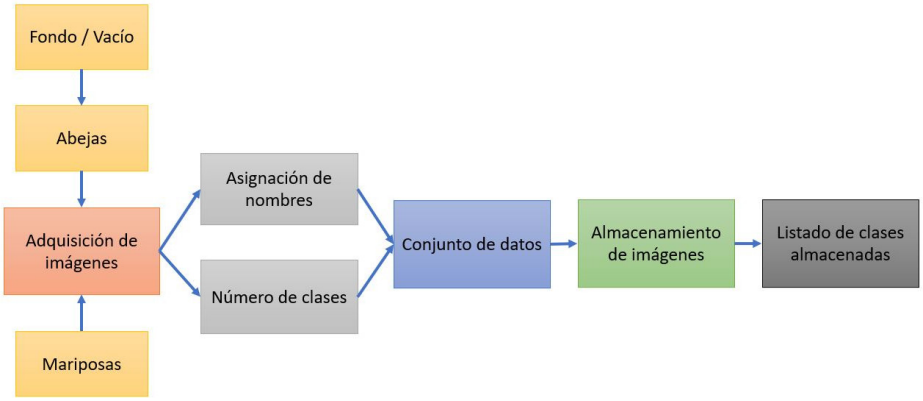


Fig. 2. Diagrama de adquisición de imágenes.

El sistema está basado en un procesador digital de arquitectura abierta (Raspberry Pi 3), que incluye una cámara para la toma de fotografía, además se incluye un sensor de movimiento para precisar el momento de captura de imagen.

El artículo está organizado de la siguiente manera, en la sección 2 se presentan los trabajos relacionados, en particular los sistemas similares para la detección automática de polinizadores, en la sección 3 se detalla el desarrollo del sistema propuesto, en la sección 4 se muestran los resultados obtenidos en la identificación de polinizadores, finalmente en la sección 5 se establecen las conclusiones, así como el trabajo futuro.

2. Trabajos relacionados

Existen diversos trabajos que clasifican polinizadores, algunos de ellos utilizan técnicas de inteligencia artificial, sin embargo, muchos de ellos realizan la detección fuera de línea. Entre los estudios enfocados en la monitorización de polinizadores, en general los apicultores revisan la colmena para inspeccionar la cantidad de polen y néctar, lo que permite la detección temprana de enfermedades o plagas, las revisiones regulares permiten prevenir plagas, virus o infecciones bacterianas y este proceso se hace de manera manual.

Sin embargo, regularmente las colmenas se encuentran en lugares aislados y las revisiones manuales interfieren en el ciclo de vida de las colonias de abejas. La monitorización automática hace posible capturar grandes cantidades de datos acerca del comportamiento de abejas, sin interferir en su cotidianidad. Las variables como la temperatura interna de la colmena, el sonido de los zumbidos, la variación de peso de la colmena, fotografías y vídeos del flujo de abejas aportan importante información sobre la salud de la colmena.

El análisis de estos datos de manera continua apoya a la identificación de patrones que alerta cambio en la vida normal de la colmena. En [6] se realiza un amplio estudio de las grabaciones de los zumbidos, se ha determinado que se pueden generar dos tipos zumbidos; el primero se produce por el rápido batido de de las alas que genera un zumbido audible y perceptible por los dispositivos de grabación; el segundo se

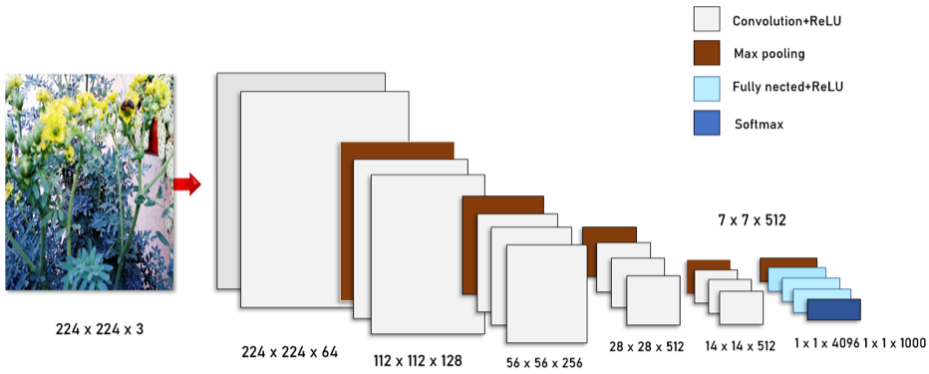


Fig. 3. Arquitectura red neuronal convolucional VGG-16.

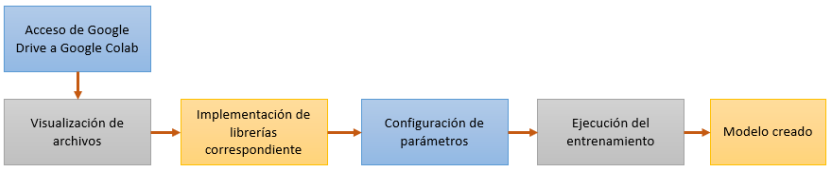


Fig. 4. Diagrama de creación del modelo.

produce cuando las abejas visitan una flor, éstas generan una vibración usando los músculos de sus alas y tórax lo que causa que el polen se transfiera a otras flores, produciéndose así la polinización. Utilizando un sistema de adquisición de datos automático que usa muestras de audio que a su vez alimentan modelos de aprendizaje automático para asistir a los cuidadores de las colmenas, reduciendo o evitando así las revisiones manuales.

Existen otros sistemas de monitorización miniatura, sin embargo, son invasivos, como el mostrado en [4], que incluye un sistema de ubicación geográfica, sensores de humedad, temperatura y radiación solar, este dispositivo se instala pegándolo en la espalda de un abejorro. El trabajo presentado en [10] desarrolla un sistema de visión para el conteo y reconocimiento de insectos voladores en la agricultura inteligente.

La imagen digital de los insectos voladores se obtiene cuando llegan a los sembradíos y los atrapan, para después detectar las diferentes especies de vuelo, como también el ambiente en el que se encuentre. La técnica utilizada es una red YOLO y una máquina de soporte vectorial (SVM, por sus siglas en inglés), en conjunto logran una exactitud aproximada al 90 %.

3. Desarrollo

La Figura 1 muestra la propuesta del sistema de visión artificial inteligente para la detección y monitorización de polinizadores: MonPo. El sistema está colocado en el área de monitorización (planta), donde se detecta la presencia de los diferentes insectos voladores que circulan, por medio de un sensor modelo HC-SR501, conocido como

Tabla 1. Conjunto de datos de entrenamiento y prueba.

Clase	Imágenes de entrenamiento	Imágenes de prueba	Total
Abeja	1154	385	1539
Mariposa	116	372	1488
Fondo	1071	357	1428

sensor infrarrojo piroeléctrico (PIR, por sus siglas en inglés), este sensor detecta el movimiento de cuerpos que emiten radiación electromagnética infrarroja dependiendo de su temperatura. Una vez que el insecto haya sido detectado, el sensor manda una señal digital en alto para proceder a la activación y toma de fotos mediante la cámara, se toma una ráfaga de imágenes en un lapso de tiempo determinado, mientras que el insecto está realizando la polinización en las flores.

Posterior a la captura de fotografías, las imágenes se procesan con ayuda de la Raspberry Pi, mediante un modelo elaborado a través de una red neuronal convolucional (VGG-16 [9]), la cual está entrenada con un conjunto de datos.

Una vez entrenada, se realiza el análisis de la clasificación y predicción en tiempo real acerca de los insectos mencionados anteriormente. Por último, una vez terminado la experimentación de la predicción, se envía un bit que contenga la etiqueta que pertenece a cada clase del insecto detectado.

3.1. Adquisición de imágenes

En la Figura 2 se muestra el proceso para realizar la adquisición de imágenes y la preparación de éstas para la configuración del entrenamiento.

Antes de adquirir las imágenes, es importante dividir en clases los diferentes tipos de polinizadores utilizados, los cuales son: “abejas” y “mariposas”, además se considera una clase extra, la cual pertenece a la planta y se denomina “fondo”, las imágenes se guardan en formato “jpg” o “png”.

– Asignación de nombres.

Se realiza el cambio de nombre de las fotografías, se modifica el nombre por defecto generado por la cámara a una cadena que compuesta por una numeración consecutiva para cada imagen.

– Número de clases.

Posteriormente, se etiquetan las clases con números, es decir, para la clase abejas se le proporciona el número “1”, para “mariposa” el número “2” y por último para “fondo/vacío” el número “3”.

– Conjunto de datos.

Para que el sistema realice la identificación y detección de insectos polinizadores, es necesario entrenar la red neuronal convolucional (VGG16, ver Figura 3), por medio de imágenes referentes a abejas, mariposas y fondo, es importante mencionar que algunas imágenes fueron obtenidas de la base de datos de la plataforma

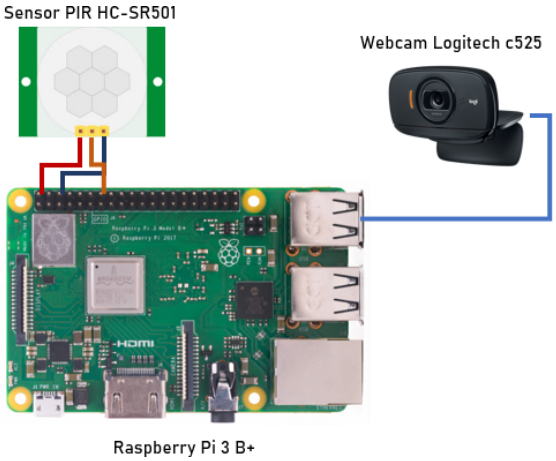


Fig. 5. Diagrama electrónico del sistema.

"Kaggle", además se complementó con fotografías propias adquiridas en pruebas en el área de experimentación. El conjunto de datos se divide con el número total de imágenes por cada clase, teniendo un 75 % para el apartado de entrenamiento y un 25 % para validación.

– Almacenamiento de imágenes.

El modelo de la red neuronal convolucional VGG16 se genera por medio del notebook "Google Colab", debido a esto, es necesario subir las carpetas de las clases mencionadas anteriormente a la plataforma de "Google Drive".

– Listado de clases almacenadas.

Por otra parte, antes de proceder a la configuración de entrenamiento, se genera un archivo de texto con extensión ".txt", con el propósito de listar las imágenes que pertenecen al conjunto de datos de entrenamiento para cada clase.

3.2. Configuración de la red y generación del modelo

Con base en el método de aprendizaje profundo se hace el proceso de entrenamiento, con el propósito de realizar predicciones, es decir, crear un modelo que se adapte a la necesidad del proyecto, en este caso, enfocado para el análisis de los diferentes insectos polinizadores, de manera que realice la detección y clasificación en tiempo real en un entorno adaptado y monitorizado.

La Figura 4 muestra el proceso para la configuración y generación del modelo, debido a que se trabaja en Google Colab [2], primero se da acceso a las carpetas de Google Drive. Teniendo los permisos solicitados, para comenzar con el proceso de entrenamiento se requiere hacer la importación de las librerías necesarias tales como, el módulo de keras [3] para el procesamiento de las imágenes, creación del modelo, el módulo numpy para la creación de vectores y módulos como sklearn [8] que contiene algoritmos de clasificación.

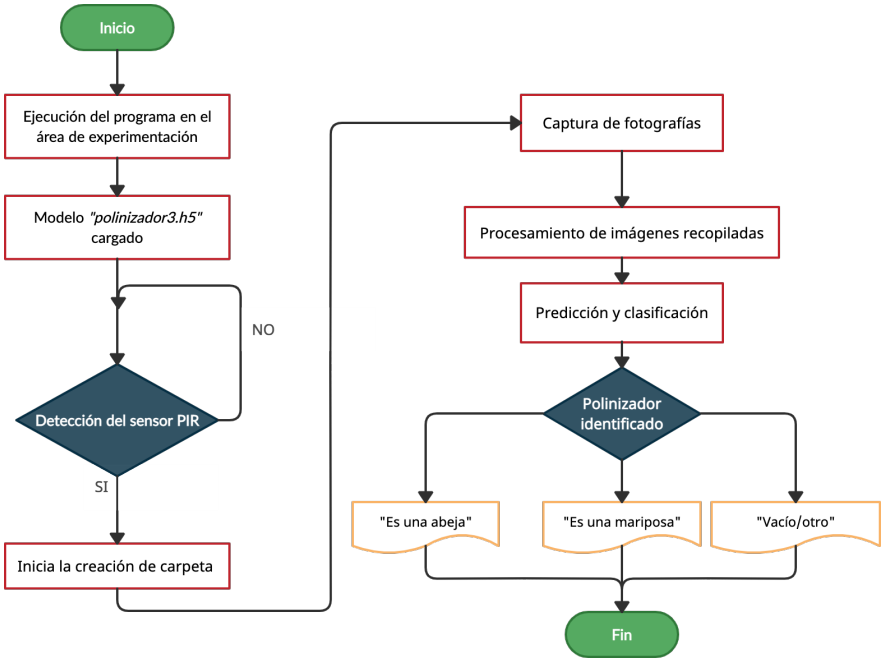


Fig. 6. Diagrama de flujo del funcionamiento del sistema.

Una vez establecido el modelo de la red neuronal convolucional, se establecen los parámetros para el entrenamiento. Finalmente, con el conjunto de datos distribuido como se muestra en la Tabla 1 se procede a la creación del modelo denominado "polinizadores3.h5.", el cual se exporta a la Raspberry Pi 3.

3.3. Diagrama electrónico

Para el sistema desarrollado, el sensor PIR utilizado es el modelo hc-sr501, el cual tiene la función de detectar el movimiento. Además, cuenta con tres terminales de conexión, una para voltaje de alimentación (5V), la segunda para la señal de salida, la cual está conectada al GPIO 23 (pin de propósito general de entrada/salida, General Purpose Input/Output) de la tarjeta, finalmente, la tercera se conecta a tierra.

Por medio de este sensor se detecta el movimiento del insecto cuando está realizando la polinización, mandando una señal digital en alto cada vez que manifieste algún movimiento. De igual manera, se emplea una cámara web, modelo Logitech C525, conectada por medio del puerto USB 2.0 a la tarjeta Raspberry Pi 3. La Figura 5 muestra el diagrama electrónico del sistema.

3.4. Proceso del sistema MonPo

La Figura 6 muestra el diagrama de flujo del proceso que realiza el sistema MonPo para la detección y clasificación de polinizadores. Primero, los dispositivos se colocan en el área de experimentación, después se carga el modelo "polinizadores3.h5".

Fig. 7. Posicionamiento de la cámara en el jardín.

Una vez cargado el modelo y ejecutado el programa, el sensor PIR se queda a la espera de la presencia de algún polinizador en el área específica, la señal de movimiento enviada por el sensor activa el funcionamiento del sistema, después se genera una carpeta para el almacenamiento de las fotografías capturadas por la cámara web de los insectos en movimiento. Posteriormente, se procesan las imágenes para la clasificación, en caso que, el modelo haya detectado algún polinizador, envía un mensaje en el puerto serial con el nombre del insecto.

Es importante destacar que el viento puede provocar la activación del sensor, por esta razón existe la clase "fondo", además el sensor es capaz de accionarse con el desplazamiento de algún objeto, incluso de la misma planta, lo anterior implica que el polinizador debe estar en el ángulo de alcance, el cual es de 120^o.

4. Resultados

Para las pruebas de experimentación se realizan dos procesos diferentes, uno cuenta con un sensor para la detección de movimiento de los polinizadores, el segundo se basa en la ejecución del programa con las imágenes almacenadas en la Raspberry Pi 3 para la validación.

El proceso de experimentación es por medio de ejecución remota, es decir, justo después de colocar el sistema completo en el área de prueba, se procede a ejecutar el programa en el momento que se encuentre un polinizador en la zona, para que a realice la captura de fotografías mientras que los insectos llevan a cabo la polinización, posterior a eso, se realiza el análisis para la clasificación e identificación.

Para la primera prueba del modelo denominado se obtienen resultados satisfactorios debido a que la estación se colocó en una posición en donde la cámara hace el encuadre al jardín donde se encuentran las flores para tomar fotografías de "vacío/fondo", como se muestra en la figura 7.

Fig. 8. Fotografías capturadas por la Raspberry Pi y Webcam.

La figura 8 muestra imágenes capturadas por la Raspberry Pi y la Webcam, obteniendo fotografías nítidas de algunas flores del jardín para su posterior análisis y clasificación. Considerando las muestras obtenidas de la experimentación pasada, se opta por entrenar el modelo para aumentar su exactitud de predicción, con el fin de ponerlo en práctica en la segunda fase de experimentación, por lo que se encuentra automatizado gracias a la implementación del sensor de movimiento mencionado anteriormente.

Como se muestra en la Figura 7 para realizar las pruebas en un entorno real de experimentación, se coloca en el jardín de experimentación del Centro de Investigación e Innovación en Tecnologías de la Información y Comunicación (INFOTEC), donde el funcionamiento del sistema se puede efectuar al momento de que el insecto volador se presente frente al sensor, al mismo tiempo que éste realiza la actividad de polinización.

Al momento de cumplir alguno de los dos casos anteriores, es decir, si el sensor es activado, se procede a crear una carpeta con el nombre de cada prueba capturada y el número de detección al que pertenece, una vez teniendo este espacio, se procede a capturar por medio de la cámara web, una ráfaga de 8 imágenes.

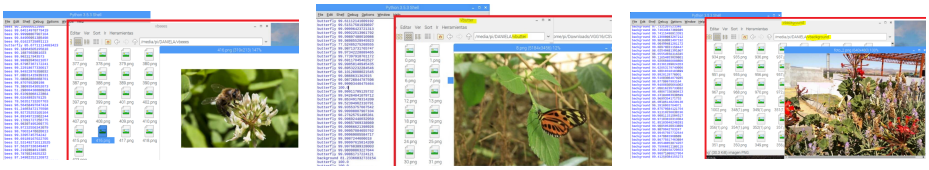
Posteriormente, se lleva a cabo el análisis de clasificación e identificación de la información obtenida por esta misma, el resultado se muestra en un archivo CSV con la predicción realizada por el modelo mediante una etiqueta.

El proceso para evaluar el modelo entrenado se realiza con la validación del conjunto de datos reservado para ello (25 %). Los resultados obtenidos se muestran en la Tabla 2. Las métricas de precisión, recall y F1-score muestran que el desempeño de la red VGG-16 supera en la mayoría de las clases el 90 %.

Finalmente, en la figura 9 se muestra un ejemplo de la validación del experimento realizado en el sistema inteligente, en las figuras 9a, 9b y 9c se encuentran cada una las clases; abeja, mariposa y fondo respectivamente.

Tabla 2. Resultados de las métricas de validación para cada clase.

	Precision	**Recall**	**F1-score**	**Support**
Abeja	0.9828	0.8909	0.9346	385
Mariposa	0.9046	0.9946	0.9475	372
Fondo	0.9607	0.9580	0.9593	357
Accuracy			0.9470	1114
Macro avg	0.9494	0.9478	0.9471	1114
Weighted avg	0.9496	0.9470	0.9468	1114



(a) Abeja (b) Mariposa (c) Fondo

Fig. 9. Validación de clases.

5. Conclusiones

En el presente trabajo se hizo uso de una red convolucional VGG-16, además se incluye un escenario de internet de las cosas (IoT, por sus siglas en inglés), con el objetivo de clasificar y detectar polinizadores, específicamente abejas y mariposas. Asimismo, se muestra el uso de procesadores digitales, como lo es una tarjeta Raspberry Pi 3, para la implementación de un modelo de aprendizaje profundo, el cual se aplica en la clasificación de imágenes.

Los resultados presentados en la Tabla 2 muestran un buen desempeño en las métricas establecidas, en cuanto a la prueba sin la implementación del sensor PIR, se obtuvo porcentajes mayores al 95 % de exactitud en 199 imágenes capturadas. Para la última sección de clasificación con los datos de validación pertenecientes al 25 % del total de imágenes de cada una de las clases, en la mayoría de los casos tuvo como resultante porcentajes mayores al 90 %.

El sistema MonPo permite almacenar los datos, particularmente la identificación de polinizadores, así como su comportamiento con las plantas, debido a que se captura la fecha y hora en la cual estuvo presente el insecto. Lo anterior, permitirá realizar un análisis referente a la presencia de polinizadores en las diferentes estaciones del año, así como las plantas que favorecen la polinización. Además, el sistema puede tener más datos para identificación de más factores agroclimáticos, mediante el uso y lectura de más sensores, como lo pueden ser de temperatura, humedad, velocidad del viento, entre otros.

Agradecimientos. Los autores agradecen al Consejo Nacional de Ciencia y Tecnología. En particular, a los proyectos números 576, 735 y 737 del Programa Cátedras CONACyT.

Referencias

1. Bartomeus, I., Dicks, L. V.: The need for coordinated transdisciplinary research infrastructures for pollinator conservation and crop pollination resilience. Environmental Research Letters, vol. 14, no. 4 (2019) doi: 10.1088/1748-9326/ab0cb5
2. Carneiro, T., Da Nóbrega, R. V. M., Nepomuceno, T., Bian, G. B., De Albuquerque, V. H. C., Reboucas Filho, P. P.: Performance analysis of google colaboratory as a tool for accelerating deep learning applications. IEEE Access, vol. 6, pp. 61677–61685 (2018) doi: 10.1109/access.2018.2874767
3. Chollet, F., et al.: Keras. https://keras.io (2015)
4. Iyer, V., Nandakumar, R., Wang, A., Fuller, S. B., Gollakota, S.: Living IoT: A flying wireless platform on live insects. In: Proceedings of the 25th Annual International Conference on Mobile Computing and Networking, pp. 1–15 (2019) doi: 10.1145/3300061.3300136
5. Kearns, C. A., Inouye, D. W., Waser, N. M.: Endangered mutualisms: The conservation of plant-pollinator interactions. Annual Review of Ecology and Systematics, vol. 29, no. 1, pp. 83–112 (1998) doi: 10.1146/annurev.ecolsys.29.1.83
6. Kulyukin, V., Mukherjee, S., Amlathe, P.: Toward audio beehive monitoring: Deep learning vs. standard machine learning in classifying beehive audio samples. Applied Sciences, vol. 8, no. 9, pp. 1573 (2018) doi: 10.3390/app8091573
7. Organización de las Naciones Unidas para la Agricultura y la Alimentación: La reducción de la población de abejas es una amenaza para la seguridad alimentaria y la nutrición. URL http://www.fao.org/news/story/es/item/1194963/icode/
8. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, vol. 12, pp. 2825–2830 (2011)
9. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014) doi: 10.48550/ARXIV.1409.1556
10. Zhong, Y., Gao, J., Lei, Q., Zhou, Y.: A vision-based counting and recognition system for flying insects in intelligent agriculture. Sensors, vol. 18, no. 5, pp. 1489 (2018) doi: 10.3390/s18051489

ISSN 1870-4069

Metaheurísticas y CNN: Comparación de modelos híbridos para mejorar la clasificación de imágenes

Gerardo Treviño-Valdés[1], Jesús Alejandro Navarro-Acosta[2], Marco Antonio Aceves-Fernández[1], Jesús Carlos Pedraza-Ortega[1], Saúl Tovar-Arriaga[1]

[1] Universidad Autónoma de Querétaro,
Facultad de Ingeniería,
México

[2] Universidad Autónoma de Coahuila,
Centro de Investigación en Matemáticas Aplicadas,
México

{geratrevino115, alexnav24, marco.aceves}@gmail.com
caryoko@yahoo.com, saul.tovar@uaq.mx

Resumen. En este artículo presenta la comparación entre una red neuronal convolucional estándar y una variante híbrida de la misma incluyendo algoritmos metaheurísticos para la clasificación de imágenes. Se pretende mejorar la clasificación del algoritmo implementando un ajuste en los hiperparámetros de la red neuronal convolucional con el algoritmo híbrido propuesto. Los algoritmos metaheurísticos aquí presentados son: Algoritmos Genéticos, Optimización por enjambre de partículas, Optimización de Lobo Gris y Optimización de Ballena Jorobada. Los resultados muestran que la metodología implementada es capaz de aumentar la exactitud en la clasificación, como de algunas otras de las métricas para evaluar la clasificación.

Palabras clave: Red neuronal convolucional, hibridismo, metaheurísticas, clasificación de imágenes.

Metaheuristics and CNN: Hybrid Model Comparison to Improve Image Classification

Abstract. In this paper presents the comparison between a standard convolutional neural network and a hybrid variant of it, including metaheuristic algorithms for image classification. The aim is to improve the classification of the algorithm by implementing an adjustment in the hyperparameters of the convolutional neural network with the proposed hybrid algorithm. The metaheuristic algorithms presented here are: Genetic Algorithms, Particle Swarm Optimization, Gray Wolf Optimization, and Humpback Whale Optimization. The results show that the methodology implemented is able for increasing the accuracy in the classification, as well as some other of the metrics to evaluate the classification.

Keywords: Convolutional neural network, hybridism, metaheuristics, image clasification.

1. Introducción

Las redes neuronales convolucionales (CNN) han demostrado un rendimiento en el procesamiento de imágenes al aumentar su desempeño en tareas como clasificación, detección de objetos, entre otras. Así como la capacidad de adaptación de sus modelos [5]. Los enfoques de detección de objetos basados en características y clasificadores de aprendizaje automático han sido muy fructíferos hasta tiempos recientes.

Cuando se aplican a diferentes tareas o se adaptan para desafíos adicionales, estos requieren de un ajuste de parámetros y una reducción dimensional para lograr un rendimiento aceptable. Las redes neuronales convolucionales, en los últimos años, han propiciado un importante avance en tareas que involucran visión artificial, tales como clasificación, localización, detección y segmentación de objetos, descripción de escenas, entre otras, ya sea en imágenes o vídeo.

Los resultados que se obtienen actualmente se puedan emplear en una gran variedad de aplicaciones. Sin embargo, el desempeño de algoritmos como las CNN en la detección de objetos depende en gran medida de la elección y ajuste de diversos hiperparámetros que pueden determinar la tasa de aprendizaje de las mismas, por tal motivo en este artículo se presenta la combinación de dos técnicas de inteligencia artificial como lo son los algoritmos metaheurísticos y las redes neuronales convolucionales para realizar un entrenamiento más robusto en comparación del entrenamiento estándar de estas redes [13].

Y de esta forma lograr un modelo eficiente y eficaz para la clasificación de objetos. Con el fin de ajustar los hiperparámetros de la CNN se implementan y comparan cuatro metaheurísticas ampliamente utilizados en el estado del arte como son optimización por enjambre de partículas (PSO), algoritmos genéticos (GA), optimización de lobos grises (GWO) y optimización de ballena jorobada (WOA).

2. Marco teórico

2.1. Redes neuronales convolucionales

Las redes convolucionales, son un tipo especializado de red neuronal para procesar datos que tiene una topología similar a una cuadrícula (matriz). Las redes convolucionales han tenido un éxito extraordinario en aplicaciones prácticas. Las redes convolucionales son simplemente redes neuronales que usan la convolución en lugar de la multiplicación general de matrices en al menos una de sus capas.

Estas son muy potentes para todo lo que tiene que ver con el análisis de imágenes, debido a que son capaces de detectar características simples como por ejemplo detección de bordes, líneas, etc. Y extraer características más complejas hasta llegar a su objetivo.

Consta de capas convolucionales y de reducciones alternadas, y en sus capas finales tiene capas de conexión total como una red perceptrón multicapa Figura 1. En la convolución se realizan operaciones de productos y sumas entre la capa de partida y los n filtros que genera un mapa de características. Las características extraídas corresponden a cada posible ubicación del filtro en la imagen original [6].

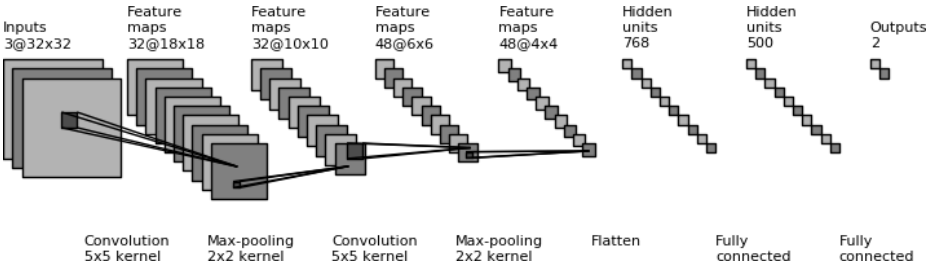


Fig. 1. Arquitectura de las CNN [6].

La ventaja es que el mismo filtro sirve para extraer la misma característica en cualquier parte de la entrada, con esto que consigue reducir el número de conexiones y el número de parámetros a entrenar en comparación con una red multicapa de conexión total [2].

2.2. Metaheurísticas

Las técnicas de optimización metaheurísticas se han inspirado principalmente en conceptos muy simples. Estos algoritmos suelen basarse en fenómenos físicos, comportamientos de animales o conceptos evolutivos. Tienen mayor flexibilidad a diferentes problemas sin ningún cambio especial en la estructura del algoritmo, ya que en su mayoría asumen los problemas como cajas negras.

En otras palabras, solo las entradas y salidas de un sistema son importantes para una metaheurística [7]. Las metaheurísticas tienen capacidades superiores para evitar los óptimos locales en comparación con las técnicas de optimización convencionales.

Esto se debe a la naturaleza estocástica de las metaheurísticas que les permiten evitar el estancamiento en las soluciones locales y adentrarse extensamente en todo el espacio de búsqueda. El cual para problemas reales suele ser desconocido o muy complejo y con una gran cantidad de óptimos locales, por lo que las metaheurísticas tienen buen desempeño únicamente teniendo claro el objetivo [11].

Algoritmos genéticos (GA). El algoritmo genético es una metaheurística inspirada en el proceso de selección natural creado por John Henry Holland en el año 1970, surgió con este algoritmo base de muchas representaciones metaheuristicas [15]. Un algoritmo genético estándar requiere dos requisitos previos, es decir, una representación genética del dominio de la solución y una función de aptitud para evaluar a cada individuo.

La idea central del algoritmo genético es permitir que los individuos evolucionen a través de algunas operaciones genéticas como se muestra en el algoritmo. Las operaciones populares incluyen selección, mutación, cruce. El proceso de selección nos permite preservar a los individuos fuertes mientras eliminamos a los débiles. Las formas de realizar la mutación y el cruce a menudo se basan en las propiedades del problema específico [8].

Optimización por enjambre de partículas (PSO). El algoritmo de optimización con enjambre de partículas (PSO) fue desarrollado por J. Kennedy y R. C. Eberhart [3], el cual se basa del comportamiento de parvadas de aves, colonias de abejas, bancos de peces, entre otros.

Fig. 2. Muestra de la base de datos Natural Images[12].

Se puede utilizar para resolver problemas de optimización que carecen de conocimiento del dominio. La población está constituida por una serie de partículas. Cada uno de ellas representa un individuo. Busca la mejor solución actualizando velocidad y vector de partículas de acuerdo con las ecuaciones (1) y (2). Donde v_{id} representa la velocidad de la partícula i en la d-ésima dimensión, x_{id} representa la posición de la partícula i. P_{id} y P_{gd} son los mejores locales y el mejor global, r_1, r_2 son números aleatorios entre 0 y 1, mientras que w, c_1 y c_2 son peso de inercia y coeficientes de aceleración para explotación y aceleración para los coeficiente de exploración, respectivamente:

$$V_{id}(t+1) = w * v_{id}(t) + c_1 * r_1 * (P_{id} - x_{id}(t)) + c_2 * r_2 * (P_{gd} - x_{id}(t)), \quad (1)$$

$$x_{id}(t+1) = x_{id}(t) + v_{id}(t+1). \quad (2)$$

Optimizador lobo gris (GWO). El algoritmo metaheurístico del lobo gris salió a la luz en 2014 por obra de Seyedali Mirjalili [10]. Donde se muestra el comportamiento de este animal su forma de caza y su conducta social y de particular interés es que tienen una jerarquía social dominante muy estricta donde llamamos a estos grupos como alfa, beta, delta y omega, cada una de estos grupos juega un papel importante en la manada.

Para modelar matemáticamente la jerarquía social de los lobos, consideramos la solución más adecuada como la alfa (a). En consecuencia, la segunda y tercera mejores soluciones se nombran beta (b) y delta (d) respectivamente. Se supone que el resto de las soluciones candidatas como omega (x).

En el algoritmo GWO la búsqueda está guiada por a, b y d. Los lobos x siguen a estos tres lobos y así estos rodean a sus presas durante la caza. Matemáticamente se representa en la ecuación (3) y (4), donde t indica la iteración actual, $\vec{A}$ y $\vec{C}$ son vectores de coeficientes, $\vec{X_p}$ es el vector de posición de la presa, $\vec{X}$ indica el vector de posición de un lobo gris:

$$\vec{D} = |\vec{C} \cdot \vec{X_p}(t) - \vec{X}(t)|, \quad (3)$$

$$\vec{X}(t+1) = \vec{X_p}(t) - \vec{A} \cdot \vec{D}, \quad (4)$$

Tabla 1. Clases y numero de imágenes del dataset Natural Images [12].

Clase	Nombre	Imágenes por clase
1	Avión	727
2	Automóvil	968
3	Gato	885
4	Perro	702
5	Flor	843
6	Fruta	1000
7	Motocicleta	788
8	Persona	986

donde los componentes de $\vec{a}$ se reducen linealmente de 2 a 0 en el transcurso de las iteraciones y r_2, r_2 son vectores aleatorios en [0, 1]:

$$\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a}, \tag{5}$$

$$\vec{C} = 2 \cdot \vec{r_2}. \tag{6}$$

Algoritmo de optimización de ballenas (WOA). El algoritmo de optimización de la ballena jorobada se presenta en 2017 por Seyedali Mirjalili [9]. Se puede interpretar como una modificación al algoritmo del lobo gris (GWO) donde en este caso representa de igual manera su comportamiento de caza, las ballenas jorobadas pueden reconocer la ubicación de sus presas y rodearlas.

Las ecuaciones principales son las descritas en el algoritmo GWO a diferencia del este método, una ecuación en espiral es creado entre la posición de la ballena y la presa para imitar el movimiento en forma de hélice de las ballenas jorobadas dada la siguiente ecuación (9):

$$\vec{X}(t+1) = \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X}*(t), \tag{7}$$

donde $\vec{D'}$ indica la distancia de la ballena a la presa (la mejor solución obtenida hasta ahora), b es una constante para definir la forma de la espiral logarítmica, l es un valor aleatorio de $[-1, 1]$ y $\cdot$ es una multiplicación elemento por elemento.

Aquí se tiene en cuenta el vector donde una ballena crea un círculo que se contrae para llegar a su presa, se asume una probabilidad del 50 por ciento para elegir esta distinción al modelo circular GWO:

$$\vec{X}(t+1) = \begin{cases} \vec{X}*(t) - \vec{A} \cdot \vec{D} & \text{si} \quad p < 0{,}5\ , \\ \vec{D'} \cdot e^{bl} \cdot cos(2\pi l) + \vec{X} * (t) & \text{si} \quad p \geq 0{,}5\ , \end{cases} \tag{8}$$

donde p es un número aleatorio uniforme de [0,1].

Algoritmo 1. Algoritmo híbrido CNN-Metaheurístico

```
Preprocesamiento ← clases Imágenes
modelo ← Se crea el modelo CNN;
while i <= epoch do
    Época CNN ← Ejecución de una época en el entrenamiento CNN
    get weights → hiperparámetros
    Inicio Metaheurístico:
    Parámetros iniciales. ← it
    Ingreso de hiperparámetros al algoritmo ← se evita empeorar la salida
    while j <= it do
        Generación aleatoria de individuos
        Ejecución
        Aptitud de los individuos ← Max f(x) = w1 * Ex + w2 * F1
        Mejores individuos
    end while
     return Hiperparámetros
    set weights ← hiperparámetros
end while
modelo entrenado
Evaluación
     return clasificación de imagen
```

3. Materiales y métodos

3.1. Metaheurístico adaptado para el ajuste fino de hiperparámetros CNN

El objetivo del algoritmo en este artículo es aprovechar el beneficio de la búsqueda exhaustiva de los algoritmos metaheurísticos sobre los hiperparámetros de la red neuronal convolucional centrándose en las capas densas donde se trabaja con una extensa cantidad de hiperparámetros.

Las capas convolucionales en los algoritmos de clasificación como detección de objetos son de suma relevancia. Estas capas juegan un papel importante en la parametrización de las entradas (imágenes) y dan paso a las capas densas para realizar las operaciones necesarias y llegar a la clasificación. Los algoritmos metaheurísticos tienen mecanismos libres de derivación.

A diferencia de los enfoques de optimización basados en gradientes que por su practicidad se utilizan en redes neuronales y redes neuronales convolucionales. Las metaheurísticas optimizan estocásticamente los problemas. El proceso de optimización comienza con soluciones aleatorias y no es necesario calcular la derivada de los espacios de búsqueda para encontrar el óptimo. Esto hace que la metaheurística sea adecuada para problemas reales con información desconocida o compleja [1].

Se tiene claro que la combinación de estos métodos puede realizarse de infinitas formas, por ejemplo: utilizar el metaheurístico después de cada etapa de entrenamiento o época, en determinadas épocas del entrenamiento, únicamente para su ajuste final, por mencionar algunas. Todo esto se determina arbitrariamente dependiendo de la complejidad de la arquitectura, efectividad de convergencia, el tiempo de entrenamiento, entre otras [14].

Tabla 2. Clases y numero de imágenes del dataset.

Capa	Kernel	Parámetros
Conv2d	3×3	640
Max Pooling	2×2	-
Conv2d	3×3	36928
Max Pooling	2×2	-
Flatten	-	-
Dense1	32	663584
Dense2	32	1056
Dense3	8	264

En este caso se examina únicamente el algoritmo descrito en el pseudocódigo del Algoritmo 1 donde al finalizar cada época se ejecutan los algoritmos de optimización para el ajuste fino de estos hiperparámetros sumado las operaciones que se ejecutan a través de las épocas en la CNN. Se genera un punto de partida para homogeneizar las condiciones iniciales de cada algoritmo híbrido iniciando con una época base para todos los algoritmos comparados.

3.2. Base de datos

La base de datos Natural Images empleada en este artículo consta de 6899 imágenes distintas divididas en 8 clases diferentes [12]. En la Figura 2 se muestra una imagen representativa de las imágenes a trabajar observamos que son de diferentes tamaños y estilos. Las clases que contiene el dataset se muestran en la Tabla 2, con su respectivo nombre y el número de imágenes por clase, para poder emplear esta base de datos se toma el total de imágenes y aplica un pequeño pre-procesamiento al normalizar a 80×80 pixeles.

3.3. Implementación del algoritmo híbrido

Parámetros. La arquitectura utilizada en este artículo se basa en la propuesta original ilustrada en la Figura 1 y se representa en la Tabla 2 de manera detallada, similar a la LeNet-5 propuesta por Yann LeCun [6]. Se realizará el ajuste de hiperparámetros en las capas densas durante 10 épocas de entrenamiento. Por lo tanto, los algoritmos descritos requieren parámetros de inicialización para su funcionamiento.

Cada uno de los metaheurísticos se estableció con 30 individuos, 10 iteraciones por cada época de la red neuronal convolucional, una ventana de 25, y una longitud del problema igual a la cantidad de hiperparametros a evaluar después de las capas convolucionales en este caso particular sería de 664,896 variables.

3.4. Función objetivo

En este trabajo se propone una función objetivo a maximizar (Ec. 9) por parte de los enfoques híbridos presentados. Esta, integra la exactitud Ex (Ec. 10) y la medición F1-score $F1$ (Ec. 13), las cuales son dos de las métricas más utilizadas en el campo de la clasificación:

Tabla 3. Evaluación de los algoritmos.

Método	Exactitud	Precisión	Sensitividad	F1-score
CNN	0.8202	**0.9962**	0.1178	0.2054
CNN-GA	0.8224	0.9318	0.0901	0.1605
CNN-PSO	0.8260	0.9740	0.1079	0.1911
CNN-GWO	**0.8355**	0.9740	0.2450	0.3864
CNN-WOA	0.8347	0.9956	**0.4403**	**0.6053**

$$\text{Max } f = w1 * Ex + w2 * F1, \tag{9}$$

donde $w1 = 0.4$ y $w2 = 0.6$, son ponderaciones de la importancia de dichas métricas. Estos valores se obtuvieron mediante pruebas exhaustivas donde se observó que la CNN presenta mejor desempeño usando dicha configuración.

Para fines de comparación, en los resultados se presentan por separado los valores de Exactitud (Ex), F1-score, así como los de precisión (Pre) (Ec. 11) y sensitividad (Sens) (Ec. 12). Métricas que componen la F1-Score [4]. Donde TP es Verdadero Positivo (abreviado por sus siglas en inglés), TF es Verdadero Negativo, FP es Falso Positivo y FN corresponde a Falso Negativo:

$$\text{Ex} = \frac{\text{TP} + \text{TF}}{\text{TP} + \text{TF} + \text{FP} + \text{FN}}, \tag{10}$$

$$\text{Pre} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{11}$$

$$\text{Sens} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{12}$$

$$\text{F1} = \frac{2 \times \text{Pre} \times \text{Sens}}{\text{Pre} + \text{Sens}} = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}}. \tag{13}$$

4. Resultados experimentales y discusión

Una vez implementada la metodología propuesta se obtienen los datos presentados en la Tabla 3 para los máximos valores alcanzados por los 4 algoritmos híbridos y la CNN estándar de las métricas antes descritas. El entrenamiento de la red neuronal convolucional se realizó con una relación 80/20 en los datos de entrenamiento y prueba, se realizaron 10 distintos experimentos hasta alcanzar los máximos valores que podemos apreciar en dicha tabla.

Comparando los modelos híbridos con el método CNN obtenemos que en la precisión los mejores resultados fueron logrados por el método estándar de la CNN con un diferencia de $0.06\,\%$ al segundo mejor y del $6.44\,\%$ al peor. La sensitividad fue mejorada por CNN-WOA en $32.25\,\%$ respecto a CNN. En la evaluación F1-score se destaca nuevamente WOA con una diferencia de $40.0\,\%$. Finalmente para la exactitud se mejoró por el algoritmo CNN-GWO un $1.53\,\%$ al modelo estándar CNN.

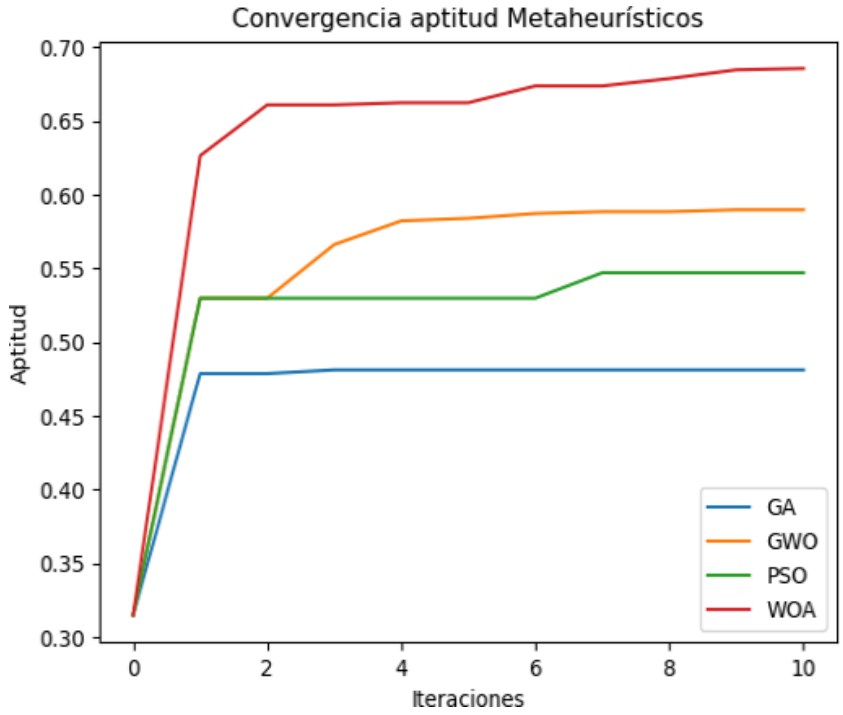


Fig. 3. Comparación de convergencia de métodos metaheurísticos, 1 de 10 épocas.

Como se mostró en los resultados al modificar la arquitectura de la CNN utilizando la metodología híbrida propuesta en este trabajo se observa una mejora en algunas de las métricas pero la más notoria es F1-score lo cual nos indica que el utilizar algoritmos híbridos de inteligencia artificial puede mejorar la robustez del modelo final por lo tanto mejorar la clasificación de imágenes.

Analizando los resultados obtenidos podemos concluir que algunos de los resultados en los modelos hibridos no superaron el modelo estándar, esto podría deberse a la naturaleza de los mismos, debido a las pocas iteraciones en las pruebas realizadas. Y se concluye para la tarea de clasificación aquí presentada el algoritmo con mejor desempeño es CNN-WOA.

En la Figura 1 se representa gráficamente la convergencia al mayor valor de exactitud obtenido en la experimentación a través de las épocas establecidas para cada uno de los métodos. El comportamiento de los algoritmos durante las épocas se intercala utilizando el método CNN y el algoritmo metaheurístico. Podemos notar que existen fluctuaciones en estos cambios siendo la mas notoria en CNN-PSO, a través de las épocas cae la puntuación y en la siguiente itración es recuperada.

Esto podría deberse a la elección de los hiperparámetros de la entrada a la CNN, no logra aprovechar las propiedades del algoritmo híbrido por sus bajas iteraciones en la etapa metahuerística. En la Figura 3 observamos un ejemplo de la convergencia en la primera época de la CNN evaluada por los metaheurísticos, desde esta instancia se aprecia como destaca el algoritmo WOA.

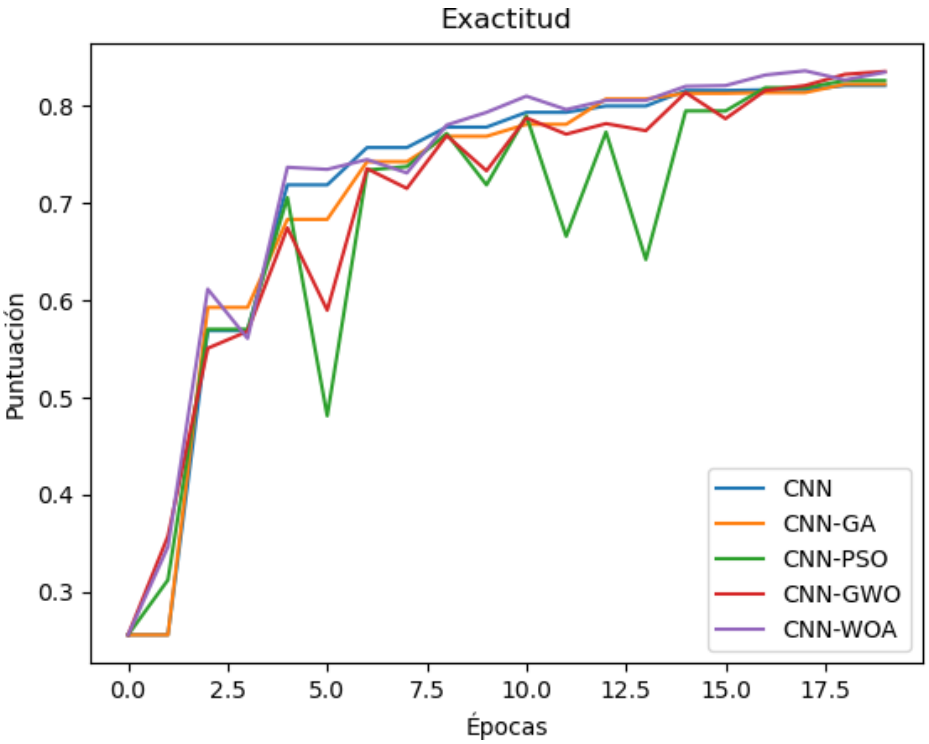


Fig. 4. Gráfica sobre épocas de la Exactitud.

5. Conclusiones y trabajo futuro

A lo largo de esta investigación se analizó el uso de metaheurísticas para el ajuste de hiperparámetros en las redes neuronales convolucionales. Dichos modelos híbridos presentan una mejora considerable para algunas de las métricas utilizadas en el campo de la clasificación como sensitividad, precisión, exactitud y F1. Siendo esta última una de las más relevantes en la clasificación de imágenes por medio de CNNs la cual demostró una diferencia significativa favorable en el modelo CNN-WOA.

La metodología aquí implementada se aventaja de la búsqueda exhaustiva de estos algoritmos de optimización ampliando el espacio de búsqueda. En este trabajo de investigación se experimentó con la intercalación de capas utilizando el método CNN y el algoritmo de optimización con el fin de mejorar el desempeño de la red por época. Como trabajo futuro se plantea realizar el ajuste únicamente en la última capa de la red neuronal convolucional implicando cambios relevantes de los hiperparámetros para mejorar la eficiencia general en la red neuronal.

Acceder a equipo de computo robusto en la cual se puedan realizar mayor número iteraciones en los algoritmos híbridos presentados obteniendo mejor desempeño en las variantes GA y PSO. De igual forma se plantea la implementación de estos modelos híbridos en distintas arquitecturas complejas de las CNN, y aplicarlo en diversas áreas como lo son la detección de objetos en imágenes y vídeo.

Referencias

1. Bonabeau, E., Dorigo, M., Theraulaz, G.: Swarm intelligence: From natural to artificial systems. vol. 1, no. 1 (1999) doi: 10.1093/oso/9780195131581.001.0001
2. Cai, Z., Fan, Q., Feris, R. S., Vasconcelos, N.: A unified multi-scale deep convolutional neural network for fast object detection. In: European Conference on Computer Vision, pp. 354–370 (2016) doi: 10.48550/ARXIV.1607.07155
3. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory. In: Proceedings of the Sixth International Symposium on Micro Machine and Human Science, pp. 39–43 (1995) doi: 10.1109/mhs.1995.494215
4. Hossin, M., Sulaiman, M. N.: A review on evaluation metrics for data classification evaluations. International Journal of Data Mining and Knowledge Management Process, vol. 5, no. 2, pp. 1–11 (2015) doi: 10.5121/ijdkp.2015.5201
5. Krizhevsky, A., Sutskever, I., Hinton, G. E.: ImageNet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems, vol. 25, pp. 1097–1105 (2012)
6. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324 (1998) doi: 10.1109/5.726791
7. Lones, M., Luke, S.: Essentials of metaheuristics (2011) doi: 10.1007/s10710-011-9139-0
8. McCall, J.: Genetic algorithms for modelling and optimisation. Journal of computational and Applied Mathematics, vol. 184, no. 1, pp. 205–222 (2005), doi: 10.1016/j.cam.2004.07.034
9. Mirjalili, S., Lewis, A.: The whale optimization algorithm. Advances in Engineering Software, vol. 95, pp. 51–67 (2016) doi: 10.1016/j.advengsoft.2016.01.008
10. Mirjalili, S., Mirjalili, S. M., Lewis, A.: Grey wolf optimizer. Advances in Engineering Software, vol. 69, pp. 46–61 (2014) doi: 10.1016/j.advengsoft.2013.12.007
11. Osman, I. H., Laporte, G.: Metaheuristics: A bibliography (1996) doi: 10.1007/bf02125421
12. Roy, P., Ghosh, S., Bhattacharya, S., Pal, U.: Effects of degradations on deep neural network architectures (2018) doi: 10.48550/ARXIV.1807.10108
13. Syulistyo, A. R., Purnomo, D. M. J., Rachmadi, M. F., Wibowo, A.: Particle swarm optimization (PSO) for training optimization on convolutional neural network (CNN). Jurnal Ilmu Komputer dan Informasi, vol. 9, no. 1, pp. 52–58 (2016) doi: 10.21609/jiki.v9i1.366
14. Wang, B., Xue, B., Zhang, M.: Particle swarm optimisation for evolving deep neural networks for image classification by evolving and stacking transferable blocks. In: Proceedings of the IEEE Congress on Evolutionary Computation (CEC), pp. 1–8 (2020) doi: 10.48550/ARXIV.1907.12659
15. Weile, D. S., Michielssen, E.: Genetic algorithm optimization applied to electromagnetics: A review. IEEE Transactions on Antennas and Propagation, vol. 45, no. 3, pp. 343–353 (1997) doi: 10.1109/8.558650

ISSN 1870-4069

Implementación de una red neuronal convolucional para el reconocimiento de acciones humanas

Mitchell Ángel Gómez Ortega

Universidad Politecnica del Valle De México,
División de Ingeniería Mecatrónica y Mecánica Automotriz,
México

mitchell.gomez@outlook.com

Resumen. En esta investigación se desarrollan técnicas de deep learning para la implementación de una red neuronal convolucional para el reconocimiento de acciones humanas. Enunciando los principios que rigen a las redes neuronales convolucionales, sus problemas de implementación en sus respectivos tiempos al igual que la evolución tecnológica que han tenido a lo largo de las ultimas décadas. Ademas, considerando situaciones en las cuales el deep learning tiene una alta eficiencia y efectividad en la resolución de problemas en comparación de como lo hace una persona a través de la visión. Uno de los principales problemas en la implementación de deep learning para reconocimiento de acciones humanas esta relacionado a la cantidad de información que se requiere para realizar el entrenamiento en conjunto con el hardware requerido para su procesamiento e incremento de tiempo mediante las GPU's. Hoy en día gracias a las tecnologías de la información y comunicación y al acceso a gran cantidad de información por medio del Internet, este problema se reduce sin dejar de considerar que en ocasiones los datos son variantes tanto en resolución como en dimensiones y si son ocupados para realizar entrenamientos a redes convolucionales puede ocasionar un funcionamiento no deseado durante su implementación. Se propone una arquitectura para la clasificación de 5 acciones humanas tales como (Aplaudir, boxear, caminar, correr y empujar) por medio de la creación de una base de datos actualizada, el uso de googLeNet como red pre-entrenada para la extracción de características y la creación de una red para la clasificación de acciones humanas mediante espacios temporales y memorias de largo y corto plazo bidireccionales, que ayudan a reducir los tiempos de entrenamiento mejorando la precisión por cada clasificación realizada. De esta forma se previene el sobreajuste mediante el uso de algoritmos de entrenamiento y regularización que estabilizan la red y mejoran su precisión durante el proceso de entrenamiento. Finalmente, se realiza una concatenación de googLeNet y la red con memoria a largo y corto plazo para realizar la clasificación de videos de forma secuencial donde se presenta los resultados experimentales de cada una de las acciones con datos que no fueron ingresados para el entrenamiento de la red.

Palabras clave: GoogleNet, HAR, deep learning, Nvidia, RTX 3070, videos, secuencias, acciones humanas.

Mitchell Ángel Gómez Ortega

Implementation of a Convolutional Neural Network for the Recognition of Human Actions

Abstract. In this research, deep learning techniques are developed for the implementation of a convolutional neural network for the recognition of human actions. Enunciating the principles that govern convolutional neural networks, their implementation problems in their respective times as well as the technological evolution they have had in recent decades. Also, considering situations where deep learning has high efficiency and effectiveness in solving problems compared to how a person does it through vision. One of the main problems in the implementation of deep learning for the recognition of human actions is related to the amount of information that is required to perform the training together with the hardware required for its processing and time increase through the GPUs. Today, thanks to information and communication technologies and access to a large amount of information through the Internet, this problem is reduced without forgetting that sometimes the data are variants both in resolution and in dimensions and if they are used to form convolutional networks they can cause unwanted damage. behavior during execution. An architecture is proposed for the classification of 5 human actions such as (clapping, boxing, walking, running and pushing) through the creation of an updated database, the use of googLeNet as a pre-trained network for the extraction of characteristics and the creation of a network for the classification of human actions through temporary spaces and bidirectional long- and short-term memories, which help reduce training times by improving the accuracy of each classification made. In this way, overfitting is avoided by using training and regularization algorithms that stabilize the network and improve its accuracy during the training process. Finally, a concatenation of googLeNet and the network with long and short term memory is performed to sequentially classify the videos, where the experimental results of each of the actions with data that were not entered for training are presented net.

Keywords: GoogleNet, HAR, Deep Learning, Nvidia, RTX 3070, videos, footage, human actions.

1. Introducción

En los últimos años la inteligencia artificial ha revolucionado la forma en que el ser humano desarrolla actividades mientras hace su vida cotidiana y laboral mas sencilla. Muchas de las aplicaciones han sido impulsadas por la innovación tecnológica que se aplica en ordenadores, el internet, el Big Data, etc.

Para desarrollar aplicaciones con inteligencia artificial se requiere implementar técnicas Deep Learning ya que sigue siendo una extensión de las Redes Neuronales Artificiales (ANN) y es una técnica de Machine Learning que emplean las redes neuronales profundas.

Para que las las redes neuronales profundas tuvieran el éxito y la importancia que actualmente tienen, pasaron alrededor de 30 años debido a que no se encontró una

regla de aprendizaje adecuada para la red neuronal multicapa que hacia que durante el entrenamiento la información almacenada en la red neuronal fuera considerada inútil.

En 1986 se resolvió el problema del entrenamiento de la red neuronal multicapa cuando se introdujo el algoritmo de retropropagación (Back Propagation). Sin embargo, al momento de resolver problemas prácticos no cumplió con las expectativas en los diversos intentos de superar sus limitaciones en donde se realizo el incremento de capas ocultas y nodos de las capas ocultas que carecieron de éxito. Lo que ocasiono que se decidiera que la red neuronal no tenia posibilidad de mejora y fuera olvidada durante un largo periodo de tiempo.

A mediados de la década de 2000, cuando se introdujo el aprendizaje profundo las redes neuronales profundas tuvieron una nueva oportunidad aunque tardo tiempo en producir rendimiento suficiente debido a las dificultades para entrenar la red neuronal profunda. En conjunto con las tecnologías actuales en Deep Learning arrojan resultados sorprendentes en rendimiento que supera a otras técnicas de Machine Learning así como a otras redes neuronales, y prevalece en los estudios de Inteligencia Artificial [1].

2. Trabajos relacionados

Dentro de las aplicaciones de la inteligencia artificial con mayor demanda es la clasificación de imágenes y vídeos que se realiza con la implementación de Redes Neuronales Convolucionales (CNN) y algunas de sus aplicaciones es el reconocimiento de acciones humanas (HAR) [2, 3, 4, 5], el uso de la lógica difusa para mejorar el entrenamiento de la CNN en el reconocimiento de acciones humanas [6], la creación de base de datos (*datasets*) con un gran numero de vídeos en combinación con el uso de GPU's y multicores que doten a las CNN de una gran velocidad durante entrenamiento logrando un incremento en el desempeño ocupando el procesamiento en paralelo, y aumentando la velocidad de 10 hasta 12 veces con respecto al procesamiento serial.

[9], clasificar acciones humanas mediante técnicas de esqueletización para el modelo del cuerpo humano en 2D para la obtención de secuencias espaciales [7, 8, 10, 12]. Alternativamente se han empleado acelerómetros como fuente de información para la clasificación de acciones humanas [13, 14], utilizar arquitecturas de CNN propias mejorando la extracción de características para incrementar el nivel de precisión al momento de hacer el entrenamiento y pruebas ocupando algoritmos de Machine Learning [15, 16].

Tomando como punto clave la eficiencia debido a las características de los datos de entrada, en donde representan la información de movimiento y de apariencia, pueden incrementar la precisión ocupando una CNN 3D que tiene como tarea procesar toda la secuencia de vídeo como entra. No obstante, en comparación con la percepción humana con respecto a las acciones de otro, confirman, que en esta tarea específica, las características de movimiento son cruciales.

Esto puede significar que utilizar todo el vídeo puede generar redundancia y ruido en el método de aprendizaje. Por lo tanto, si se logra mitigar la redundancia y ruido se generan resultados superiores que sugieren que las características de movimiento pueden ser más importantes para la tarea de identificar comportamientos agresivos [17, 18, 19, 20].

3. Metodología y materiales

Para realizar una clasificación de acciones humanas mediante secuencias de vídeos, es necesario partir de una red pre-entrenada que permita agilizar y utilizar los pesos pre establecidos para definir las categorías a clasificar y permita partir de los patrones aprendidos para generar nuevo conocimiento, a este proceso se le conoce como *Transfer Learning*.

La razón por la cual se ocupa *Transfer Learning* es para reducir la complejidad en comparación con la creación de una CNN desde cero, las redes pre-entrenadas como: *AlexNet*, *ResNet50* y *GoogLeNet* fueron entrenadas con millones de imágenes que tienen como objetivo clasificar 1000 objetos con sus diferencias entre ellas y ocupando multicores para su aprendizaje y regularización con tiempos de entrenamiento mayores a una semana, las cuales requirieron diversas pruebas para poder llegar a una convergencia ideal.

Esto resulta una gran ventaja debido a que la creación de una CNN desde cero requiere que el programador posea amplia experiencia del tema a resolver, en caso de que el programador carezca de dicha experiencia al diseñar y entrenar la CNN desde cero existe una alta probabilidad de que el modelo no funcione de forma adecuada e inclusive existirían casos donde no se pueda prevenir el *Overfitting* ocasionando que la red en lugar de aprender de las características trate de memorizar todos los datos de entrenamiento, lo cual se vera reflejado que no pueda clasificar y reconocer datos de un escenario real.

3.1. Arquitectura HARNet

GoogLeNet es una red Directed Acrylic Graph (DAG) que se define con capas y conexiones entre las capas y tienen una arquitectura más compleja donde las capas pueden tener entradas o salidas a múltiples capas. Estas arquitecturas permiten entrenar Deep Networks.

Para que GoogLeNet pueda clasificar acciones humanas mediante secuencias de vídeos, es necesario realizar una modificación a la arquitectura de GoogLeNet debido a que solamente esta diseñada para clasificar imágenes.

Por lo tanto, se tiene que implementar una combinación de redes neuronales convolucionales y redes de memoria a largo y corto plazo (*Long Short-Term Memory LSTM*).

Las redes LSTM son diseñadas y utilizadas para aplicaciones en las que la entrada es una secuencia ordenada Figura 1 donde la información anterior en la secuencia puede ser importante, como lo es la clasificación de acciones humanas.

También son un tipo de redes recurrentes que son redes que reutilizan la salida de un paso anterior como entrada para el siguiente paso. Como toda las redes neuronales, el nodo realiza un cálculo utilizando las entradas y devuelve un valor de salida. en redes recurrentes, esta salida se utiliza junto con el siguiente elemento como entradas para el siguiente paso, y así sucesivamente.

El valor de entrada, la salida anterior y el estado interno se utiliza en el calculo de los nodos, los resultados del cálculo se utilizan no solo para proporcionar un valor de salida, sino también para actualizar el estado.

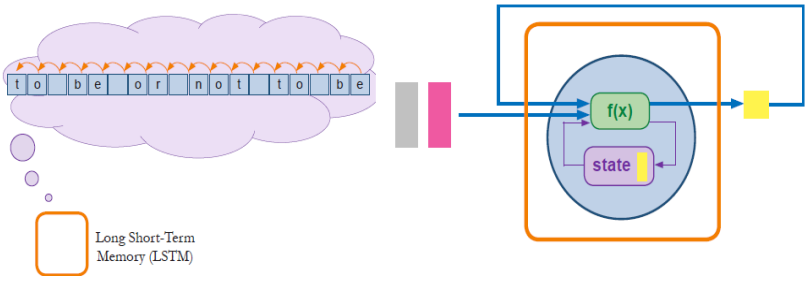


Fig. 1. Principio de funcionamiento de las redes LSTM.

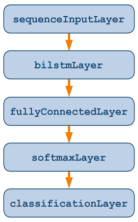


Fig. 2. Arquitectura de una red LSTM.

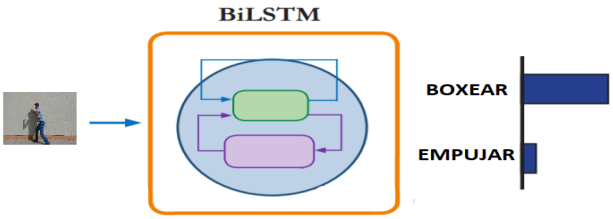


Fig. 3. Principio de la red BiLSTM para acciones humanas.

La arquitectura para clasificar secuencias se almacena en MATLAB como un vector de columnas de capas. Todas las LSTM comienzan con un capa de entrada, siguen con algunas capas LSTM o BiLSTM y terminan con las mismas capas de salida que una CNN como se muestra en la Figura 2.

Las redes LSTM incluyen la capa Bidireccional de memoria a largo y corto plazo (BiLSTM). En muchas situaciones, las BiLSTM pueden lograr una mayor precisión que las LSTM debido que al comienzo de la secuencia tiene el contexto final de la secuencia. Una capa LSTM solo tiene información sobre la datos anteriores en la secuencia.

En los primeros datos, la red no tiene información previa, por lo que la puntuación de predicción solo suele ser 0.5. Más tarde en la secuencia, la red obtiene suficiente información previa para hacer una predicción segura.

Las BiLSTM confía en su predicción a lo largo de la secuencia mientras procesan la secuencia hacia adelante y hacia atrás, por lo que el primer elemento de la secuencia tiene información sobre el resto de la secuencia como se muestra en la Figura 3.

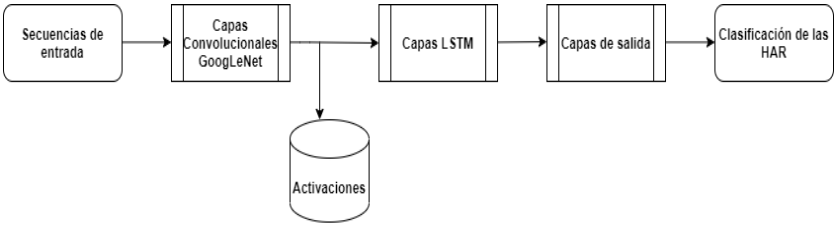


Fig. 4. Arquitectura para la clasificación de acciones humanas.

La unión de GoogLeNet y las capas BiLSTM se representa mediante el diagrama mostrado en la Figura 4, donde se utiliza las capas convolucionales de la red pre-entrenada para la extracción de características de las secuencias de vídeo y el arreglo de red BiLSTM para realizar una clasificación de cada secuencia ingresada por medio del entrenamiento y regularización de la red.

3.2. Categorías y dataset

La base de datos que se creo para el desarrollo de este trabajo fue realizada por 37 personas que se divide en 5 tipos de acciones humanas (Aplaudir, Boxear, Caminar, Correr y Empujar) que se le asigno el nombre de Human Action Recognition (HAR), la cantidad de videos por categoría se muestra en la Figura 5 en donde se seleccionaron diferentes escenarios como (Interiores, Exteriores, Exteriores con diferente ropa y Exteriores con zoom).

La base de datos contiene un total de 3690 vídeos que fueron grabados en ambientes urbanos tratando de conservar la homogeneidad con una velocidad de fotograma de 30 fps. La dimensión de los vídeos es de 320x240 píxeles y con duración exacta de 10 segundos por vídeo en el formato MP4. No obstante, se puede dividir en conjuntos de vídeos para entrenamiento, validación y test sin alterar la estructura de la base de datos. La cantidad exacta de vídeos por categoría se muestra en la Tabla 1.

A diferencia de otras dataset publicas como puede ser KTH Action dataset que esta hecha en escala de grises y con resoluciones muy bajas que ya no satisfacen resoluciones minimas de los dispositivos actuales.

La dataset HAR tiene ventajas y beneficios considerables es que la duración exacta de cada vídeo evita que la red tienda al sobre entrenamiento y baje la precisión debido a que evita la secuencias largas que generan redundancia en el entrenamiento y reduce el coste computacional.

3.3. Configuración de HARNet en MATLAB

La arquitectura mostrada en la Figura 4 y la base de datos se pueden utilizar y configurar en MATLAB mediante el Toolbox de Deep Learning. Para ello es necesario instalar la red pre-entrenada GoogLeNet por medio del Add-Ons de MATLAB como se muestra en la Figura 6. Una vez instalada, es posible cargarla directamente en el Workspace.

Para realizar las modificaciones de GoogLeNet para la clasificación de acciones humanas es necesario leer los vídeos correspondientes a la base de datos HAR.

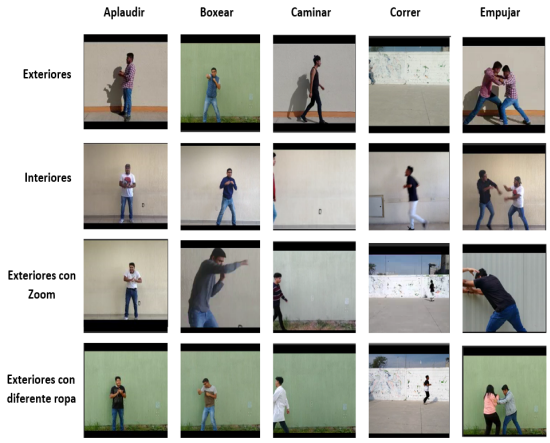


Fig. 5. Ejemplos de Dataset HAR. Se muestran ejemplos de los videos de la base de datos correspondientes a las 5 categorías y los diferentes escenarios en los cuales fueron grabados.

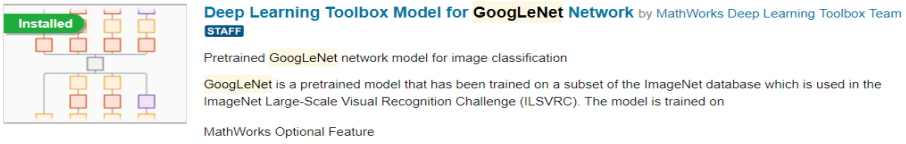


Fig. 6. Red GoogLeNet para MATLAB.

MATLAB no cuenta con una función nativa. Por lo que es necesario almacenar la etiqueta correspondiente al video y (H=Tamaño, W=Ancho, C=Número de canales y S=Número de frames del vídeo) que se almacenan en un vector de 1x4.

Una vez que se adquieren los datos de cada vídeo, se utiliza GoogLeNet como extractor de características para obtener las activaciones de cada vídeo al momento de ingresarlos a la red mediante la conversión de vídeos a secuencias de vectores característicos, en donde dichos vectores son la salida de la función de activación de la capa *pool57x7_s1*.

Se tiene que considerar que al momento de ingresar los vídeos se deben ajustar de acuerdo a los requerimientos de GoogLeNet. Habitualmente el proceso es muy tardado y es proporcional al tamaño de los vídeos de entrada, dado que depende directamente de las características de la computadora, si se llega a ocupara computadoras sin GPU's el proceso de horas pasaría a terminar de ejecutarse en semanas.

Una vez finalizado el proceso, para visualizar cada secuencia se crea una matriz de DxS donde: D=número de características que corresponde al tamaño de la salida de la capa de agrupación y S=número de fotogramas del vídeo.

Adquiriendo las secuencias de vídeo de la base de datos se preparan los datos de entrada en dos secciones para entrenamiento y validación ver en el cuadro 3.2. Correspondientes a la partición aleatoria de las secuencias de vídeo totales al 90 % para el entrenamiento y el 10 % para la validación.

Tabla 1. DataSet HAR.

Categoría	Cantidad de vídeos
Caminar	740
Correr	730
Aplaudir	740
Empujar	740
Boxear	740

Tabla 2. Conjunto de secuencias para el entrenamiento y validación.

Concepto	Cantidad de secuencias
Entrenamiento	3321
Validación	369

De acuerdo con lo mencionado anteriormente, la base de datos tiene la ventaja de reducir el sobre entrenamiento debido a que no existe redundancia por el tiempo de duración de los vídeos ayudando a la red a mejorar la precisión, sin embargo, es posible visualizar las secuencias totales de los datos de entrenamiento mediante histogramas como se muestra en la Figura 7.

En caso de que las secuencias de vídeo fueran variables y excedentes, es posible mejorar la precisión del clasificador eliminando las secuencias que exceden un valor promedio del total de secuencias ya que son mínimas junto con sus respectivas etiquetas.

Una vez procesados los datos para el entrenamiento y validación se requiere una red LSTM para la clasificación de los vectores característicos con las secuencias de vídeo que se procesaron mediante GoogLeNet. La aquitectura de la red LSTM es simple ya que no se requiere pre-procesar los datos de entrada su diseño final se muestra en la Tabla 3.

3.4. Resultados experimentales

Los resultados experimentales que se obtuvieron al entrenar e implementar HARNet en entornos reales con datos desconocidos. La arquitectura de HARNet se diseño en el capitulo anterior con el Toolbox de Deep Learning de MATLAB 2020. El entrenamiento se realizó ocupando una computadora con las siguientes características:

– Procesador AMD Ryzen 5 5600x.
– Tarjeta Gráfica Nvidia RTX3070.
– 64Gb de RAM.
– 1Tb de SSD.

En el proceso de entrenamiento, el conjunto de las 369 secuencias de vídeo fueron separadas para la validación y se realizaron pruebas conforme avanzaba el entrenamiento por cada época, clasificando cada secuencia de vídeo dentro del conjunto de validación.

Tabla 3. Arquitectura de HARNet.

Numero de capa	Nombre	Tipo	Activaciones
1	Secuencias de entrada	Sequence Input	1024
2	BiLSTM	BiLSTM	4000
3	Dropout 50 %	Dropout	400
4	Fc	Fully Connected	5
5	Softmax	Softmax	5
6	HAR Clasification	Classification Output	5

Tabla 4. Opciones de entrenamiento.

Solver	Adam
MinibatchSize	16
LearningRate	0.0001
GradientThreshold	2
Epochs	30

La precisión generada durante la clasificación ocasionaba ajustes en los pesos para reducir la función de perdida e incrementar la precisión de HARNet.

El entrenamiento se realizó en un tiempo 5 horas y 31 minutos en donde se ocuparon las configuraciones que se muestra en la Tabla 4. Se realizaron diferentes pruebas incrementando el numero de épocas con valores de 40, 80, y 100, sin embargo, el entrenamiento obtenía el mínimo error a partir de la época 30. Obteniendo una precisión de validación durante el entrenamiento de 94.31 %.

Para comprender el resultado del entrenamiento se genera una matriz de confusión, que es una tabla que se utiliza para describir el desempeño de un modelo de clasificación de un conjunto de datos de prueba para los que se conocen los valores verdaderos.

En la Figura 8a se muestra la matriz de confusión del entrenamiento en la Figura 8b se muestra la matriz de confusión para el conjunto de secuencias de validación que se considera la precisión real de HARNet generada durante el entrenamiento, donde se clasificaron de forma correcta 348 secuencias de vídeo de un total de 369, ocasionando errores de clasificación en la acción de boxear, correr y empujar.

Se puede inferir que dichos errores corresponden a los generados durante el entrenamiento y por los datos ingresados a HARNet.

El uso de las capa BiLSTM ayudo de forma eficiente en analizar las secuencias de videos de forma rapida, reduciendo los tiempos de entrenamiento y aumentando la precisión aprendiendo de forma adecuada de cada video dentro de la dataset HAR.

4. Conclusiones

La dataset HAR fue creada para entrenar a HARNet lo que incremento la eficiencia debido a su homogeneidad, duración exacta, FPS, la prevención de la redundancia dentro de cada video y la actualización de resoluciones en comparación con bases de datos existentes, creadas con resoluciones 160x120 pixeles en escala de grises que no son compatibles con los dispositivos actuales.

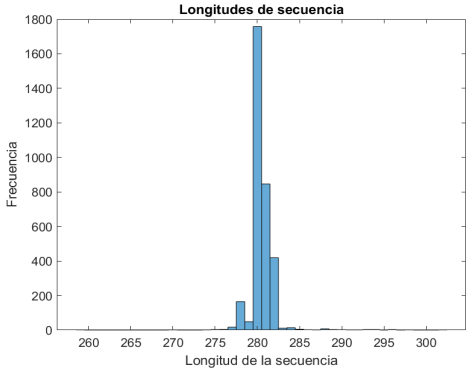
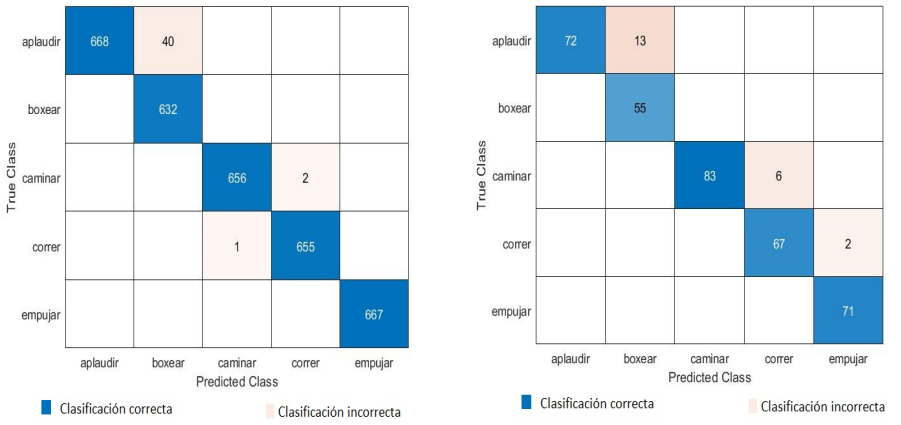


Fig. 7. Longitudes de secuencia para el entrenamiento.



(a) Matriz de confusión del entrenamiento de HARNet. **(b)** Matriz de Confusión de la validación de HARNet.

Fig. 8. Comparativa entre la matríz de confusión del entrenamiento y de validación.

Para realizar el entrenamiento por deep learning, tecnológicamente ya se cuenta con hardware de mayores capacidades como lo son las GPU's que ayuda directamente a reducir los tiempos de entrenamiento de datos del orden de miles hasta millones.

La arquitectura propuesta para HARNet desarrollada en MATLAB 2020a con el toolbox de Deep Learning incluye el uso de la red pre-entrenada googLeNet que funciona como un extractor de características para realizar la conversión de videos a secuencias de video y una red BiLSTM para el entrenamiento y clasificación, que reduce la complejidad, el costo computacional y la memoria ocupada para guardar los pesos creados durante el entrenamiento.

La configuración de las opciones de entrenamiento se realizo la selección de *Adam* como solver principal, los *minibatches* para evitar el sobreajuste, la *validación* y los *parámetros de regularización* de gradiente que generaron precisiones del *98.71 %* para entrenamiento y *94.31 %* para validación.

Mediante los entrenamientos que se realizaron a HARNet se determino que tenia una convergencia adecuada con forme se incremento el números de épocas y que la configuración de las opciones de entrenamiento prevenían el sobreajuste con cada época del entrenamiento.

En las pruebas realizadas a HARNet con datos desconocidos y ambientes no homogéneos, se clasificaron de forma correcta obteniendo niveles bajos de error, dichas pruebas demuestran la robustez de HARNet para la clasificación de las 5 acciones humanas.

La aplicación de esta red se puede trasladar sistemas de vigilancia remotos o embebidos para realizar el monitoreo de comportamientos, acciones agresivas, prevención de delitos entre otras mas, dado que el comportamiento de HARNet en entornos no homogéneos da la posibilidad de incrementar el numero de acciones humanas para su clasificación, sus resoluciones y su posible detección en tiempo real con detectores *You Only Look Once* (*YOLO*) mediante la implementación en sistemas que se encuentren embebidos en cámaras de seguridad con la finalidad de detectar acciones agresivas que pongan en riesgo la salud e integridad en un entorno especifico tales como escuelas, lugares cerrados, centros comerciales etc.

No obstante, incrementando el numero de videos de HAR y ocupando el procesamiento en paralelo con GPU's dotara a HARNet de mayor robustas y fiabilidad en conjunto con las herramientas proporcionadas por MATLAB en versiones futuras.

Referencias

1. Phil, K.: MATLAB deep learning with machine learning, neural networks and artificial intelligence (2017)
2. Valle, E. A., Starostenko, O.: Recognition of human walking/running action based on neural networks. In: 10th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), pp. 239–244 (2013) doi: 10.1109/ICEEE.2013.6676005
3. Ijjina, E. P., Mohan, C. K.: Human action recognition based on MOCAP information using convolution neural networks. In: 13th International Conference on Machine Learning and Applications, pp. 159–164 (2014) doi: 10.1109/ICMLA.2014.30
4. Ijjina, E. P., Mohan, C. K.: Human action recognition based on recognition of linear patterns in action bank features using convolutional neural networks. In: 13th International Conference on Machine Learning and Applications, pp. 178–182 (2014) doi: 10.1109/ICMLA.2014.33
5. Ijjina, E. P., Mohan, C. K.: One-Shot periodic activity recognition using convolutional neural networks. In: 13th International Conference on Machine Learning and Applications, pp. 388–391 (2014) doi: 10.1109/ICMLA.2014.69
6. Ijjina, E. P., Mohan, C. K.: Human action recognition based on motion capture information using fuzzy convolution neural networks. In: Eighth International Conference on Advances in Pattern Recognition (ICAPR), pp. 1–6 (2015) doi: 10.1109/ICAPR.2015.7050706
7. Rajeswar, M. S., Sankar, A. R., Balasubramaniam, V. N., Sudheer, C. D.: Sacling up the training of deep CNNs for human action recognition. In: IEEE International Parallel and Distributed Processing Symposium WorkShop, pp. 1172–1177 (2015) doi: 10.1109/IPDPSW.2015.93
8. Du, Y., Fu, Y., Wang, L.: Skeleton based action recognition with convolutional neural network. In: Proceedings of 3rd IAPR Asian Conference on Pattern Recognition, pp. 579–583 (2015) doi: 10.1109/ACPR.2015.7486569

9. Sun, L., Jia, K., Yeung, D. Y., Shi, B. E.: Human action recognition using factorized spatio-temporal convolutional networks. In: Proceedings of the IEE International Conference on Computer Vision, pp. 4597–4605 (2015)
10. Huang, C. D., Wang, C. Y., Wang, J. C.: Human action recognition system for elderly and children using three stream ConvNet. In: International Conference on Orange Technologies (ICOT), pp. 5–9 (2015) doi: 10.1109/ICOT.2015.7498476
11. Mahasseni, B., Todorovic, S.: Regularizing long short term memory with 3D human-skeleton sequences for action recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3054–3062 (2016)
12. Liu, L., Hu, F., Zhao, J.: Action recognition based on features fusion 3D convolutional neural networks. In: 9th International Synposium on Computational Intelligence and Design, vol. 1, pp. 178–181 (2016) doi: 10.1109/ISCID.2016.1048
13. Lee, S. M., Yoon, S. M., Cho, H.: Human activity recognition from accelerometer data using convolutional neural network. In: IEEE International Conference on Big Data and Smart Computing (BigComp), pp. 131–134 (2017) doi: 10.1109/BIGCOMP.2017.7881728
14. Gammulle, H., Denman, S., Sridharan, S., Fookes, C.: Two stream LSTM : A deep fusion framework for human action recognition. In: IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 177–186 (2017) doi: 10.1109/WACV.2017.27
15. Sargano, A. B., Wang, X., Angelov, P., Habib, Z.: Human action recognition using transfer learning with deep representations. In: International Joint Conference on Neural Networks, pp. 463–469 (2017) doi: 10.1109/IJCNN.2017.7965890
16. Li, J., Wang, T., Zhou, Y., Wang, Z., Snoussi, H.: Using Gabor filter in 3D convolutional neural networks for human action recognition. In: 36th Chinese Control Conference, pp. 11139–11144 (2017) doi: 10.23919/ChiCC.2017.8029134
17. Liu, M., Yuan, J.: Recognizing human actions as the evolution of pose estimation maps. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1159–1168 (2018)
18. Zhou, D., Feng, X., Yi, P., Yang, X., Zhang, Q., Wei, X., Yang, D.: 3D human motion synthesis based on convolutional neural network. IEEE Access, vol. 7, pp. 66325–66335 (2019) doi: 10.1109/ACCESS.2019.2917609
19. Kamel, A., Sheng, B., Yang, P., Li, P., Shen, R., Feng, D. D.: Deep convolutional neural networks for human action recognition using depth maps and postures. IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 49, no. 9, pp. 1806–1819 (2019) doi: 10.1109/TSMC.2018.2850149
20. Wang, P., Yang, Y., Li, W., Zhang, L., Wang, M., Zhang, X., Zhu, M.: Research on human action recognition based on convolutional neural network. In: 28th Wireless and Optical Communication Conference (WOCC), pp. 1–5 (2019) doi: 10.1109/WOCC.2019.8770575

ISSN 1870-4069

Sistema de percepción no centralizado para enjambres de robots RAOI basado en aprendizaje profundo

Erik Ricardo Palacios-Garza, Luis Martín Torres-Treviño

Universidad Autónoma de Nuevo León,
Facultad de Ingeniería Mecánica y Eléctrica,
México

{erik.palaciosgrz, luis.torrestrv}@uanl.edu.mx

Resumen. En el área de la robótica, la implementación de sistemas de percepción es de gran importancia. Actualmente, con el desarrollo del aprendizaje profundo, ha sido posible desarrollar sistemas de visión extremadamente complejos. Sin embargo, en la robótica de enjambres, esta tecnología aún no ha sido adaptada a la percepción de cada robot. Con este trabajo se ha implementado un sistema de visión basado en el aprendizaje profundo en un miembro de un enjambre de robots de configuración diferencial. Esta implementación se llevó a cabo en un NVIDIA® Jetson Nano™ 2GB Developer Kit, utilizando la cámara Raspberry Pi. Los resultados fueron el aumento en la cantidad de estímulos que el enjambre puede detectar, lo que permite el desarrollo de más comportamientos de enjambre.

Palabras clave: Robótica de enjambres, deep learning, percepción, visión artificial.

Decentralized Perception System for RAOI Robot Swarms based on Deep Learning

Abstract. In the area of robotics, the implementation of perception systems is of great importance, currently with the development of deep learning it has been possible to develop extremely complex vision systems, however in swarm robotics this technology has not yet been adapted to the perception of each robot, with this work a vision system based on deep learning has been implemented to a member of a swarm of differential configuration robots, this implementation was carried out on a NVIDIA® Jetson Nano™ 2GB Developer Kit, using the raspberry pi camera. The results were the increase in the amount of stimuli that the swarm can detect, allowing the development of more swarm behaviors.

Keywords: Swarm robotics, deep learning, perception, artificial vision.

1. Introducción

Dentro de la robótica se han implementado diversos sistemas de percepción para generar tareas cada vez más complejas en diferentes áreas de la ciencia, en la medicina se aplica en robots quirúrgicos o mecanismos para la realización de diferentes estudios médicos, en la industria se utiliza en robots controladores de calidad, robots ensambladores entre muchos otros, también dentro de la agricultura, con su implementación en robots controladores de plagas, sin embargo, en la robótica de enjambres los sistemas de percepción han sido limitados a sensores sencillos como ultrasónicos, infrarrojos y sensores de iluminación por mencionar algunos, limitando no solo la percepción de cada robot si no también el desarrollo de comportamientos nuevos.

Los sistemas de percepción han pasado por una gran evolución en los últimos años con la implementación de modelos de redes neuronales en el reconocimiento, clasificación y detección de objetos en imágenes, gracias a esto se ha ampliado el campo de aplicaciones con sistemas de visión, ya que anteriormente las técnicas de procesamiento de imágenes estaban limitadas a condiciones del ambiente en el cual se capturaban las imágenes, por lo que la complejidad del diseño de un sistema de percepción para enjambres de robots basado en procesamiento de imágenes se vuelve enorme, una posible adaptación de estas técnicas es la utilización de modelos centralizados, sin embargo dentro de la robótica de enjambres limita la movilidad y la diversidad de aplicaciones que este tipo de arquitecturas pueden realizar.

La robótica de enjambres trata de resolver problemas complejos usando un conjunto de robots de baja complejidad y bajo costo, como por ejemplo exploración, localización, construcción etc.

Una de las ventajas de trabajar con estas configuraciones es que pueden cumplir con la tarea asignada aun con la pérdida o inclusión de miembros nuevos al enjambre, así como disminuir la complejidad de los algoritmos en cada robot, al emerger comportamientos autónomos del enjambre.

En la mayoría de los casos, como se menciona en [8] estos robots son construidos con un cierto nivel de percepción, el cual consiste en un conjunto de sensores que les permiten tener cierta información del espacio que los rodea, los obstáculos que se podrían encontrar, u objetos con los cuales interactuar, por ejemplo los robots utilizados en [8] utilizan sensores de proximidad y luz para medir la distancia con robots vecinos u objetos, además de obtener información con la luz del lugar donde se encuentren, con esta percepción asignada a cada robot se han obtenido resultados sorprendentes como los mencionados en el artículo, sin embargo tiene la limitante de solo obtener información local de medio ambiente proporcionada por estos sensores.

En [11] se menciona una problemática de percepción similar que en [8] donde la comunicación a corta distancia, representa un reto que propicia el desarrollo de algoritmos más complejos para la robótica de enjambres, presentando la percepción asignada a cada robot como restrictiva.

Existen algunos otros sistemas de percepción para robótica de enjambres más complejos, pero que a su vez proporcionan una mayor capacidad de trabajo a cada robot y al sistema completo, como es el caso de [3], donde utilizan gestos de manos para manipular el comportamiento grupal de los robots, este reconocimiento de gestos manuales está basado en un preprocesado de la imagen capturada de la mano, una

extracción de características y un clasificador de SVM´s. Claramente agregando un sistema de visión se puede observar que se tienen comportamientos que considerando un sistema por sensores serian complicados de obtener.

En el procesamiento de imágenes existen mascaras prediseñadas para la extracción de ciertas características, pero debido a que muchas veces estas mascaras no son suficientemente especializadas para encontrar las características deseadas se recurre a las redes neuronales convolucionales (CNN) las cuales buscan mascaras óptimas para la extracción de características específicas de una imagen y así poder hacer reconocimiento, clasificación o detección de objetos en imágenes [13], comúnmente las CNN constan de una parte donde se utilizan çonvolucionesçon las máscaras antes mencionadas y esta se conecta a un perceptrón [12] para proporcionar una salida de clasificación, siendo esta configuración una de las más utilizadas.

En trabajos posteriores al antes mencionado, se plantea la idea de usar redes neuronales convolucionales [6], dichas redes tienen la ventaja de tener una alta precisión en la tarea de visión que se les asigne, además de poseer gran robustez a las condiciones ambientales donde se capture la imagen a analizar, en la actualidad existe equipo en donde se pueden embeber y utilizar en sistemas robóticos móviles, la desventaja ahora presente es el poder de cómputo que se necesita para su entrenamiento, donde es necesario tanto un gran número de imágenes como computadoras suficientemente potentes para ejecutar el entrenamiento que requieren.

Regresando a [6] se propone un modelo de CNN donde cambian la red de perceptrón por una SVM, dando como resultado una mejora en la precisión de la CNN. Con esta implementación en un enjambre de robots se podrían abrir una serie de tareas y comportamientos más complejos o mejorar las tareas que ya se realizan en la actualidad.

Sin embargo, si en lugar de implementar una CNN entrenada para detectar gestos realizados con las manos, se entrenan para poder percibir su medio ambiente, se podría dar lugar a comportamientos más eficientes y más complejos que podrá realizar el enjambre sin la necesidad de intervención humana, un ejemplo de estas nuevas alternativas a estos comportamientos es la solución a una problemática planteada en [7] donde se menciona que con sensores sencillos un robot no tiene la posibilidad de conocer la orientación de sus vecinos, con la implementación de una CNN se puede abrir esta capacidad a los robots.

2. Sistema de percepción no centralizado para enjambres de robots RAOI basado en Deep Learning

En este trabajo se presenta un sistema de percepción no centralizado aplicable a enjambres de robots, el cual cumple con la función de detectar diversos estímulos de influencia y objetos de interés dentro de las zonas de trabajo de los enjambres, con el aumento drástico en los estímulos de influencia del enjambre este puede generar nuevos y más complejos comportamientos.

El resto del documento se organiza de la siguiente manera: En la siguiente sección se explican los modelos de Deep learning utilizados para la detección correcta de los estímulos de influencia, así como conceptos básicos de enjambres de robots RAOI.

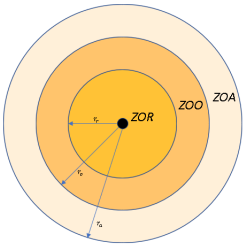


Fig. 1. Zonas de repulsión, orientación y atracción.

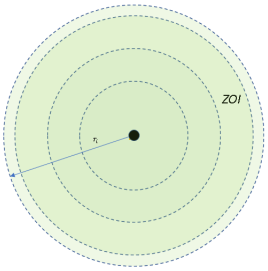


Fig. 2. Zona de influencia.

Luego, se explica el hardware y desarrollo del prototipo robótico de experimentación. Posteriormente, se ilustran los resultados de experimentos de búsqueda de un estímulo de influencia, finalmente se presenta la discusión sobre los resultados, las conclusiones y trabajos futuros.

3. Metodología

En esta sección se explican los elementos teóricos más relevantes para este trabajo, en cuanto al área de robótica de enjambres se mencionan conceptos de áreas RAOI y su generación de comportamientos en enjambres de robots, en seguida se presentan algunos conceptos básicos de las redes de clasificación y detección utilizadas, siendo estas una red residual de 18 capas y una red single shot detector (ssd), finalmente se presenta el comportamiento de experimentación que se utilizó para demostrar la utilidad del sistema de percepción propuesto.

3.1. Enjambres de robots RAOI

El sistema de percepción que se propone, está enfocado para el desarrollo de comportamientos para enjambres de robots RAOI propuestos en Couzin [2], este tipo de comportamientos están basados en características de animales sociales, por ejemplo, abejas, hormigas, lobos, entre otros. Donde se consideran tres características básicas de su movimiento para la generación comportamientos similares a los de ellos.

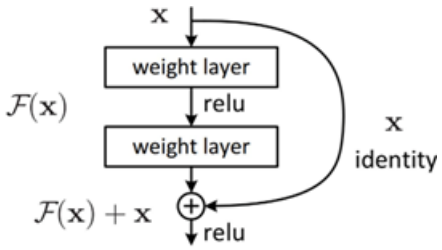


Fig. 3. Modelo básico de un bloque residual [4].

Estas características constan de áreas en las cuales cada animal acostumbra acercarse, alejarse u orientarse, estas fueron nombradas zonas de repulsión, orientación y atracción [10], estas se pueden observar en la figura 1, dentro de la primera zona el individuo trata de alejarse de sus vecinos ya que los considera demasiado cercanos, ya sea para realizar algún movimiento o por una posible colisión, la segunda zona es cuando el individuo considera tener una distancia adecuada de sus vecinos este tratará de orientarse a la misma dirección que ellos, en la última zona si el individuo se encuentra muy distanciado de sus vecinos este tenderá a acercarse a ellos para integrarse al enjambre.

Basándose en estas tres zonas se pueden generar comportamientos similares a los que realizan los animales mencionados anteriormente, sin embargo, en [9] se agregó una nueva zona llamada de influencia la cual se puede apreciar en la figura 2, donde si se tiene la existencia de algún estímulo para el enjambre esta tenderá a realizar la tarea que el estímulo indique, sin embargo debido a las limitaciones que se tienen en los sistemas de percepción de los enjambres propuestos en los trabajos anteriores, el número y tipo de estímulos de influencia se limitaba a las capacidades de los sensores simples que utilizan.

3.2. Redes neuronales convolucionales

Dentro de la visión artificial se trata de cumplir con diferentes objetivos teniendo como principales, el reconocimiento, la clasificación y la detección de objetos, estos objetivos se pueden cumplir con la ayuda de redes neuronales artificiales, las cuales aprenden a extraer características específicas de una imagen para realizar tarea que se desea cumplir, estas redes constan principalmente de los siguientes elementos:

– *Capas convolucionales:* Constan de un mapeo de una matriz de pesos o filtro a lo largo y ancho de una determinada imagen, multiplicando los pesos del filtro por los correspondientes píxeles de la imagen sumando estas multiplicaciones en cada avance dentro del mapeo:

$$C = f\left(I * K + b\right), \tag{1}$$

$$C(i,j) = f\left(\sum_{u=0}^{Fi}\sum_{v=0}^{Co} I(i+u, j+v)\cdot K(u,v) + b\right). \tag{2}$$

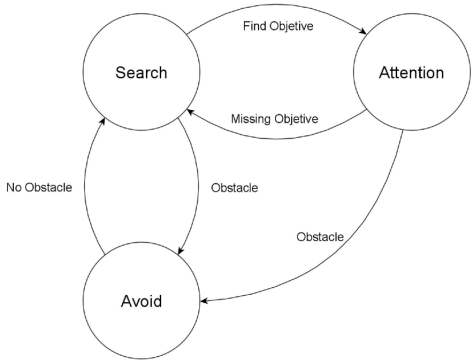


Fig. 4. Comportamiento experimental.

– *Capas de pooling:* Esta etapa redimensiona una determinada imagen para facilitar su procesamiento por capas convolucionales posteriores a esta operación, existen diferentes esquemas de pooling, a continuación, se presenta el más común utilizado, el cual divide la imagen original en grupos de 4 píxeles vecinos, obtiene el promedio de estos cuatro píxeles y los anexa al píxel correspondiente a la vecindad elegida en la imagen original:

$$S(i,j) = \frac{1}{4} \sum_{u=0}^{1} \sum_{v=0}^{1} C(i+u, j+v). \tag{3}$$

– *Padding:* Consiste en agregar un marco de ceros a la imagen de entrada, sirve para realizar una convolución completa a la imagen.
– *Vectorización:* Este proceso consiste en el redimensionamiento de la última capa convolución a un vector para poder ser procesado por la parte densa de la red:

$$v(k) = C(i,j). \tag{4}$$

Para $i = 0, 1, 2...Fi$, $j = 0, 1, 2...Co$.

3.3. Redes neuronales residuales

Durante el proceso de convolución se tiene una pérdida de información al momento de tratar de incrementar el número de capas para mejorar la eficiencia de la red, esta pérdida de información se puede compensar con la utilización de conexiones residuales, las cuales realizan la integración de información de la entrada de una determinada capa a la salida de otra, permitiendo extender la profundidad de capas sin pérdida de porcentaje en métricas de evaluación, el conjunto de todas las capas que salta la conexión residual se le llama bloque residual y al conjunto de bloques residuales se le conoce como red residual o Resnet, en la figura 3 se puede apreciar un bloque residual.

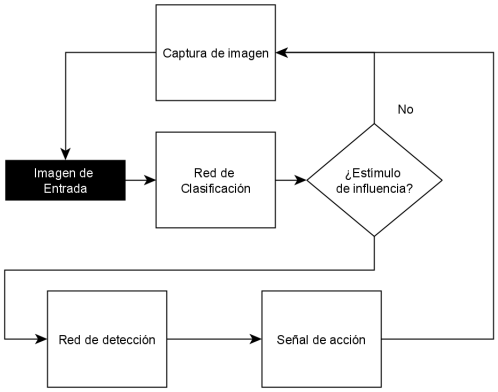


Fig. 5. Esquema de percepción propuesto.

3.4. Redes de detección

La limitación de usar redes de clasificación es que no se puede identificar la posición del objeto en la imagen, posicionar el objeto en la imagen es de gran utilidad para el direccionamiento de los robots, existen diferentes modelos de detección en este caso se utilizara el modelo de SSD (single shot detector) [5].

Esta red se basa en dividir la imagen a analizar en celdas, donde cada celda dará las coordenadas, ancho y largo de una serie de encuadres también llamados “bounding boxes” , los cuales trataran de encerrar el objeto de interés de manera precisa, también cada celda otorga información acerca de la posible existencia de un objeto de interés en la celda y la clase a la cual puede pertenecer el objeto.

3.5. Robots diferenciales

La configuración de robot móvil que se utilizó fue la diferencial, teniendo su modelo matemático basado en A. Bara [1] a continuación:

– Modelo Cinemático:

$$\begin{bmatrix} \dot{x_c} \\ \dot{y_c} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{r}{2}cos(\theta) - \frac{rd}{2R}sin(\theta) & \frac{r}{2}cos(\theta) + \frac{rd}{2R}sin(\theta) \\ \frac{r}{2}sin(\theta) + \frac{rd}{2R}cos(\theta) & \frac{r}{2}sin(\theta) - \frac{rd}{2R}cos(\theta) \\ \frac{r}{2R} & -\frac{r}{2R} \end{bmatrix}. \tag{5}$$

– Modelo Dinámico:

$$\mathbf{M} \cdot \dot{v} + H(v) = \mathbf{B} \cdot \tau, \tag{6}$$

donde:

$$\begin{bmatrix} m & 0 \\ 0 & I_p + m \cdot d^2 \end{bmatrix} \begin{bmatrix} \dot{v} \\ \dot{w} \end{bmatrix} + \begin{bmatrix} -m \cdot d \cdot \dot{\theta}^2 \\ m \cdot d \cdot v \cdot \dot{\theta} \end{bmatrix} = \begin{bmatrix} \frac{1}{r} & \frac{1}{r} \\ \frac{R}{r} & -\frac{R}{r} \end{bmatrix} \begin{bmatrix} \tau_d r \\ \tau_d l \end{bmatrix}. \tag{7}$$

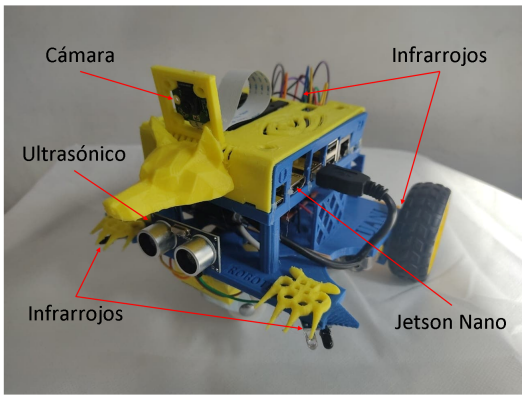


Fig. 6. Prototipo experimental.

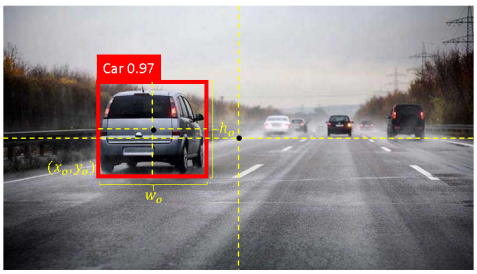


Fig. 7. Obtención de magnitudes.

4. Propuesta de sistema de percepción

4.1. Comportamiento de experimentación

Para poner a prueba el sistema de percepción propuesto en un robot, se utilizó un comportamiento de búsqueda aplicado como una máquina de estados finita ilustrada en la figura 4, compuesta por tres estados, “búsqueda”, “atención” y “evasión”. Donde el estado de búsqueda se encargará de encontrar un estímulo de influencia dentro de la imagen captada por el robot, el estado de atención encuentra la orientación del estímulo de influencia así como una aproximación de cuan alejado esta, por último el estado de evasión se encarga de evitar obstáculos mediante sensores infrarrojos.

4.2. Esquema propuesto de sistema de percepción para enjambres de robots

Teniendo las herramientas del Deep Learning que se mencionaron anteriormente y utilizando los módulos proporcionados por la librería jetson inference, se pueden utilizar en conjunto para poder cumplir con las demandas de percepción que tiene cada robot, por lo que se propone un sistema compuesto por dos redes neuronales, una de clasificación y una red de detección, esto disminuye el coste computacional en

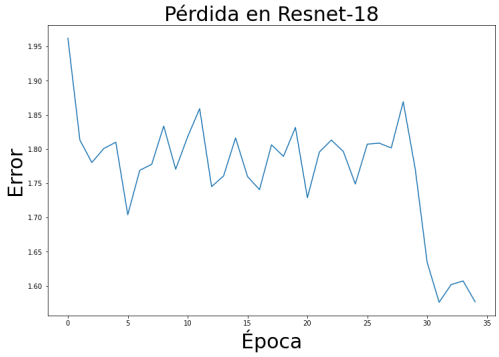


Fig. 8. Evolución del error en la red de clasificación.

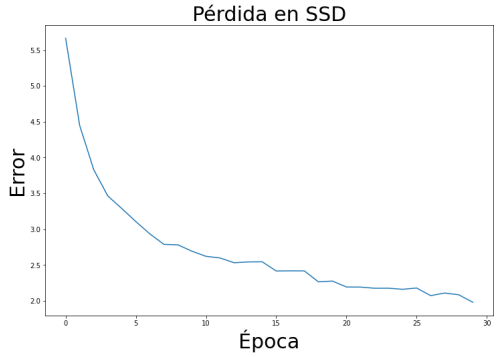


Fig. 9. Evolución del error en la red de detección (épocas 0-50).

la tarjeta utilizada ya que el tiempo de ejecución de la red de detección es muy alto se propone reducir su tiempo activo, por lo que para fines del comportamiento que se desea reproducir en este artículo estará activa la mayor parte del tiempo la red de clasificación y solo en cuanto se detecte una señal de influencia del enjambre se activará la red de detección como se puede observar en la figura 5.

La red de clasificación consiste en una Resnet-18, la cual consta de 8 bloques convolucionales divididos cada uno en dos etapas de convolución con filtros de 3x3, el primer bloque consta de dos etapas de 64 filtros con una imagen de salida de 56x56, a continuación, el segundo bloque con dos etapas de convolución de 128 filtros con una salida de 28x28, el tercero consta de dos etapas de 256 filtros con salida de 14x14 y por ultimo un bloque de 512 filtros con salida de 7x7.

Como ya se mencionó antes se utilizó una red residual de tres bloques como los que se muestran a continuación:

En cuanto a la red de clasificación como ya se mencionó en la sección anterior se utilizará la red SSD, esto por su precisión y velocidad demostradas en la literatura.

Fig. 10. Estímulos de influencia.

4.3. Prototipo experimental Wolfbot

Para poner a prueba el sistema de percepción se sometió a un comportamiento de búsqueda un solo robot como el que se aprecia en la figura 6.

El hardware del robot se describe a continuación:

- *Microcontrolador:* La tarjeta electrónica *Jetson Nano* ™ de NVIDIA® la cual contiene el código computacional del controlador de los motores y del sistema de percepción. Las características más importantes para la selección de esta tarjeta son principalmente, la capacidad de trabajo con redes neuronales y procesamiento de imágenes, además de mantener en la medida de lo posible el rasgo minimalista del hardware de los robots.
- *Motores de CD:* El robot tiene dos motores de corriente directa controlados con controladores PD.
- *Sensores infrarrojos:* Se utilizan para evitar posibles obstáculos que se interpongan con el robot.
- *Sensor ultrasónico:* Cumple la función de la detección de los obstáculos quitándole complejidad a las redes neuronales.

Las pruebas de experimentación se realizaron en un ambiente de dimensiones controladas donde se colocaron 3 estímulos de influencia diferentes para la atracción de cada robot.

4.4. Control

Para lograr el control correcto de los motores se utilizaron dos controladores PD, uno controla la orientación del robot y otro controla la distancia entre el estímulo y el robot, los dos funcionan tomando como referencia la información que entrega la SSD acerca de las bounding boxes que encuadran al estímulo, el controlador de la orientación utiliza como señal de retroalimentación el centro de la bounding box y como señal de referencia el centro de la imagen que recibe, para el controlador de distancia se utiliza como señal de retroalimentación el ancho W_0 de la bounding box y como referencia se coloca el valor de repulsión al estímulo que se desee, un ejemplo de visualización de estos datos es el de la figura 7.

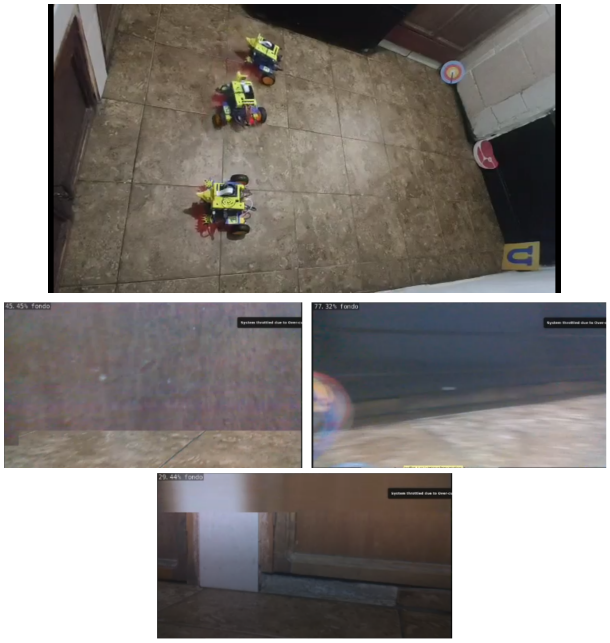


Fig. 11. Muestra de estado de búsqueda.

5. Resultados

5.1. Entrenamiento de red de clasificación

El entrenamiento se realizó con 600 imágenes de 3 diferentes clases: ”Vació”, .ºbjetivo”, Ü”, çarne”, robot.ª 35 épocas, en las figuras 8 se puede apreciar la evolución del error de la red de clasificación a lo largo del entrenamiento.

5.2. Entrenamiento de red de detección

El entrenamiento se realizó con 600 imágenes de 3 diferentes clases: ”Vació”, .ºbjetivo”, Ü”, çarne”, robot.ª 30 épocas, en la figura 9 se puede apreciar la evolución del error de la red detección a lo largo del entrenamiento, así como la evolución del error de regresión y clasificación marcados por la función de error de la red SSD [5].

5.3. Experimentación de comportamiento de búsqueda

La experimentación se realizó utilizando tres robots, sin aplicar reglas de comportamientos colectivos se evaluaron individualmente a cada robot, los experimentos se realizaron colocando estímulos de influencia en diferentes posiciones del área de trabajo de los robots. Estos estímulos de influencia se muestran en la figura 10.

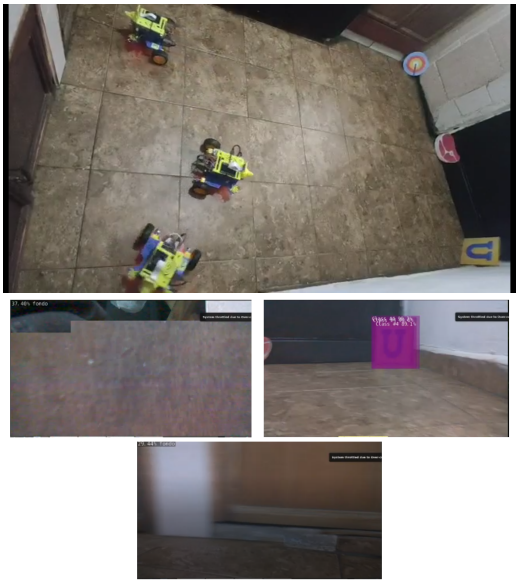


Fig. 12. Muestra de estado de atención.

Estado de búsqueda: En la figura 11 se puede observar el estado de búsqueda de cada robot, donde se encuentra activa la red de reconocimiento la cual se encarga de decir si existe un objeto de interés en la imagen que está observando el robot, como no se aprecia el estímulo de influencia se mantiene en el estado de búsqueda.

Estado de atención: Una vez que se detectado un estímulo de influencia se entra en el estado de atención comenzando a actuar la red de detección como se aprecia en la figura 12 donde se localiza el objeto, se ajusta la orientación del robot hacia el estímulo para a continuación aproximarse hacia él, deteniéndose hasta que se esté por alcanzar la zona de repulsión 13.

Estado de evasión: El estado de evasión no se puede apreciar claramente en las figuras de resultados sin embargo en la figura 12 se puede apreciar como el robot de la parte inferior de la imagen esta por colisionar considerando que el área de trabajo es totalmente cerrada tomando la forma rectangular"que se aprecia en las imágenes, tomando de referencia este robot se puede observar en la figura 13 como ha logrado no colisionar y cumplir con el objetivo de encontrar algún estímulo de influencia.

Se realizaron en total 6 experimentos de los cuales dos fueron considerando un obstáculo en el centro del área de trabajo y un solo estímulo de influencia y dos considerando varios estímulos de influencia, logrando cumplir con el objetivo de aproximarse a un estímulo cada robot.

Fig. 13. Muestra de estado de atención (aproximación).

6. Conclusiones

Los experimentos muestran resultados aceptables cumpliendo con el objetivo del comportamiento de búsqueda, al tener una buena precisión en las redes de clasificación y detección se logra encontrar el estímulo de interés, sin embargo al aumentar el número de estímulos de influencia en la red de clasificación incrementa la dificultad de generalizar el concepto de fondo dentro de la red, es decir lograr que la red de clasificación pueda decir si existe o no un objeto de interés.

La implementación del sistema de visión propuesto mostró que es posible la localización de objetos en tiempo real, utilizando un enfoque descentralizado el cual puede ser utilizado para desarrollar nuevos comportamientos dentro de la robótica de enjambres, dejando de depender de sensores limitados a la medición de magnitudes básicas del entorno del robot, cabe mencionar que estos sensores siguen siendo indispensables para el funcionamiento correcto de un enjambre al contribuir a la detección de señales del medio ambiente del robot que no serían claramente reconocidas por un sistema de visión convencional, como por ejemplo, cambios sutiles en iluminación o sonidos que pudieran ser de utilidad en el desarrollo de comportamientos más complejos, además de poder demostrar que con la implementación de sistemas de visión basados en Deep Learning es posible obtener múltiples estímulos de influencia en el enjambre dando paso al cumplimiento de tareas más diversas por el enjambre, además de notar que la utilización de una o más redes a un sistema de visión de cualquier modalidad, permite mejorar la precisión del sistema de visión, disminuir la carga computacional en su ejecución y disminuir la complejidad del aprendizaje de las redes.

Referencias

1. Bara, A., Dale, S.: Dynamic modeling and stabilization of wheeled mobile robot. In: Proceedings of the 5th WSEAS International Conference on Dynamical Systems and Control (2009)
2. Couzin, I. D., Krause, J., James, R., Ruxton, G. D., Franks, N. R.: Collective memory and spatial sorting in animal groups. Journal of theoretical biology, vol. 218, no. 1, pp. 1–11 (2002)
3. Giusti, A., Nagi, J., Gambardella, L., Di Caro, G. A.: Cooperative sensing and recognition by a swarm of mobile robots. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 551–558 (2012)
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770–778 (2016)
5. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A. C.: Ssd: Single shot multibox detector. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14, pp. 21–37 (2016)
6. Nagi, J., Di Caro, G. A., Giusti, A., Nagi, F., Gambardella, L. M.: Convolutional neural support vector machines: Hybrid visual pattern classifiers for multi-robot systems. In: 2012 11th International Conference on Machine Learning and Applications, vol. 1, pp. 27–32 (2012)
7. Ordaz-Rivas, E., Rodriguez-Liñan, A., Aguilera-Ruíz, M., Torres-Treviño, L.: Collective tasks for a flock of robots using influence factor. Journal of Intelligent & Robotic Systems, vol. 94, pp. 439–453 (2019)
8. Ordaz-Rivas, E., Rodriguez-Linan, A., Torres-Treviño, L. M.: Collective behaviors in swarms of builder robots. Research in Computing Science, vol. 148, no. 11, pp. 103–114 (2019)
9. Ordaz-Rivas, E. D. J.: Colaboración emergente en enjambres de robots, con reglas inspiradas en el comportamiento de animales sociales. Ph.D. thesis, Universidad Autónoma de Nuevo León (2020)
10. Reynolds, C. W.: Flocks, herds and schools: A distributed behavioral model. In: Proceedings of the 14th annual conference on Computer graphics and interactive techniques, pp. 25–34 (1987)
11. Schmickl, T., Möslinger, C., Crailsheim, K.: Collective perception in a robot swarm. In: Swarm Robotics: Second International Workshop, SAB 2006, pp. 144–157 (2007)
12. Zhang, Z.: Derivation of backpropagation in convolutional neural network (cnn). University of Tennessee, Knoxville, TN, vol. 22, pp. 23 (2016)
13. Zhao, Z. Q., Zheng, P., Xu, S. T., Wu, X.: Object detection with deep learning: A review. IEEE transactions on neural networks and learning systems, vol. 30, no. 11, pp. 3212–3232 (2019)

ISSN 1870-4069

CircuitAR: Ambiente inteligente de realidad aumentada para aprendizaje de circuitos eléctricos

Ramón Zatarain Cabada, María Lucia Barrón Estrada,
Aldo Uriarte Portillo, Luis Marcos Plata Delgado

Instituto Tecnológico de Culiacán,
México

{ramon.zc, lucia.be, aldo.up}@culiacan.tecnm.mx,
luis_plata@itculiacan.edu.mx

Resumen. Desde que se estudian las ciencias exactas y la ingeniería, ha existido dificultad para llevar a cabo un aprendizaje efectivo en los temas abstractos y complejos que estas áreas del conocimiento incluyen, afectando la motivación de los estudiantes. Hoy en día, existen muchos sistemas que utilizan la tecnología como medio para gestionar el aprendizaje, sin embargo, no se están aprovechando todas las herramientas disponibles para captar de mejor manera la atención de los estudiantes, así como facilitar la comprensión de la gran variedad de temas abstractos que forman parte de diversas áreas del conocimiento. El trabajo aquí presentado pretende contribuir a la enseñanza del tema "Circuitos eléctricos", facilitando el aprendizaje y la comprensión gracias a las capacidades de visualización e interacción en tiempo real que ofrece la tecnología de la realidad aumentada, además adaptándose al ritmo de aprendizaje de cada estudiante utilizando técnicas de inteligencia artificial.

Palabras clave: Realidad aumentada, lógica difusa, aprendizaje móvil, ambientes inteligentes de aprendizaje.

CircuitAR: Intelligent Augmented Reality Environment for Learning Electrical Circuits

Abstract. Ever since basic sciences and engineering have been studied, there has been tough carrying out effective learning in the abstract and complex topics that these areas of knowledge include, affecting student motivations. Nowadays, many systems use technology as a platform to manage learning, however, not all available tools are being used to achieve the attention of the students, as well as facilitate the understanding of the great variety of abstract topics that are part of these areas of knowledge. The work presented here aims to contribute to the teaching of the topic "Electrical Circuits", facilitating learning and understanding by using real-time visualization and interaction capabilities offered by augmented reality technology, and also by adapting to the learning pace of each student using artificial intelligence techniques.

Keywords: Augmented reality, fuzzy logic, mobile learning, intelligent learning environments.

1. Introducción

En los últimos años, se ha incrementado el uso de nuevas tecnologías dirigidas a mejorar el proceso enseñanza-aprendizaje de los estudiantes, principalmente en las Ciencias, Tecnologías, Ingeniería y Matemáticas (STEM por sus siglas en inglés) para todos los niveles educativos (desde preescolar hasta posgrados), ya sea como actividad de clase o como actividad complementaria [1]. El ser humano cada día usa con mayor frecuencia dispositivos móviles que le permite realizar diversas actividades durante el día, siendo los teléfonos inteligentes los dispositivos que presentan mayor frecuencia de uso [12].

Este aumento de demanda en el uso de teléfonos inteligentes, y la creciente capacidad de cómputo de dichos dispositivos, ha facilitado la integración de tecnologías, tales como los ambientes virtuales tridimensionales, la realidad virtual y la realidad aumentada, las cuales han demostrado ser eficaces para promover el aprendizaje [4]. La realidad aumentada (RA) es una tecnología que complementa la percepción del mundo real de los usuarios a través de una capa contextual de información tridimensional [2], dando al usuario la posibilidad de una interacción en tiempo real con los elementos digitales presentados, brindando al usuario la posibilidad de visualizar elementos abstractos y complejos, difíciles de imaginar.

Por otra parte, la inteligencia artificial ha impactado diversas áreas, y la educación no es la excepción, ya sea al aplicar visión por computadora para evaluar tareas o artículos, evaluar a estudiantes y maestros a través de métodos de aprendizaje adaptativos o enfoques de aprendizaje personalizados, o crear ambientes de aprendizaje interactivos a través de reconocimiento facial, laboratorios virtuales, realidad aumentada o virtual [13].

La principal contribución de este trabajo es el diseño e implementación de un entorno de aprendizaje que, basado en la tecnología de realidad aumentada, apoya a que los estudiantes de ingeniería complementen su proceso de aprendizaje en un tema complejo y de alto nivel de abstracción que es el de circuitos eléctricos. Para lograrlo, se ha combinado la RA con un sistema de lógica difusa, con el fin de guiar al estudiante de manera personalizada a solucionar los diferentes ejercicios que se le presenten, superponiendo información acerca del ejercicio por medio de modelos 3D para representar a los componentes de los circuitos electrónicos y con ello fomentar el aprendizaje activo.

Este artículo está organizado de la siguiente forma: En la sección 2 se presentan los trabajos relacionados acerca del uso de la realidad aumentada en la educación y de lógica difusa aplicada a entornos educativos; en la sección 3 se plantea y se explica la estructura del entorno de aprendizaje; en la sección 4 se presentan los resultados obtenidos de la implementación de la aplicación con estudiantes; y finalmente, en la sección 5 se abordan las conclusiones y se plantean los trabajos futuros.

2. Trabajos relacionados

En esta sección se presentan los trabajos relacionados dentro de la RA enfocados en el aprendizaje de las STEM, y de trabajos que usen lógica difusa centrada en el aprendizaje.

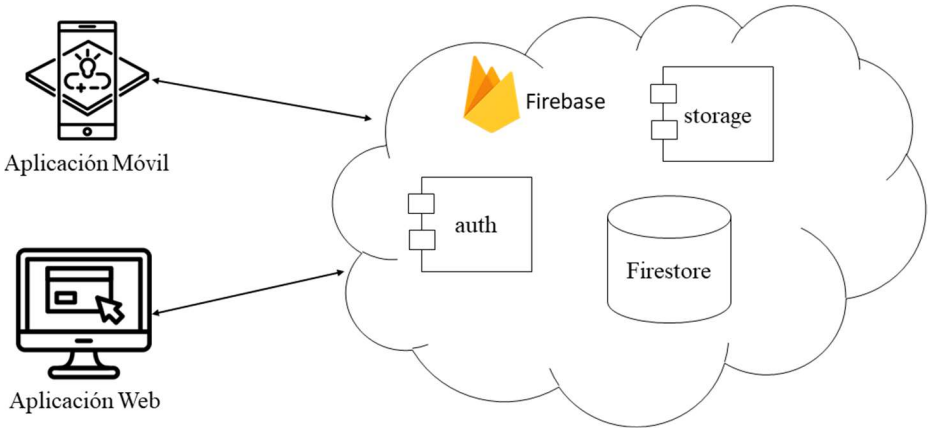


Fig. 1. Arquitectura general de la herramienta de aprendizaje.

Dentro de los trabajos en el área de realidad aumentada enfocados en el ámbito educativo, los trabajos se orientan en mejorar la motivación y la habilidad espacial del estudiante.En el trabajo de Liao et al. [3], se desarrolló un sistema asistente para resolver un cubo de Rubik utilizando realidad aumentada, mostrando pistas y ayudas en el proceso de solución y examinando los efectos en términos de la mejora de los estudiantes, y aprendiendo conceptos de volumen y superficie de cuerpos geométricos.

En el trabajo de [6] (Rossano et al), los autores diseñaron geo+, una aplicación orientada a la solución de problemas de geometría en niños de primaria, destacando la facilidad de uso, habilidades espaciales y la ganancia de aprendizaje de los estudiantes. Por otra parte, en [7], los autores crearon una aplicación orientada a aumentar la eficiencia del aprendizaje con base en leyes de fisica aplicadas en un laboratorio, facilitando así la formación y las actividades cognitivas de los estudiantes, y mejorando la calidad de la adquisición de conocimientos, promoviendo el interés en un tema y el desarrollo de habilidades de investigación.

Ibáñez et al. [4], presentó una revisión del estado del arte en la realidad aumentada, revisando 28 aplicaciones que utilizan esta tecnología para fomentar el aprendizaje STEM, clasificándolos y haciendo un análisis de las mediciones que toman. Una conclusión es que los estudios de los trabajos presentados miden principalmente parámetros afectivos y cognitivos de los estudiantes a través de experimentos transversales, y que existe una necesidad de diversificar las medidas para obtener una comprensión más profunda que vaya más allá de ayudar a recordar hechos y contenido.

En otro trabajo de Ibáñez et al. [11], se evalúa la adquisición de conocimientos de estudiantes en el tema de “electromagnetismo”, comparando estudiantes que usan una plataforma Web contra estudiantes que usan una aplicación móvil usando realidad aumentada. En este trabajo se miden variables relacionadas al flujo de aprendizaje de los estudiantes, concluyendo que los estudiantes que usaron la aplicación de realidad aumentada tuvieron una mayor satisfacción de uso, pero que había aspectos a mejorar en cuanto a la usabilidad en la misma.

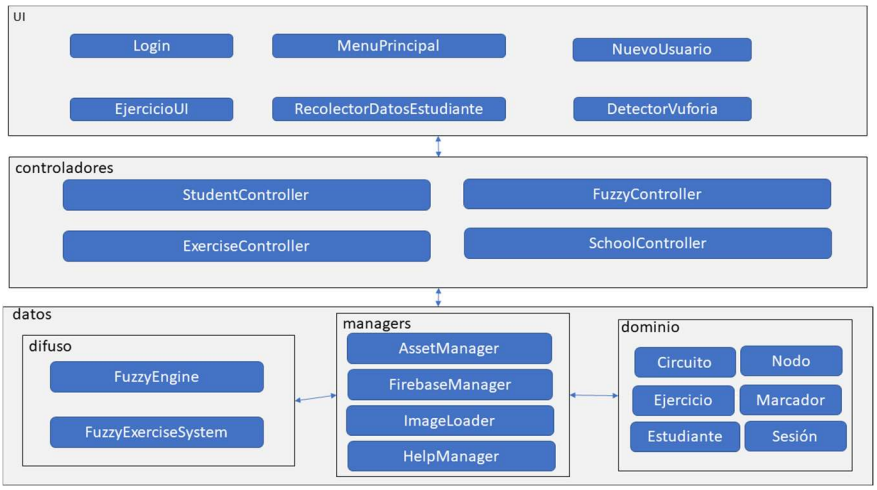


Fig. 2. Arquitectura de aplicación móvil denominada CircuitAR.

Por otra parte, existen también trabajos donde se implementan modelos de lógica difusa para evaluar diferentes aspectos del estudiante en sistemas de aprendizaje. Ozdemir et al. [8] propone determinar el efecto de un juego móvil en la actitud de estudiantes de ingeniería utilizando lógica difusa y variaciones del mismo modelo, concluyendo que para situaciones donde la situación es incierta y para evaluar emociones y pensamientos, la lógica difusa es una técnica efectiva.

Rathore y Jayanthi [14], utilizan un sistema de inferencia difuso para predecir colocación de estudiantes en una escuela utilizando hojas de cálculo y Matlab, y eligiendo la lógica difusa debido a la gran cantidad de datos y variables involucradas, logrando predecir y analizar grandes conjuntos de estudiantes.

En otro estudio, K. O. Gogo et al. [15] desarrollan un modelo que recomienda contenido de aprendizaje relevante a estudiantes, empleando un enfoque de conocimiento del contexto para obtener datos relacionados al estudiante, para después utilizar un modelo de lógica difusa para recomendar contenido de aprendizaje tomado de una base de datos de contenido de libros, tutoriales y videos, consiguiendo reducir el tiempo en el que un estudiante logra obtener contenido de aprendizaje de acuerdo a su nivel de dominio de un tema.

Finalmente, Karaci [9] propone un sistema tutor inteligente que usa lógica difusa para detectar errores mientras los estudiantes hacen cuestionarios, creando un modelo que le permite elegir de manera personalizada las siguientes preguntas que se le plantearan, mejorando el desempeño global de los estudiantes.

3. Estructura del entorno de aprendizaje

CircuitAR es una herramienta de aprendizaje centrada en la solución de problemas de la Ley de Ohm, enfocado al aprendizaje de circuitos eléctricos en estudiantes de primer grado de Ingeniería. CircuitAR le brinda al estudiante elementos

Fig. 3. Interfaz gráfica de CircuitAR.

tridimensionales que le permiten visualizar su forma física, su descripción y la composición del circuito eléctrico usando baterías y resistencias.

Para el desarrollo de CircuitAR, se utilizó una metodología para desarrollo de software de enfoque iterativo e incremental [16], en el cual los requerimientos más importantes son desarrollados en una primera versión de software, y versiones posteriores son liberadas para cumplir los otros requerimientos, permitiendo tomar en cuenta la retroalimentación de las versiones anteriores. El entorno de aprendizaje está integrado por la aplicación móvil y la aplicación web, las cuales se describirán en la siguiente subsección.

3.1. Arquitectura

CircuitAR es una herramienta de aprendizaje desarrollada en Unity 2020 para dispositivos Android, que implementa Vuforia para la RA, y Firebase para la persistencia de datos. Como complemento, se desarrolló CircuitWeb, una aplicación web que se encarga de gestionar toda la la información generada por la interacción de los estudiantes, así como para la descarga de los marcadores.

En la Figura 1 se puede apreciar la arquitectura de la plataforma desde la vista más general. Se puede apreciar en la Figura 1 que la plataforma contiene 2 clientes principales: la aplicación móvil de CircuitAR y la aplicación CircuitWeb. Estos 2 clientes realizan peticiones a Google Firebase, que ofrece diferentes servicios para el desarrollo de plataformas en la nube. Los servicios utilizados son:

- Auth: Registro y autenticación de estudiantes.
- Storage: Almacenamiento de marcadores.
- Firestore: Base de datos NoSQL orientada a documentos. Se utiliza para el almacenamiento de los datos de los estudiantes, ejercicios, marcadores y exámenes aplicados.

Fig. 4. Marcadores convertidos a modelos 3D.

En la Figura 2 se muestra la arquitectura de la aplicación móvil CircuitAR, con sus capas y componentes que la conforman. A continuación, se describen los componentes que integran CircuitAR.

La capa datos contiene componentes que realizan tareas de comunicación con Firebase a través de FirebaseManager, cuya función es administrar, almacenar y obtener los datos necesarios para que el estudiante complete los ejercicios propuestos. ImageLoader obtiene los archivos alojados en dicha plataforma. Toda esta interacción se dará a través de objetos del componente dominio.

Dentro del componente difuso, FuzzyExerciseSystem y FuzzyEngine se encargan de tomar los datos de desempeño del estudiante y aplicar las reglas del modelo difuso. La capa controladores contiene componentes que reciben peticiones de la capa UI, ya sea solicitando o enviando información. Su función es orquestar y organizar la lógica de la aplicación. Esto aplica para cada uno de los módulos de la aplicación, siendo estos: estudiante (StudentController), ejercicios (ExerciseController), escuelas (SchoolController) y el módulo de lógica difusa (FuzzyController).

La capa UI contiene scripts de Unity que muestran al usuario las interfaces gráficas a través de las cuales el estudiante se registra y se autentifica en el sistema. EjercicioUI es el componente con el que el estudiante realiza los ejercicios. Éste se apoya en DetectorVuforia para poder llevar a cabo la realidad aumentada y en RecolectorDatosEstudiante para obtener información acerca de cómo el estudiante está desempeñándose mientras realiza los ejercicios.

3.2. Realidad aumentada

En la capa **UI**, se encuentra el componente **EjercicioUI**, que usa la cámara del dispositivo móvil para poder llevar a cabo la realidad aumentada. Utilizando a **DetectorVuforia** para interpretar la imagen de esta misma y combinando la realidad con elementos virtuales para cada uno de los ejercicios propuestos por la plataforma, la aplicación mostrará un circuito eléctrico incompleto, así como elementos de interfaz gráfica que sirvan al alumno como apoyo didáctico. Ejemplos de estos elementos son

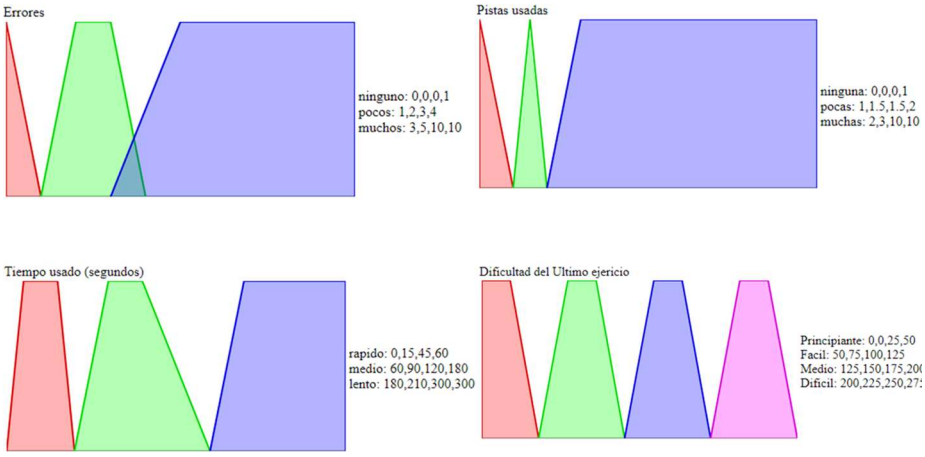


Fig. 5. Variables lingüísticas de entrada del sistema difuso.

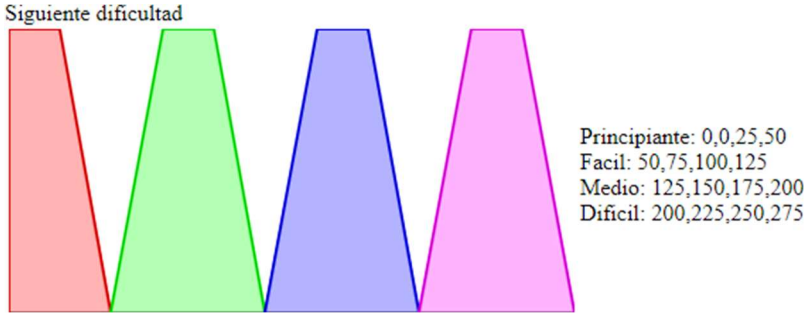


Fig. 6. Variables lingüísticas de salida del sistema difuso.

las ayudas, las instrucciones, el nombre del ejercicio y el contador de tiempo. Estos elementos pueden verse en la Figura 3.

Para que los estudiantes puedan resover los ejercicios, se necesita que el motor de realidad aumentada (Vuforia), efectúe el reconocimiento de un marcador, colocando el mismo dentro del campo de visión de la cámara del dispositivo. Vuforia al reconocer el marcador, superpone los elementos digitales, es decir, los posibles componentes necesarios para completar el circuito mostrado.

Cada marcador es reconocido de manera única, lo cual permite al motor de Unity procesar salidas aumentadas de acuerdo a la posición y orientación del marcador proporcionado por el estudiante[5]. En la Figura 4 se puede ver la interfaz gráfica de CircuitAR mostrando los modelos 3D correspondientes de 3 marcadores que han sido previamente detectados.

3.3. Modelo de lógica difusa

Para poder adaptar el sistema al comportamiento y desempeño del estudiante durante el uso de la plataforma, se implementó un modelo de lógica difusa que evalúa el nivel de complejidad del ejercicio siguiente con base en el número de errores cometidos por

```
SI ultima_dificultad ES principiante Y SI errores ES ninguno Y pistas ES ninguno ENTONCES siguiente_dificultad ES facil
SI ultima_dificultad ES facil Y SI errores ES ninguno Y pistas ES ninguno ENTONCES siguiente_dificultad ES medio
SI ultima_dificultad ES medio Y SI errores ES ninguno Y pistas ES ninguno ENTONCES siguiente_dificultad ES dificil
SI ultima_dificultad ES facil Y SI errores ES muchos Y pistas ES muchas ENTONCES siguiente_dificultad ES principiante
SI ultima_dificultad ES medio Y SI errores ES muchos Y pistas ES muchas ENTONCES siguiente_dificultad ES facil
SI ultima_dificultad ES facil Y SI errores ES muchos Y pistas ES muchas ENTONCES siguiente_dificultad ES principiante
SI ultima_dificultad ES dificil Y SI errores ES muchos Y pistas ES muchas ENTONCES siguiente_dificultad ES dificil
```

Fig. 7. Reglas del sistema difuso.

el estudiante, las ayudas solicitadas y el tiempo empleado para solucionar el ejercicio en turno.

Estas variables lingüísticas y sus funciones de membresía pueden verse en la Figura 5. Sus valores se tomaron de acuerdo con la experiencia de varios profesores consultados, expertos en el área de electrónica. El subcomponente **FuzzyController** (ver Figura 2) selecciona los ejercicios que los estudiantes deben realizar, con base en las recomendaciones realizadas por la máquina de inferencia difusa, y brindar retroalimentación al estudiante. La máquina de inferencias difusa adapta el modelo pedagógico a las necesidades del alumno. Esta máquina contiene las variables lingüísticas, los conjuntos difusos y las etiquetas.

Las variables difusas o lingüísticas de entrada son el número de errores cometidos por el alumno, el número de veces que el alumno solicita ayuda y el tiempo necesario para resolver el ejercicio. El resultado de la inferencia es una variable difusa de salida denominada "siguiente dificultad, que representa al nivel de dificultad aplicable al siguiente ejercicio con los valores difusos de principiante, fácil, medio y difícil. El subcomponente **AssetManager** (ver Figura 2) administra los accesos a las bases de datos de modelos prefabricados, y otros recursos para configurar el ejercicio actual según el del usuario.

El subcomponente **HelpManager** administra los mensajes de ayuda que se le presentarían al estudiante para resolver los ejercicios. Una vez definidas las variables lingüísticas que se pueden ver en las Figuras 5 y 6, el sistema difuso aplicará las reglas para cada conjunto de valores de las 3 variables de entrada, dado que se tienen 4 variables de entrada, 3 de ellas con 3 posibles valores y una con 4, se han definido 81 reglas, cada una dando un valor correspondiente a la variable de salida. Algunas de las reglas utilizadas se muestran en la Figura 7.

4. Evaluación y experimentos

Para evaluar la efectividad de CircuitAR, se consideraron 2 aspectos: la funcionalidad de la plataforma y una intervención con estudiantes del segundo semestre de la carrera de ingeniería electrónica, del Instituto Tecnológico de Mazatlán, en modalidad virtual a distancia. Para ambos aspectos, nuestro sistema se adaptó por completo a la pandemia para realizar todos los experimentos y evaluaciones a distancia y en línea. Esto se logró con las siguientes acciones:

– Los marcadores para RA están disponibles en CircuitWeb, ya sea para su impresión o su visualización desde la web.

Tabla 1. Comentarios relevantes por parte de los estudiantes.

Alumno	Comentarios
Alumno 1	Es una gran aplicación y es de gran utilidad, son temas de nivel ingeniería y por lo tanto es de mucha relevancia hacer una buena aplicación para la enseñanza de esta materia.
Alumno 2	Creo que la aplicación aporta demasiado, siento que aporta demasiado a las prácticas de circuitos eléctricos y ahora con el tema de la contingencia sanitaria es de gran utilidad para esos docentes que imparten esa materia.
Alumno 3	Fuera de algunos detalles en cuanto a la usabilidad, me pareció útil y con buen contenido didáctico, espero que en el futuro le agreguen más ejercicios y componentes de circuitos eléctricos.
Alumno 4	Estamos tan acostumbrados a las tecnologías aplicadas mediante apps en dispositivos móviles, que con la ayuda de estas aplicaciones orientadas a la enseñanza de materias facilitaría más el trabajo a la hora de aprender y practicar, ya que no todos cuentan con los recursos de obtener componentes o piezas para realizar una práctica de manera física.
Alumno 5	Debido a la situación actual provocados por la contingencia sanitaria, muchas escuelas se vieron limitadas a la hora de las prácticas de forma física, como lo son los circuitos eléctricos y pienso que esta aplicación será de gran utilidad para que los alumnos realicen prácticas desde donde se encuentren.

- La aplicación está disponible para su descarga en la Google Play de Android1, evitando problemas de distribución y de permisos en los equipos, así como aumentando la confianza de los estudiantes al estar en un ambiente conocido por ellos.
- Los exámenes pre-test y post-test están también disponibles en CircuitWeb.

Estas acciones permitieron llevar un mejor control del experimento, logrando así que las sesiones con los estudiantes se pudieran llevar a cabo de manera remota, así mismo, que los exámenes de evaluación pre-test y post-test estén disponibles dentro de la plataforma, les dio a los estudiantes un mayor compromiso a realizarlas, ya que de esta manera no tenían que salir de la aplicación para poder contestarlos, mejorando el compromiso y confianza de los estudiantes con el proceso.

En cuanto a la funcionalidad de la plataforma, se recibió retroalimentación de los alumnos durante las sesiones de uso de la herramienta. Para ello, se recopilaron observaciones que nos permitieron mejorar el ambiente de aprendizaje para futuras iteraciones del desarrollo. En la tabla 1 se muestran las más relevantes de ellas.

En general, se puede apreciar que los alumnos coincidieron en que las interfaces de usuario de los ejercicios de CircuitAR son innovadoras y usables (sencillas de usar), y

[1] https://play.google.com/store/apps/details?id=com.mcc.circuitar

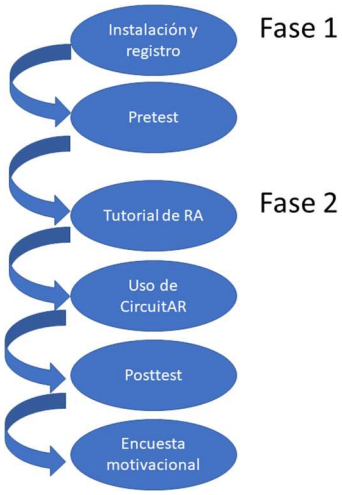


Fig. 8. Proceso de pruebas de la aplicación CircuitAR.

que es una manera innovadora de acercarse al tema de circuitos eléctricos sin la necesidad de tener componentes físicos.

Además de las retroalimentaciones, y siguiendo una de las metodologías propuestas por Woolf [9], se diseñó una intervención que consta de 2 fases: en la primera fase el estudiante debe descargar e instalar la aplicación. Una vez instalada, el estudiante procede a registrarse. Al iniciar sesión, se habilita la opción para responder un examen pre-test con una duración de 15 minutos, integrado por diez reactivos, con cuatro posibles respuestas acerca del tema de circuitos eléctricos.

La segunda fase se inicia con un tutorial acerca del uso de la tecnología de realidad aumentada a través de videos, así como de un ejemplo del manejo de la técnica de colisión de marcadores, con una duración de 10 minutos. Posteriormente, se brindan las instrucciones acerca de cómo interactuar con la aplicación CircuitAR, y durante 20 minutos, los estudiantes deberán resolver los ejercicios propuestos en la aplicación.

Durante ese tiempo, el equipo de investigación atiende y resuelve dudas acerca del manejo de la aplicación. Una vez finalizada la interacción, el estudiante deberá responder un post-test con una duración de 15 minutos, integrado por diez preguntas con el mismo grado de complejidad del pre-test. Para finalizar la sesión, el estudiante deberá responder una encuesta motivacional, integrada por 36 preguntas con 5 posibles respuestas, en un lapso de 20 minutos.

La Figura 8 muestra la metodología aplicada en el proceso de pruebas y experimentos. En este proyecto participaron 24 estudiantes de segundo semestre de ingeniería (edad=19-23, M=19.2380, SD=1.338), en donde 3 estudiantes no completaron algunas de las pruebas y, por lo tanto, no fueron considerados para este estudio. Al realizar el análisis del examen pre-test, se obtuvo una media de 51.4 de calificación y un porcentaje de aprobados de 38.1%. De manera similar, con el examen post-test se obtuvo un 65.2 de promedio y un porcentaje de aprobados de 66.7%. La Figura 9 muestra los resultados obtenidos. De la gráfica anterior se puede observar una mejora de 28.6% antes y después de los exámenes.

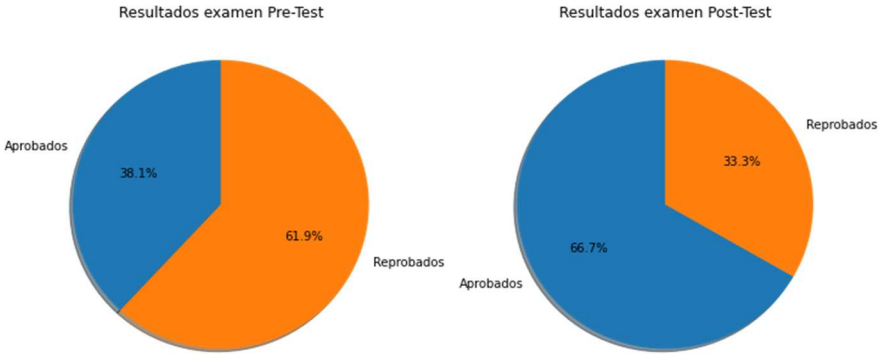


Fig. 9. Resultados de los exámenes pre-tests y post-tests.

5. Conclusiones y trabajo a futuro

La plataforma desarrollada es capaz de guiar a los alumnos a través de ejercicios de realidad aumentada de manera personalizada gracias al modelo de lógica difusa que se ha diseñado. A partir de la evaluación cualitativa aquí mostrada por los alumnos al usar la aplicación, se puede concluir que los alumnos se han sentido identificados con los elementos mostrados por la aplicación, la cual es innovadora en la manera en que ésta muestra los componentes con los que los estudiantes interactúan durante sus prácticas didácticas.

Así mismo, los estudiantes hicieron observaciones acerca de la usabilidad y el rendimiento de la aplicación en distintos dispositivos, ayudando esto último a detectar y resolver algunos problemas de resolución y rendimiento dentro de CircuitAR.

Por otro lado, y dados los resultados de las pruebas realizadas, se puede considerar que CircuitAR como entorno de aprendizaje, es efectivo para mejorar el rendimiento de los estudiantes en la comprensión del tema de "circuitos eléctricos", al ponerlo en práctica con los ejercicios que se han planteado.

Como trabajos futuros, se planea llevar a cabo pruebas en grupos con mayor cantidad de estudiantes, así como en grupos de diferentes grados y carreras, con el fin de poder analizar mejor el impacto de la herramienta en contextos diferentes al planteado en este artículo.

También se planea ampliar la propuesta didáctica de la plataforma, aumentando la cantidad de ejercicios disponibles, así como la variedad de elementos eléctricos que los estudiantes utilizan para interactuar con la plataforma. Finalmente, se realizará la interpretación de los resultados de las encuestas motivacionales utilizando algún instrumento de medición adecuado.

Referencias

1. Alshamrani, A., Bahattab, A.: A comparison between three SDLC models waterfall model, spiral model, and Incremental/Iterative model. International Journal of Computer Science Issues, vol. 12, no. 1, pp. 106–111 (2015)

2. Azuma, R. T.: A survey of augmented reality. Presence: Teleoperators and Virtual Environments. MIT Press-Journals, vol. 6, no. 4, pp. 355–385 (1997) doi: 10.1162/pres.1997.6.4.355
3. Gogo, K. O., Nderu, L., Mwangi, R. W.: Fuzzy logic based context aware recommender for smart E-learning content delivery. In: 5th International Conference on Soft Computing and Machine Intelligence (2018) doi: 10.1109/iscmi.2018.8703247
4. Gonzalez, H. B., Kuenzi, J. J.: Science, technology, engineering, and mathematics (STEM) education: A primer. Washington, DC: Congressional Research Service, Library of Congress (2014)
5. Holmes, W., Bialik, M., Fadel, C.: Artificial intelligence in education: Promises and implications for teaching and learning (2019)
6. Hruntova, T. V., Yechkalo, Y. V., Striuk, A. M., Pikilnyak, A. V.: Augmented reality tools in physics training at higher technical educational institutions. In: Proceedings of the 1st International Workshop on Augmented Reality in Education, no. 2257, pp. 33–40 (2018)
7. Ibáñez, M. B., Di Serio, Á., Villarán, D., Delgado Kloos, C.: Experimenting with electromagnetism using augmented reality: Impact on flow student experience and educational effectiveness. Computers and Education, vol. 71, pp. 1–13 (2014) doi: 10.1016/j.compedu.2013.09.004
8. Ibáñez, M. B., Delgado-Kloos, C.: Augmented reality for STEM learning: A systematic review. Computers and Education, vol. 123, pp. 109–123 (2018) doi: 10.1016/j.compedu.2018.05.002
9. Karaci, A.: Intelligent tutoring system model based on fuzzy logic and constraint-based student model. Neural Computing and Applications. Science and Business Media LLC, vol. 31, no. 8, pp. 3619–3628 (2018) doi: 10.1007/s00521-017-3311-2
10. Liao, Y. T., Yu, C. H., Wu, C. C.: Learning geometry with augmented reality to enhance spatial ability. In: Proceedings of the International Conference on Learning and Teaching in Computing and Engineering (2015) doi: 10.1109/latice.2015.40
11. Ozdemir, A., Balbal, K. F.: Fuzzy logic based performance analysis of educational mobile game for engineering students. Computer Applications in Engineering Education, vol. 28, no. 6, pp. 1536–1548 (2020) doi: 10.1002/cae.22325
12. Patil, S., Prabhu, C., Neogi, O., Joshi, A. R., Katre, N.: E-learning system using augmented reality. In: 2016 International Conference on Computing Communication Control and automation (2016) doi: 10.1109/iccubea.2016.7860038
13. Rathore, R. K., Jayanthi, J.: Student prediction system for placement training using fuzzy inference system. Journal on Soft Computing, vol. 7, no. 3, pp. 1443–1446 (2017) doi: 10.21917/ijsc.2017.0199
14. Rossano, V., Lanzilotti, R., Cazzolla, A., Roselli, T.: Augmented reality to support geometry learning. vol. 8, pp. 107772–107780 (2020) doi: 10.1109/access.2020.3000990
15. Woolf, B. P.: Building intelligent interactive tutors: Student-centered strategies for revolutionizing e-learning (2010)

ISSN 1870-4069

Reconocimiento automático de personalidad aparente contra prueba estandarizada

Ramón Zatarain Cabada[1], María Lucía Barrón Estrada[1], Hugo Jair Escalante[2], Héctor Manuel Cárdenas López[1], Víctor Manuel Bátiz Beltrán[1]

[1] Instituto Tecnológico de Culiacán, Posgrado e Investigación, México

[2] Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla, México

{ramon.zc,lucia.be,victor.bb}@culiacan.tecnm.mx, hugojair@inaoep.mx, hector_cardenas@itculiacan.edu.mx

Resumen. En el ámbito del reconocimiento automático de la personalidad se han realizado diversos estudios que alcanzan diferentes niveles de certeza con base a conjuntos de datos de video y voz previamente etiquetados; por otra parte, existen pruebas estandarizadas de personalidad que permiten con base a un modelo de factores de la personalidad determinar el nivel de desarrollo de cada factor en una persona. Sin embargo, no existe una plataforma que permita al investigador poder recolectar por una parte nuevos conjuntos de datos de video y voz y así mismo, permita aplicar una prueba de personalidad estandarizada y almacenar dicha información para posteriormente evaluar la certeza de los reconocedores automáticos aplicados a los conjuntos de datos recolectados. Por lo anterior, el presente trabajo muestra el desarrollo de una plataforma de recolección de datos para poder realizar análisis de la efectividad de reconocedores automáticos de la personalidad con respecto a los resultados de una prueba estandarizada de personalidad del mismo participante y de esta forma, contar con elementos que permitan la mejora de los modelos evaluados.

Palabras clave: reconocimiento automático de personalidad, aprendizaje profundo, plataforma web, pruebas estandarizadas de personalidad.

Automatic Recognition of Apparent Personality Against Standardized Test

Abstract. In the area of automatic personality recognition, various studies have been carried out that reach different levels of certainty based on previously labeled video and voice data sets; On the other hand, there are standardized personality tests that allow, based on a model of personality factors, to determine the level of development of each factor in a person. However, there is no platform that allows the researcher to collect new video and voice data sets and likewise, allow to apply a standardized personality test and store this information to later evaluate the accuracy of the automatic recognizers applied to the collected data

sets. Therefore, the present work shows the development of a data collection platform to be able to perform analysis of the effectiveness of automatic personality recognizers with respect to the results of a standardized personality test of the same participant and in this way, to have elements that allow the improvement of the evaluated models.

Keywords: Automatic recognition of personality, deep learning, web platform, standardized personality tests.

1. Introducción

En la actualidad, conocer los rasgos de personalidad es importante porque permite entender mejor a los individuos y con ello adaptar de una mejor forma los procesos de enseñanza o colocar a la persona en el puesto adecuado dentro de una organización, buscando lograr el mayor impacto posible en el aprendizaje o desempeño del individuo en su proceso cognitivo o práctica profesional.

En los últimos años los modelos más utilizados y aceptados para determinar la personalidad con base a pruebas escritas, son los modelos basados en rasgos y específicamente el modelo conocido como los cinco grandes (Big-Five).

Este modelo es usualmente representado por el acrónimo OCEAN donde cada letra se refiere a un término en inglés que representa cada uno de los cinco grandes rasgos de personalidad: Apertura a la experiencia (Openness to Experience), Responsabilidad (Conscientiousness), Sociabilidad o Extroversión (Extraversion), Amabilidad (Agreeableness) y Neuroticismo o Estabilidad Emocional (Neuroticism) [1].

Uno de los esfuerzos más relevantes en cuanto a la definición de los reactivos o preguntas (ítems, como se les conoce en el ámbito de la sicología) a utilizar para el modelo de los cinco grandes, es el realizado por el International Personality Item Pool (IPIP), el cual podemos considerarlo como un laboratorio científico para el desarrollo de medidas avanzadas de rasgos de personalidad y otras diferencias individuales que son del dominio público gracias a su sitio Web [2].

Dicho sitio mantiene un inventario de miles de ítems y cientos de escalas para la medición de los rasgos de personalidad y en general se basan en los estudios realizados por Goldberg [3, 4, 5]. Por otra parte, en los últimos años, se han realizado investigaciones que buscan implementar reconocedores automáticos de la personalidad por medio del aprendizaje automático.

Estos estudios se han encaminado a utilizar el modelo de los cinco grandes para detectar personalidad aparente con base en características en el texto, la voz o rasgos faciales. El principal reto que enfrentan estas investigaciones es la dificultad de contar con un conjunto de datos (dataset) representativo y, por otra parte, la necesidad de etiquetar las imágenes y el hecho de que típicamente estos esfuerzos no son del dominio público y por lo tanto es complicado reproducir sus resultados [1].

La principal contribución de este trabajo es el desarrollo de un ambiente integrado que permita valorar los rasgos de personalidad de un individuo mediante el uso de una prueba estandarizada, basada en el modelo de los cinco grandes y también permita capturar interacciones de video para usar reconocedores automáticos que busquen determinar los mismos aspectos de personalidad. Lo anterior, con la finalidad de poder evaluar la eficacia de dichos reconocedores automáticos con respecto a la prueba

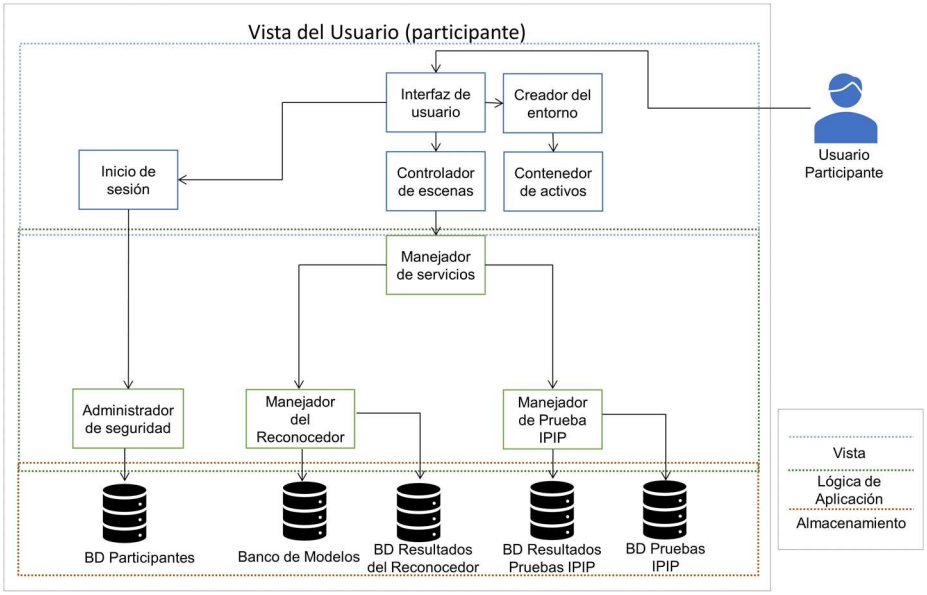


Fig. 1. Vista lógica de la plataforma.

estandarizada y contar así con información relevante para mejorar los modelos utilizados por los reconocedores.

Este artículo está estructurado en el siguiente orden: en la Sección 2 presentamos los trabajos relacionados en las áreas de pruebas estandarizadas y reconocimiento automático de la personalidad; en la Sección 3 se presenta un análisis de la plataforma de recolección de datos propuesta; en la Sección 4 se plantea el flujo de trabajo para la recolección de datos utilizado; en la Sección 5 se describen los experimentos, pruebas y resultados, y finalmente en la sección 6 se presentan las conclusiones y trabajos futuros.

2. Trabajos relacionados

En esta sección describimos algunos trabajos de investigación relacionados con el área de las pruebas estandarizadas y del reconocimiento automático de la personalidad. Estos trabajos, aunque son esfuerzos separados, guardan similitud con elementos del presente trabajo de investigación y fueron considerados como base para el desarrollo e integración de este proyecto.

Diversos estudios han demostrado que uno de los mejores enfoques para la detección de la personalidad es el modelo de los cinco grandes (Big Five) o modelo de los cinco factores (FFM por sus siglas en inglés) de la personalidad. Su fortaleza recae en que de forma general se acepta que los rasgos de personalidad, aunque se observen algunos cambios, se mantienen relativamente estables a lo largo de la vida de una persona [6].

En los últimos años se han realizado diversos trabajos que presentan adaptaciones a diversos idiomas de los ítems proporcionados por el IPIP, con la finalidad de evaluar su aplicabilidad en diversas culturas, encontrándose resultados positivos. Como

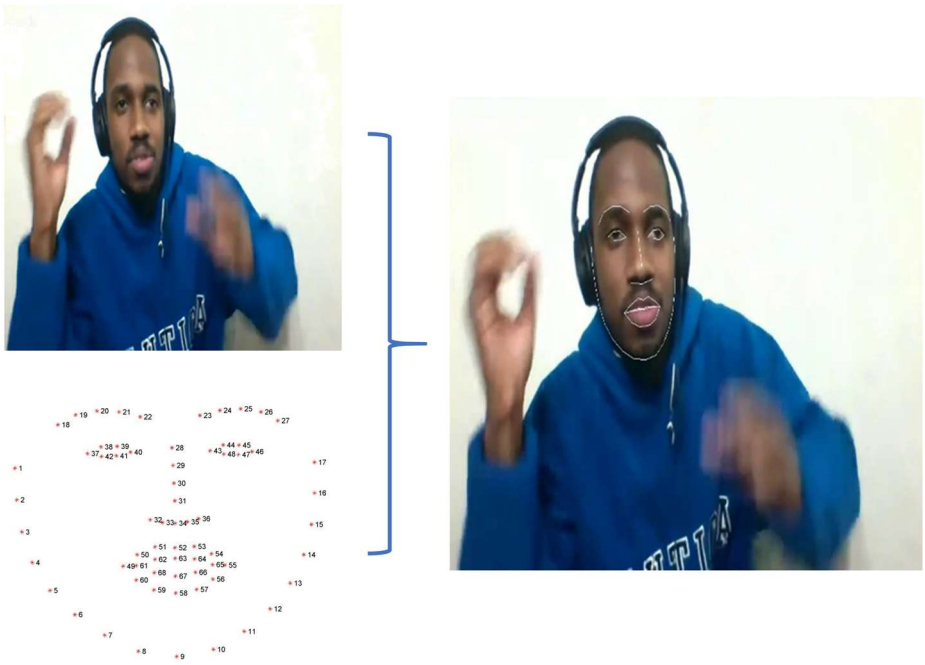


Fig. 2. Proceso de etiquetado del rostro.

ejemplo de lo anterior tenemos el estudio realizado por Gross y Cupani para la adaptación y contextualización de 100 ítems del IPIP en el medio argentino, obteniendo resultados satisfactorios en sus estudios de confiabilidad [7] y por otra parte la adaptación realizada por Laverdière et al. [6] para la aplicación de una versión reducida del cuestionario IPIP en participantes franceses.

Dicha versión constaba de 20 ítems en total, en donde cada uno de los rasgos de la personalidad se evaluó con 4 ítems, obteniendo como resultado la confirmación de la pertinencia transcultural de los indicadores de personalidad, del modelo de los cinco grandes en participantes con antecedentes idiomáticos y culturales diversos. En el ámbito del reconocimiento automático, en los últimos años se han propuesto diversos enfoques para el reconocimiento de la personalidad aparente.

Algunos estudios han trabajado en el reconocimiento automático con base a información textual, como por ejemplo la información generada por los usuarios en redes sociales como Facebook, Twitter y YouTube [8].

Otros estudios han trabajado en el reconocimiento de la personalidad aparente con base en la voz de los participantes [9]. Asimismo, se han realizado investigaciones para la detección de la personalidad aparente con base en imágenes extraídas de videos de los participantes, utilizando diversos modelos de redes neuronales [1, 10].

El renovado interés por el mundo de la inteligencia artificial y el aprendizaje automático, así como la existencia de competencias como las realizadas por ChaLearn Looking at People han ayudado al crecimiento de diversos modelos de redes

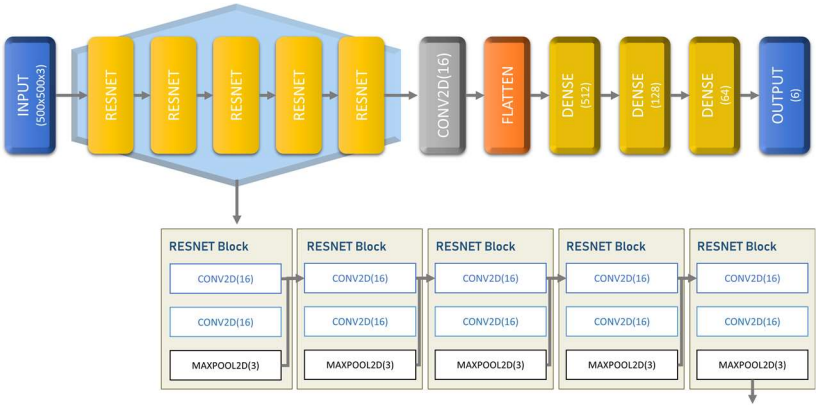


Fig. 3. Topología de red neuronal convolucional discreta.

neuronales para la detección de personalidad aparente con base en primeras impresiones [1, 11, 15].

3. Plataforma para la recolección de datos para el reconocimiento automático

Como estamos trabajando con información de personas, es importante resaltar que los participantes son notificados de que los datos recabados de la prueba estandarizada y los videos son utilizados de forma interna para los experimentos, por el equipo de investigadores, por lo que no se publica información que revele o comprometa su identidad sin consentimiento previo.

No es la intención de la investigación hacer público el conjunto de datos obtenido, y solo se haría exclusivamente con el consentimiento de los participantes. Para ello, la plataforma solicita siempre el registro de estos para contar con sus datos de contacto.

3.1. Arquitectura

La recolección de datos presenta un reto, ya que debemos establecer un medio para almacenar y consultar la información. En la actualidad, debido al avance de la tecnología, podemos desarrollar ambientes que hagan uso de Internet y de esta forma poder llegar a más personas sin importar su ubicación ni el dispositivo que utilicen para conectarse al internet.

Por lo anterior, optamos por desarrollar una plataforma en la nube, que funcionara en cualquier dispositivo y que permitiera almacenar la información en un repositorio localizado en Internet para facilitar el estudio de los datos. Optamos por utilizar un modelo arquitectónico de capas sobre una arquitectura cliente/servidor. Definimos 3 capas: presentación, lógica de la aplicación y datos. En la Figura 1 podemos apreciar la vista lógica de la plataforma.

Tabla 1. Comparativa entre la técnica utilizada y otros enfoques del estado del arte.

Nombre	Año	Técnica	Exactitud (Accuracy)
TNMCUL	2021	CNN Discreta	0.936158
NJU-LAMDA	2016	Deep Multi-Modal Regression	0.912968
evolgen	2016	Multi-modal LSTM Neural Network with Randomized Training	0.912063
DCC	2016	Multi-modal Deep ResNet 2D kernels	0.910933
Ucas	2016	AlexNET, VGG, ResNet con HOG3D, LBP-TOP	0.909824
BU-NKU	2016	Deep feature extraction with regularized regression and feature level fusion	0.909387
Pandora	2016	Multi-modal deep feature extraction single frame and late fusion	0.906275
Pilab	2016	Speech features 1000 forest random trees regression	0.893602
Kaizoku	2016	Multi-modal parallel CNN	0.882571

La capa de presentación muestra a los usuarios una interfaz gráfica que les ofrece la opción de registrarse en el sistema o iniciar sesión. La capa de presentación usa a la capa de lógica de la aplicación, para ejecutar las operaciones soportadas por el sistema.

La capa de lógica de la aplicación, a su vez, se conecta a la capa de datos que contiene la base de datos donde se guarda la información referente a la identificación de los usuarios, los resultados de la prueba del IPIP y de los reconocedores automáticos y también en esta capa se tiene el almacenamiento de datos cuya función es almacenar los archivos de video (incluido el audio) relacionados con los usuarios.

La aplicación desarrollada para la recolección de datos, llamada PersonApp, es un sistema multiplataforma hospedado en la nube de Internet, utilizando el hospedaje gratuito que ofrece Google Firebase como parte de sus servicios. Para el desarrollo de la interfaz de usuario decidimos utilizar React y para la parte de lógica de la aplicación y almacenamiento de información y de archivos, optamos por los servicios gratuitos de Google Firebase.

Lo anterior, nos da la flexibilidad de tener la información disponible desde cualquier ubicación que cuente con conexión a Internet. Otra ventaja de utilizar dichos servicios es el respaldo en privacidad y seguridad de los datos que nos ofrece Firebase, ya que está certificado en los principales estándares de seguridad y privacidad [17].

3.2. Reconocimiento automático de personalidad

En este proyecto hemos decidido evaluar inicialmente un modelo de reconocimiento automático basado en aprendizaje profundo, utilizando redes neuronales convolucionales (CNN, por sus siglas en inglés). Dicho reconocedor automático fue entrenado utilizando el conjunto de datos sobre personalidad de ChaLearn el cual

Fig. 4. Flujo de trabajo para la recolección de datos.

contiene 10,000 videos con una duración aproximada de 15 segundos y que fueron extraídos de más de 3,000 videos de alta definición de la plataforma Youtube [12].

El reconocedor utiliza imágenes extraídas de los videos. Aunque PersonApp permite almacenar solo video y voz, dejaremos para estudios posteriores la evaluación de reconocedores automáticos basados en audio. Para extraer de manera efectiva las características de los datos de video y cumplir con los requisitos de CNN, es mejor muestrear los videos espacial y temporalmente.

Primero, cada video se muestrea con 32 cuadros con ayuda de la biblioteca OpenCV [13] usando un tiempo de muestra de 2 segundos. Este número es uno de los más utilizados para equilibrar la extracción efectiva de características de las señales relevantes para el movimiento y los requisitos computacionales en este conjunto de datos de video.

Luego los redimensionamos a un tamaño de 200×200×3 y usamos la biblioteca dlib [14] para detectar rostros en cada cuadro. Una vez obtenidos los vectores representativos de los puntos de referencia faciales, los usamos para dibujar líneas para enfatizar los rasgos faciales en las imágenes. La Figura 2 describe el proceso anterior. La arquitectura utilizada por el reconocedor automático evaluado es una red neuronal residual convolucional discreta (Resnet).

Se usaron imágenes de tamaño de 200×200×3 píxeles con sus correspondientes etiquetas de video para la entrada. Posteriormente procedimos a usar 5 módulos de redes residuales convolucionales para procesar los datos de las imágenes y crear vectores de atributos. Finalmente usamos una red neuronal de 3 capas para la clasificación de características con una capa de salida de 6 neuronas para la regresión.

La pérdida se midió utilizando el error promedio medio (MAE, por sus siglas en inglés). La arquitectura completa es la siguiente (ver Figura 3): primero tenemos nuestra capa de entrada de tamaño 200×200×3. Luego creamos 5 módulos Resnet, donde cada módulo Resnet contiene una capa convolucional bidimensional (Conv2D) de 16 filtros en 3 dimensiones, con activación de unidad lineal rectificada (Relu) conectada a otra capa Conv2D, con 16 filtros en 3 dimensiones con activación Relu.

Se usó una capa de concatenación para agregar las características de esta convolución con las características del último bloque de Resnet obtenido. Finalmente, esa capa se conectó a una capa de agrupación máxima bidimensional (Maxpool2D) con

Tabla 3. Resultados de la prueba IPIP y reconocedor automático Participante 2.

Factor	Prueba IPIP	TNMCUL-Video 1	TNMCUL-Video 2	TNMCUL-Video 3	TNMCUL-Video 4
Amabilidad	0.5	0.31156	0.36113	0.31811	0.32420
Apertura	0.7	0.43183	0.50609	0.44732	0.44102
Neuroticismo	0.9	0.28898	0.35877	0.29758	0.29936
Responsabilidad	0.8	0.32328	0.39110	0.33087	0.33213
Sociabilidad	0.7	0.35664	0.41527	0.35632	0.35798

un paso (stride) de 3. Luego se usó una sola capa Conv2D con 16 filtros en 3 dimensiones y una agrupación promedio global.

Se aplanó el vector de características y procedimos a conectar 4 capas densamente conectadas, cada una con 512, 128, 64 y 6 neuronas respectivamente. Las primeras 3 utilizaron la activación de Relu y la última capa usada para la regresión utilizó la activación sigmoidea.

En la Tabla 1, mostramos los resultados de exactitud (accuracy) del modelo utilizado (denominado TNMCUL) y su comparativa contra enfoques del estado del arte, incluidos en las publicaciones de los mejores participantes en los concursos de reconocimiento de personalidad aparente con base a primeras impresiones de ChaLearn [15, 16].

3.3. Prueba estandarizada de personalidad

Para el desarrollo de este proyecto, en la parte de la prueba estandarizada utilizamos una representación IPIP de 50 ítems de los marcadores mencionados por Goldberg para la estructura factorial del modelo de los cinco grandes [3].

Cada uno de los cinco factores de la personalidad es evaluado por medio de 10 ítems, los cuales a su vez son calificados por el participante en una escala de Likert de 5 elementos (totalmente en desacuerdo; parcialmente en desacuerdo; ni de acuerdo, ni en descuerdo; parcialmente de acuerdo y totalmente de acuerdo) con base a su nivel de acuerdo o desacuerdo con respecto a cada declaración mostrada.

Cada opción tiene un valor de 1 a 5 puntos, por lo que 50 es el máximo puntaje por factor. Al final, convertimos el puntaje obtenido a un valor entre 0 y 1. Esta información es almacenada en el repositorio en la nube y se registra a que usuario pertenece. Dichos valores se utilizan para compararlos contra los resultados de los reconocedores automáticos.

4. Flujo de trabajo para la recolección de datos

En la Figura 4 podemos observar, de manera gráfica, el flujo de trabajo utilizado para la recolección de datos: como primer paso, el participante debe registrarse en la plataforma e iniciar sesión. Una vez dentro de la plataforma, el participante deberá completar un proceso de dos fases; en la primera fase el participante deberá responder la prueba estandarizada IPIP de 50 ítems. Una vez completada la prueba, el participante, será dirigido a la fase 2 de la intervención.

Tabla 4. Resultados de la prueba IPIP y reconocedor automático Participante 3.

Factor	Prueba IPIP	TNMCUL-Video 1	TNMCUL-Video 2	TNMCUL-Video 3	TNMCUL-Video 4
Amabilidad	0.8	0.40334	0.39985	0.39004	0.40899
Apertura	0.8	0.55564	0.55591	0.55077	0.56592
Neuroticismo	0.9	0.40694	0.39845	0.40062	0.41116
Responsabilidad	0.8	0.48791	0.47217	0.47934	0.49052
Sociabilidad	0.3	0.49638	0.47952	0.49235	0.49898

En dicha fase deberá grabar tres diferentes videos de una duración aproximada de un minuto cada uno. PersonApp divide cada grabación en 4 videos de aproximadamente 15 segundos cada uno y los almacena en nuestro repositorio en la nube.

Con lo anterior, el participante completa su colaboración; posteriormente, los videos recopilados son procesados y evaluados utilizando el reconocedor automático y la información generada es almacenada en el repositorio en la nube, vinculando los datos al usuario. Finalmente, podemos comparar los resultados de los reconocedores automáticos o personalidad aparente contra los valores obtenidos por la prueba estandarizada o personalidad real.

5. Experimento, pruebas y resultados

En esta sección presentaremos el experimento inicial, las pruebas y los resultados obtenidos. Estos resultados son preliminares ya que el objetivo era realizar una primera prueba de la plataforma de recolección de datos y su integración con un modelo a evaluar. A cada participante se le invitó a responder la prueba IPIP y posteriormente grabar, durante un minuto, un video en donde se le pide hablar libremente sobre un tema cualquiera.

Posteriormente, el video es procesado y separado en cuatro videos de 15 segundos cada uno. Cada uno de los cuatro videos es utilizado para extraer las imágenes que alimentan al reconocedor automático y los resultados son almacenados directamente en la base de datos. A continuación, en las Tablas 2, 3 y 4, presentamos una muestra de los resultados obtenidos de tres participantes.

En dichas tablas se aprecian los valores obtenidos en cada factor de personalidad del modelo de los cinco grandes tanto en la prueba estandarizada del IPIP como el valor estimado por el reconocedor automático en cada uno de los cuatro videos.

Analizando los resultados, observamos que el reconocedor automático, aunque arroja valores diferentes entre las evaluaciones de los cuatro videos de cada participante, estas diferencias son mínimas. En la Tabla 5, podemos observar los valores promedio y la desviación estándar para cada conjunto de valores de los cinco factores de personalidad para el caso del participante 3, comprobando con ello que las diferencias entre las mediciones de los 4 segmentos de video son irrelevantes.

Por otra parte, observamos que los valores obtenidos por el reconocedor automático TNMCUL están muy distantes de los valores reportados por la prueba del IPIP en todos

Tabla 5. Valores estadísticos del reconocedor automático Participante 3.

Factor	Promedio	Desviación Estándar	Video 1	Video 2	Video 3	Video 4
Amabilidad	0.40056	0.00796	0.40334	0.39985	0.39004	0.40899
Apertura	0.55706	0.00636	0.55564	0.55591	0.55077	0.56592
Neuroticismo	0.40429	0.00583	0.40694	0.39845	0.40062	0.41116
Responsabilidad	0.48249	0.00837	0.48791	0.47217	0.47934	0.49052
Sociabilidad	0.49181	0.00863	0.49638	0.47952	0.49235	0.49898

los casos presentados. Si bien, el reconocedor automático TNMCUL, como se menciona en la Tabla 1, presentaba una exactitud de 94% con el dataset de ChaLearn podemos observar que al utilizarlo en un dataset distinto no proporciona resultados aceptables.

Entre los factores que pueden ocasionar lo anterior, consideramos que el hecho de que el dataset de ChaLearn utilizado para el entrenamiento haya sido etiquetado con base a la opinión de un evaluador al observar los videos del dataset, produce una valoración subjetiva. Por otra parte, la prueba del IPIP proporciona valores con base a la opinión que el participante tiene de sí mismo.

6. Conclusiones y trabajos futuros

La plataforma desarrollada permite una recolección de datos muy sencilla y aplicable a través de cualquier dispositivo para navegar en Internet. La decisión de optar por un almacenamiento en la nube nos permite la posibilidad de recolectar información desde cualquier ubicación y apoya la disponibilidad inmediata de los datos recolectados para su análisis.

Hemos agregado como aportación secundaria, la evaluación de un modelo de reconocimiento automático con la finalidad de revisar la funcionalidad de la plataforma PersonApp. En este primer ejercicio, hemos encontrado que el reconocedor evaluado presenta una brecha en los resultados con respecto a la prueba IPIP. Posteriormente, se continuará recolectando datos y conduciendo más experimentos para ampliar la muestra y poder presentar información estadística de mayor relevancia.

Lo anterior, nos abre un nuevo reto para las futuras investigaciones, ya que con base a la información obtenida podemos trabajar en mejorar los modelos de reconocimiento automático utilizando el dataset que se está generando y los resultados de la prueba IPIP, buscando generar un modelo más acorde al ámbito analizado y utilizando los valores de la prueba del IPIP como etiquetas de los rasgos de personalidad.

Como trabajo futuro, se pretende analizar reconocedores automáticos con base en la voz, buscando utilizar modelos del estado del arte pero que han sido entrenados con videos en inglés y corroborar si los resultados son parecidos a los de la prueba estandarizada y en su defecto trabajar en el reentrenamiento de dichos modelos, aprovechando el dataset que se está formando con videos en español.

También se tiene como objetivo futuro, el poder utilizar los reconocedores automáticos mejorados en herramientas como sistemas tutores inteligentes para personalizar la enseñanza con base a la personalidad detectada del participante.

Referencias

1. Bradski, G.: The openCV library. Dr. Dobb's Journal of Software Tools, vol. 120, pp. 122–125 (2000)
2. ChaLearn (2021) chalearnlap.cvc.uab.es/
3. Escalante, H. J., Kaya, H., Salah, A. A., Escalera, S., Gucluturk, Y., Guclu, U., Baro, X., Guyon, I., Junior, J. C., Madadi, M., Ayache, S., Viegas, E., Gürpınar, F., Sukma-Wicaksana, A., Liem, C. C. S., van Gerven, M. A. J., van Lier, R.: Modeling, recognizing, and explaining apparent personality from videos. IEEE Transactions on Affective Computing, vol. 13, no. 2, pp. 894–911 (2022) doi: 10.1109/taffc.2020.2973984
4. Farnadi, G., Sitaraman, G., Sushmita, S., Celli, F., Kosinski, M., Stillwell, D., Davalos, S., Moens, M. F., De Cock, M.: Computational personality recognition in social media. User Modeling and User-Adapted Interaction, Springer Science and Business Media LLC, vol. 26, no. 2–3, pp. 109–142 (2016) doi: 10.1007/s11257-016-9171-0
5. Goldberg, L. R., Johnson, J. A., Eber, H. W., Hogan, R., Ashton, M. C., Cloninger, C. R., Gough, H. G.: The international personality item pool and the future of public-domain personality measures. Journal of Research in Personality, vol. 40, no. 1, pp. 84–96 (2006) doi: 10.1016/j.jrp.2005.08.007
6. Goldberg, L. R.: A broad-bandwidth, public domain, personality inventory measuring the lower-level facets of several five-factor models. Tilburg, The Netherlands: Tilburg University Press, Mervielde, I., Deary , I., De Fruyt, F., Ostendorf, F. (eds.) Personality Psychology in Europe, vol. 7, pp. 7–28 (1999)
7. Goldberg, L. R.: The development of markers for the Big-Five factor structure. Psychological Assessment. American Psychological Association (APA), vol. 4, no. 1, pp. 26–42 (1992) doi: 10.1037/1040-3590.4.1.26
8. Gross, M. N., Cupani, M.: Adaptación de los 100 ítems IPIP para medir los cinco grandes factores. Revista Mexicana de Psicología, vol. 33, no. 1, pp. 17–29 (2016)
9. International Personality Item Pool: A scientific collaboratory for the development of advanced measures of personality traits and other individual differences (2021) ipip.ori.org/.
10. Junior, J., Güçlütürk, Y., Pérez, M., Güçlü, U., Andujar, C., Baró, X., Escalante, H. J., Guyon, I., van Gerven, M. A., van Lier, R., Escalera, S.: First impressions: A survey on vision-based apparent personality trait analysis. IEEE Transactions on Affective Computing, Institute of Electrical and Electronics Engineers (IEEE), vol. 13, no. 1, pp. 75–95 (2022) doi: 10.1109/taffc.2019.2930058
11. King, D. E.: DLIB-ML: A machine learning toolkit. The Journal of Machine Learning Research, vol. 10, pp 1755–1758 (2009)
12. Laverdière, O., Gamache, D., Morin, A. J., Diguer, L.: French adaptation of the Mini-IPIP: A short measure of the Big Five. European Review of Applied Psychology, vol. 70, no. 3 (2020) doi: 10.1016/j.erap.2019.100512
13. Ponce-López, V., Chen, B., Oliu, M., Corneanu, C., Clapés, A., Guyon, I., Baró, X., Escalante, H. J., Escalera, S.: ChaLearn LAP 2016: First round challenge on first impressions - Dataset and results. Lecture Notes in Computer Science, Springer International Publishing, pp. 400–418 (2016) doi: 10.1007/978-3-319-49409-8_32
14. Privacidad y seguridad en Firebase. (2021) firebase.google.com/support/privacy
15. Subramaniam, A., Patel, V., Mishra, A., Balasubramanian, P., Mittal, A.: Bi-modal first impressions recognition using temporally ordered deep audio and stochastic visual features. Lecture Notes in Computer Science,, Springer International Publishing, pp. 337–348 (2016) doi:10.1007/978-3-319-49409-8_27
16. Ventura, C., Masip, D., Lapedriza, A.: Interpreting CNN models for apparent personality trait regression. IEEE Conference on Computer Vision and Pattern Recognition Workshops (2017) doi: 10.1109/cvprw.2017.217

17. Yu, J., Markov, K., Karpov, A.: Speaking style based apparent personality recognition. Speech and Computer, Springer International Publishing, pp. 540–548 (2019) doi: 10.1007/978-3-030-26061-3_55

ISSN 1870-4069

Reconocimiento de las señas estáticas del LSM con características basadas en aprendizaje profundo

Rafael Fernández Rodríguez, Francisco Javier Peralta Rosas, Luis Ángel Zuñiga-Madrid, Pedro Arguijo

Tecnológico Nacional de México,
Campus Misantla,
México

{rafael.fernandez.rgz, francisco.peraltarosas, angel.zunigamadrid}@gmail.com, pedro_arguijo@excite.com

Resumen. El lenguaje de señas es el método de comunicación entre personas que sufren de problemas del habla y de la audición. Los métodos de extracción de características juegan un papel importante en la obtención de una alta tasa de reconocimiento. En este trabajo se evalúan las características profundas extraídas por la arquitectura VGG16 tanto en la capa fc6 y fc7, así como la concatenación de ambas para identificar las señas estáticas del lenguaje de señas mexicano. Para el reconocimiento de las señas se utilizaron los clasificadores de RF y SVM. Los resultados generales, en la clasificación de las veintiuna señas estáticas del lenguaje de señas mexicano, fueron del 92.5% y 95.32% para RF y SVM, respectivamente, con las características extraídas en la capa fc6.

Palabras clave: Reconocimiento de señas, Transferencia de aprendizaje, bosques aleatorios, máquinas de soporte vectorial.

LSM Static Sign Recognition with Deep Learning-Based Features

Abstract. Sign language is the method of communication between people who suffer from speech and hearing problems. Feature extraction methods play an important role in obtaining a high recognition rate. In this paper we evaluate the deep features extracted by the VGG16 architecture in both fc6 and fc7 layers, as well as the concatenation of both to identify the static signs of Mexican sign language. RF and SVM classifiers were used for sign recognition. The overall results, in the classification of the twenty-one static Mexican sign language signs, were 92.5% and 95.32% for RF and SVM, respectively, with the features extracted in layer fc6.

Keywords: Sign recognition, transfer learning, random forest, support vector machine.

1. Introducción

El reconocimiento de gestos es por demás significante en la interacción humano-computadora tales como la realidad virtual, juegos interactivos, procesamiento de imágenes o el reconocimiento del lenguaje de señas. El lenguaje de señas es una forma de comunicación, no verbal, utilizada por personas con problemas de audición y habla para expresar sus pensamientos y emociones. Los principales componentes del lenguaje de señas son los signos manuales y no manuales. Los signos manuales son: posición, orientación, forma y trayectoria de la mano. Mientras que los no manuales representan el movimiento del cuerpo y las expresiones faciales.

Aunque la información más importante se transmite a través de las manos, los signos no manuales permiten aclarar y enfatizar el significado de los signos manuales. Existen dos enfoques que se utilizan comúnmente para interpretar los gestos o señas en la interacción humano-computadora. El enfoque basado en guantes de datos [1, 2] se basa en dispositivos electromecánicos conectados a un guante para digitalizar los movimientos de las manos y los dedos en datos multi-paramétricos.

El principal problema de esta implementación es que requiere llevar dispositivos, los cuales son costosos y provoca comportamientos menos naturales. El segundo basado en visión artificial se divide a su vez en: enfoque basado en el modelo 3D de la mano y el basado en la apariencia. El enfoque del modelo 3D de la mano se basa en el modelo cinemático 3D de la mano e intenta estimar los parámetros de ésta mediante la comparación entre las imágenes de entrada y la posible apariencia 2D proyectada por el modelo de la mano 3D.

El enfoque basado en la apariencia utiliza características extraídas de la imagen RGB para modelar la apariencia visual de la mano y comparar estos parámetros. Como se mencionó, la mayoría de los modelos basados en apariencia se basan en extraer características que representan el contenido de las imágenes. Para hacer frente a la dependencia del punto de vista, estos métodos deben tener propiedades de invariancia a los cambios de traslación, rotación y escala.

Estas características pueden estar basadas en regiones como los momentos de Hu, momentos de Zernike o de Jacobi-Fourier [3–9], o basadas en contornos como los descriptores de Fourier [10, 11] o Histograma de Gradientes Orientados [7]. También se han utilizado características tipo Haar 3D extraídas de imágenes de profundidad [12]. Dichas características se han clasificado con redes neuronales artificiales o máquinas de soporte vectorial.

Sin embargo, se debe mencionar que sin importar el método de extracción de características se debe resolver el problema de segmentación de la mano que es un proceso complejo. Recientemente, el aprendizaje profundo mediante redes neuronales convolucionales (CNN) ha obtenido mucho éxito en tareas de reconocimiento visual como la clasificación de imágenes y la detección de objetos [13-15].

Dado que las características adquiridas a partir de estas redes neuronales son bastante potentes, es popular tratar estas CNN, entrenadas en grandes conjuntos de datos de imágenes naturales, como extractores genéricos de características; a este proceso se le conoce como transferencia de aprendizaje. Al reutilizar el conocimiento adquirido en tareas relacionadas, se vuelve más fácil abordar tareas más exigentes, como la recuperación de imágenes, la segmentación semántica y el reconocimiento de emociones, por mencionar algunas.

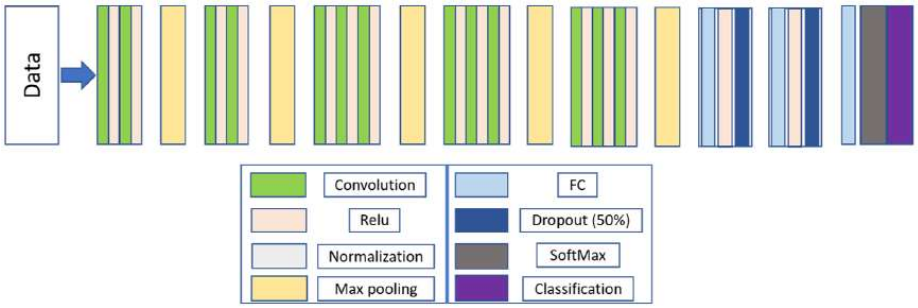


Fig. 1. Arquitectura de VGG16.

De acuerdo con lo mencionado por el Consejo Nacional para el Desarrollo y la Inclusión de las Personas con Discapacidad, la Lengua de Señas Mexicana (LSM), es la lengua que utilizan las personas sordas en México. Como toda lengua, posee su propia sintaxis, gramática y léxico [16]. Se compone de signos visuales con estructura lingüística propia y consta de 27 señas, de las cuales 21 son estáticas y las restantes dinámicas.

Este trabajo se centra en la clasificación de las señas estáticas del LSM. Proponemos extraer características profundas en dos capas completamente conectadas, fc6 y fc7, de la arquitectura VGG16 y realizar la clasificación de estas con Bosques Aleatorios (Random Forest, RF) y Máquinas de Soporte Vectorial (Support Vector Machine, SVM) tanto de manera individual como concatenando ambos conjuntos de características.

El estudio está organizado de la siguiente manera. Si bien la transferencia de aprendizaje y la arquitectura VGG16 se aborda en el Sección 2, la descripción de los clasificadores RF y SVM se dan en la sección 3. El conjunto de datos utilizado se describe en la Sección 4. En la Sección 5 se discuten los resultados. Finalmente, en la Sección 6, se presentan algunas observaciones finales.

2. Transferencia de aprendizaje

La transferencia de aprendizaje (transfer learning) es un ingrediente esencial para la inteligencia artificial, donde el conocimiento aprendido en un dominio para alguna tarea puede transferirse a otro dominio para una tarea diferente [17]. En el contexto del aprendizaje profundo, la transferencia de aprendizaje comúnmente se implementa en una red neuronal convolucional profunda, CNN, entrenada previamente en un gran conjunto de datos etiquetados.

Este enfoque se ha aplicado con éxito en muchas áreas de la visión artificial. Las capas iniciales de una CNN extraen características genéricas simples (bordes, esquinas, curvas, manchas de color, etc.), que son aplicables a todo tipo de imágenes, mientras que las capas finales representan características muy abstractas y específicas de los datos.

Por lo tanto, se espera que el uso de un modelo entrenado de forma óptima en un gran conjunto de datos y su posterior ajuste en un conjunto de datos diferente dé como

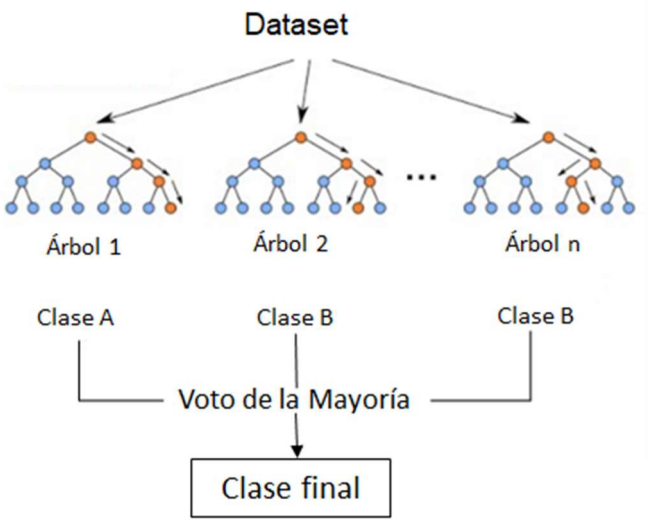


Fig. 2. Estructura de Random Forest.

resultado una mayor precisión y un proceso de entrenamiento más rápido, en comparación con el entrenamiento de las mismas CNN desde cero, debido a las características compartidas presentes en las capas iniciales. Existen dos enfoques que se pueden utilizar en la transferencia de aprendizaje:

– Extracción de características: Consiste en utilizar las características de una red previamente entrenada para representar las imágenes de nuevos conjuntos de datos. Estas características se utilizan para entrenar un nuevo clasificador. Al utilizar la CNN como extractor de características, se elimina la última capa totalmente conectada, que es la capa de salida.

Los valores de las características se pueden extraer como valores sin procesar o después de haber sido transformados por la función ReLU, donde una salida x se asigna a $\max(0, x)$. Luego, se utilizan estas características profundas para entrenamiento y clasificación.

– Ajuste fino: El ajuste fino de la red pre-entrenada es relevante cuando el conjunto de datos objetivo es muy grande. Consiste en descongelar algunas de las capas superiores de un modelo congelado que se utiliza para la extracción de características. Además, los parámetros de todas las capas (excepto las últimas) de la red pre-entrenada se inicializan, el entrenamiento se realizará más rápidamente que si la inicialización hubiera sido aleatoria.

VGG16 [18] es una arquitectura simple y ampliamente utilizada para ImageNet. Toma como entrada una imagen de 224 × 224 pixeles y devuelve un vector de tamaño 1000 con las probabilidades de pertenecer a cada clase. VGG16 contiene 13 capas de convolución, 3 capas completamente conectadas (fc6, fc7 y fc8) y 5 capas de agrupación (como se muestra en la Fig. 1).

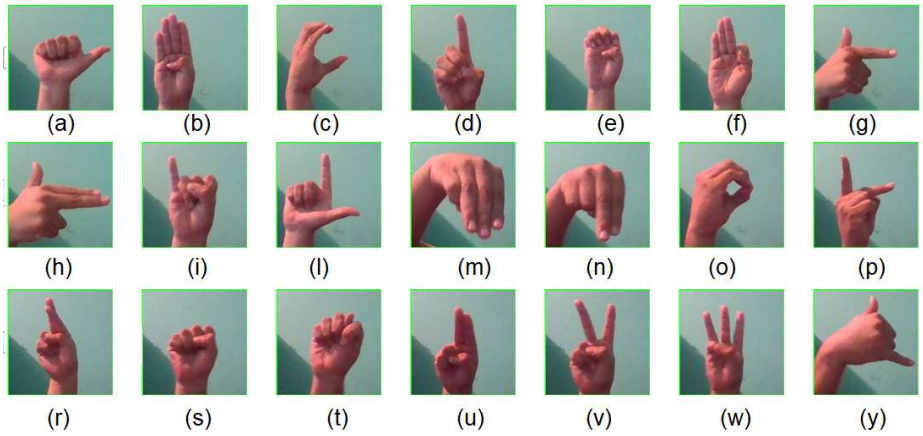


Fig. 3. Muestra representativa de las señas estáticas del LSM utilizado. Debajo de cada seña se indica a que letra corresponde.

Las 16 capas convolucionales se utilizan para extraer características de la imagen. En cada capa de convolución hay un filtro múltiple de 3x3, con 1 píxel como paso. La última capa softmax se utiliza para la clasificación. En cada capa de convolución, ReLU se aplica como función de activación. En este trabajo utilizamos la arquitectura VGG16 para extraer vectores de características profundas en las capas de activación fc6 y fc7, en ambos casos son características pre ReLU, las cuales individualmente generan un vector de 4096 características.

Seleccionamos VGG16 porque se ha aplicado ampliamente en diversas tareas de clasificación. Se trata de una de las primeras arquitecturas que exploran la profundidad de la red al ampliarla a 16 capas y utilizar filtros de convolución muy pequeños (3×3). Además, al haber sido entrenada en el conjunto de datos de Imagenet es un extractor de características capaz de generalizar muy bien las imágenes y puede aplicarse a una amplia variedad de tareas como la clasificación de objetos, la detección, la localización, etc.

3. Clasificadores

La clasificación es una tarea que requiere el uso de algoritmos de aprendizaje automático para asignar una clase a ejemplos no vistos previamente del problema. Identificar la categoría o clase a la que pertenece un nuevo dato es el objetivo de un problema de clasificación. A continuación, describimos brevemente los clasificadores utilizados en este trabajo.

3.1. Bosques aleatorios (Random Forest)

Los algoritmos de bosque aleatorio (RF) forman una familia de métodos de clasificación que se basan en la combinación de varios árboles de decisión no podados (ver Fig. 2). La particularidad de tales ensambles de clasificadores es que sus componentes basados en árboles se obtienen a partir de un cierto grado de aleatoriedad.

Tabla 1. Exactitud obtenida con RF en el conjunto de entrenamiento. Las razones de entrenamiento y prueba son las indicadas.

	fc6	fc7	fc6 fc7
10/90	0.8772	0.8026	0.87
20/80	0.925	0.9272	0.7835
30/70	0.9249	0.919	0.9259

En base a esta idea, RF se define como un principio genérico de conjuntos aleatorios de árboles de decisión [19]. La unidad básica de RF (la denominada aprendiz base) es un árbol binario construido utilizando particiones recursivas (RPART). RF Ofrece un rendimiento excelente en una serie de problemas prácticos, ya que no es sensible al ruido en el conjunto de datos y no está sujeto a un sobreajuste.

Los árboles de clasificación en el Bosque Aleatorio se construyen recursivamente utilizando el criterio de impureza Gini que se utiliza para determinar las divisiones en la variable predictiva. La división de un nodo del árbol se realiza sobre la variable de manera que reduce la incertidumbre presente en los datos y por lo tanto la probabilidad de una clasificación errónea. La división ideal de un nodo del árbol ocurre cuando es cero el valor de Gini.

El proceso de división continúa hasta que se crea un "bosque", formado por múltiples árboles. La clasificación se produce cuando cada árbol del bosque emite un voto para la clase más popular. El bosque aleatorio entonces elige la clasificación que tiene más votos sobre todos los árboles del bosque. La poda no es necesaria ya que cada clasificación se produce por un bosque final que consiste en árboles generados independientemente creados a través de un subconjunto aleatorio de datos, evitando el sobreajuste.

Las tasas de error de generalización dependen de la fuerza de los árboles individuales en el bosque y la correlación entre ellos. Esta tasa de error converge a un límite a medida que aumenta el número de árboles en el bosque. Otra ventaja de RF es que no hay validación cruzada o un conjunto de pruebas separado para obtener una estimación imparcial del error del conjunto de pruebas.

La precisión del conjunto de prueba se estima internamente en RF al ejecutar muestras fuera de bolsa (OOB). Por cada árbol que crece en RF, aproximadamente un tercio de los casos están fuera de bolsa (fuera de la muestra de bootstrap). Las muestras fuera de bolsa (OOB) pueden servir como un conjunto de prueba para el árbol cultivado en los datos que no son OOB.

3.2. Máquinas de soporte vectorial (Support Vector Machine)

Las máquinas de soporte vectorial (SVM) son una técnica de reconocimiento supervisado, desarrollada originalmente para clasificar clases de objetos linealmente separables. En la SVM, el hiperplano óptimo o frontera de decisión se determina en función de un pequeño subconjunto de todos los ejemplos de entrenamiento que maximizan la separación entre conjuntos; o sea, la capacidad de clasificación entre clases, denominados vectores de soporte.

Tabla 2. Exactitud obtenida con SVM en el conjunto de entrenamiento. Las razones de entrenamiento y prueba son las indicadas.

	fc6	**fc7**	**fc6 fc7**
10/90	0.9192	0.916	0.9333
20/80	0.9532	0.9482	0.9556
30/70	0.9535	0.9608	0.951

Si los datos de entrenamiento no se pueden separar linealmente, es decir, si no es posible construir un hiperplano en el espacio original, la separación se realiza en un espacio dimensional superior.

Para fines prácticos, es habitual utilizar una función de mapeo no lineal llamada kernel, seleccionada entre funciones polinómicas, funciones de base radial, funciones de base radial gaussiana y funciones sigmoidales. SVM se diferencia de otras técnicas como las redes neuronales artificiales, ya que el problema de mínimos locales no afecta a SVM, porque su entrenamiento se basa en problemas de optimización convexa.

4. Conjunto de datos

El conjunto de datos utilizado en este trabajo es de acceso libre [7] y contiene imágenes de las veintiuna señas estáticas del LSM. Consta de 300 imágenes por seña, además cada imagen representa la señal de la letra con una distinta variación de acuerdo con rotación, traslación y escalamiento. En la Fig. 3 se muestra una imagen representativa de cada una de las señas. En la captura de las imágenes se utilizó un fondo verde para facilitar su segmentación.

5. Resultados

El método propuesto consiste en dos etapas. Primero, todas las imágenes del LSM se redimensionaron a 224×224 píxeles de acuerdo con el requisito de entrada de la del modelo pre-entrenado VGG16. Las características profundas de extrajeron de las capas fc6 y fc7 y por capa se obtuvo un vector de 4096 dimensiones para cada imagen.

También se realizó la concatenación de ambos conjuntos de características para formar una única representación de características profundas (8192 dimensiones por imagen). Como segunda etapa se realizó el entrenamiento de los clasificadores RF y SVM para determinar la etiqueta correspondiente a las señas de los conjuntos de características extraídos en fc6, en fc7 y la concatenación de ambas.

La metodología indicada se realizó con un procesador i7-CPU, 8GB de RAM y sistema operativo de 64 bits. La implementación de un proceso de aprendizaje puede generar procedimientos que requieren mucho tiempo y recursos, dependiendo de la calidad de los datos disponibles. Debido a la cantidad total de imágenes en el dataset, para los experimentos se seleccionaron aleatoriamente tres distintas proporciones para la parte de entrenamiento y pruebas de ambos clasificadores, dichas proporciones fueron: 10/90, 20/80 y 30/70.

Consideramos estas proporciones para evaluar su influencia en la exactitud teniendo en cuenta las características estudiadas y, además, con la finalidad de comparar

resultados previamente reportados con este dataset [7]. Es importante señalar que un muestreo aleatorio de la partición de entrenamiento y prueba puede producir resultados diferentes en diferentes ejecuciones.

Las Tablas 1 y 2 muestran la exactitud obtenida con el conjunto de características de pruebas **seleccionado** para los clasificadores RF y SVM, respectivamente. En ambas tablas se indica la razón de entrenamiento/prueba, así como la capa de extracción considerada: fc6, fc7 y la concatenación de ambas fc6 fc7.

Como se puede observar en ambas tablas la exactitud en la clasificación con las características concatenadas son mayores que los obtenidos con las características de la capa fc7; con la única excepción que se muestra en la Tabla 1 para la razón 20/80.

Sin embargo, la exactitud de las características de la capa fc6 y las características concatenadas son similares. Lo cual implica que se obtiene una buena clasificación únicamente con las características de la capa fc6. Aunque los resultados presentados en las Tablas 1 y 2 muestran que SVM tiene el mejor desempeño se debe mencionar que requiere un mayor tiempo de entrenamiento que RF.

De acuerdo con los resultados publicados previamente que consideran el mismo conjunto de imágenes para el reconocimiento del LSM, nuestros resultados están dentro del rango a pesar de la gran dimensionalidad de las características consideradas.

6. Conclusiones y trabajo a futuro

Se propuso un sistema de reconocimiento del LSM basado en la extracción de características profundas. Las características extraídas con la arquitectura VGG16 se clasificaron con RF y SVM. Se utilizaron las capas fc6 y fc7 para la extracción de características, obteniendo un vector con 4096 datos por imagen.

En los tres experimentos considerados para ambos clasificadores se observó que la mejor clasificación se obtiene con las características de la capa fc6. Como se esperaba, las razones de entrenamiento/pruebas juegan un rol importante para derivar un sistema robusto de clasificación.

Los mejores resultados de clasificación se obtuvieron con SVM. Como trabajo futuro se propone realizar una comparación del desempeño de las características extraídas por diversas arquitecturas profundas previamente entrenadas. De igual manera, debido a la cantidad de imágenes con las que cuenta el dataset, implementar la transferencia de aprendizaje por ajuste fino.

Referencias

1. Saldaña-González, G., Cerezo-Sánchez, J., Bustillo-Díaz, M. M., Ata-Pérez, A.: Recognition and classification of sign language for spanish. Computación y Sistemas, vol. 22, no. 1 (2018) doi: 10.13053/cys-22-1-2780
2. Maitre, J., Rendu, C., Bouchard, K., Bouchard, B., Gaboury, S.: Basic daily activity recognition with a data glove. Procedia Computer Science, vol. 151, pp. 108–115 (2019) doi: 10.1016/j.procs.2019.04.018
3. Otiniano-Rodríguez, K., Camara-Chavez, G., Menotti, D.: Hu and Zernike moments for sign language recognition. In: Proceedings of the International Conference on Image Processing, Computer Vision, and Pattern Recognition, pp. 918–922 (2012)

4. Martinez-Perales, J. C., Flores-Carapia, R., Luna-Benoso, B.: An automated method for identification of vowels on the sign language. Contemporary Engineering Sciences, vol. 8, pp. 1499–1508 (2015) doi: 10.12988/ces.2015.59278
5. Solís, F., Hernández, M., Pérez, A., Toxqui, C.: Static digits recognition using rotational signatures and Hu moments with a multilayer perceptron. Engineering, Scientific Research Publishing, Inc, vol. 6, no. 11, pp. 692–698 (2014) doi: 10.4236/eng.2014.611068
6. Perez, L. M., Rosales, A. J., Gallegos, F. J., Barba, A. V.: LSM static signs recognition using image processing. In: Proceedings of the 14th International Conference on Electrical Engineering, Computing Science and Automatic Control, IEEE (2017) doi: 10.1109/iceee.2017.8108885
7. Mancilla-Morales, E., Vázquez-Aparicio, O., Arguijo, P., Meléndez-Armenta, R. Á., Vázquez-López, A. H.: Traducción del lenguaje de señas usando visión por computadora. Research in Computing Science, vol. 148, no. 8, pp. 79–89 (2019)
8. Joshi, G., Vig, R., Singh, S.: Analysis of Zernike moment-based features for sign language recognition. Advances in Intelligent Systems and Computing, Springer Singapore, pp. 1335–1343 (2018) doi: 10.1007/978-981-10-5903-2_140
9. Solís, F., Toxqui, C., Martínez, D.: Mexican sign language recognition using Jacobi-Fourier moments. Engineering, Scientific Research Publishing, Inc, vol. 7, no. 10, pp. 700–705 (2015). doi: 10.4236/eng.2015.710061
10. Nur Fauzan, M. H., Rakun, E., Hardianto, D.: Feature extraction from smartphone images by using elliptical Fourier descriptor, centroid and area for recognizing indonesian sign language SIBI (Sistem Isyarat Bahasa Indonesia). In: Proceedings of the 2nd International Conference on Intelligent Autonomous Systems (ICoIAS), IEEE (2019) doi: 10.1109/icoias.2019.00008
11. Kishore, P. V. V., Prasad, M. V. D., Prasad, C. R., Rahul, R.: 4-Camera model for sign language recognition using elliptical Fourier descriptors and ANN. In: Proceedings of the International Conference on Signal Processing and Communication Engineering Systems, IEEE (2015) doi: 10.1109/spaces.2015.7058288
12. Jimenez, J., Martin, A., Uc, V., Espinosa, A.: Mexican sign language alphanumerical gestures recognition using 3D Haar-like features. IEEE Latin America Transactions, vol. 15, no. 10, pp. 2000–2005 (2017) doi: 10.1109/tla.2017.8071247
13. Krizhevsky, A., Sutskever, I., Hinton, G. E.: Imagenet classification with deep convolutional neural networks. In: Proceedings of Advances in Neural Information Processing Systems 25, pp. 1097–1105 (2012)
14. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014) doi: 10.48550/ARXIV.1409.1556
15. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A.: Going deeper with convolutions. pp. 1–9 (2014) doi: 10.48550/ARXIV.1409.4842
16. Gobierno de México. Lengua de Señas Mexicana (LSM) (2022) www.gob.mx/conadis/articulos/lengua-de-senas-mexicana-lsm?idiom=es
17. Yosinski, J., Clune, J., Bengio, Y., Lipson, H.: How transferable are features in deep neural networks? In: Proceedings of the Advances in Neural Information Processing Systems 27, pp. 3320–3328 doi: 10.48550/ARXIV.1411.1792
18. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. Computer Science, Computer Vision and Pattern Recognition (2015) doi: 10.48550/arXiv.1409.1556
19. Breiman, L.: Random forests. Machine Learning, Springer Science and Business Media LLC, vol. 45, no. 1, pp. 5–32 (2001) doi: 10.1023/a:1010933404324

ISSN 1870-4069

Análisis de calidad en imágenes esteganográficas aplicando el algoritmo LSB en códigos QR embebidos

Rodrigo Hernández Moncayo, José Martin Flores Albino,
Víctor Manuel Landassuri Moreno, Saturnino Job Morales Escobar,
Ivone Rodríguez Pérez

Universidad Autónoma del Estado del México,
Maestría en Ciencias de la Computación,
México

rhernandezm403@alumno.uaemex.mx,
{jmfloresa, vmlandassurim, sjmoralese,
irodriguezp} @uaemex.mx

Resumen. Se presenta un estudio del efecto en la imagen esteganográfica al embeber un Código de Respuesta Rápida (Quick Response Code) en otro código QR por medio de la técnica esteganográfica de sustitución del Bit Menos Significativo (LSB). Se plantea su aplicación como marca de agua donde en la primera parte se calculan los parámetros para inserción de un código QR en otro código QR, basados en la versión del código contenedor y del código QR oculto. En la segunda parte, se presenta el efecto en la calidad de imagen del código QR usado como cubierta dependiente del bit usado para ocultar el segundo código QR. Se presenta también el cálculo de las métricas de calidad de imagen, similitud y fidelidad visual para observar su comportamiento. Los resultados permiten observa que el código QR embebido puede recuperarse usando desde el LSB hasta el 5to bit.

Palabras clave: Esteganografía, algoritmo LSB, código QR embebido.

Quality Analysis in Steganography Images Using the LSB Algorithm in Embedded QR Codes

Abstract. A study of the effect on the steganographic image when embedding a Quick Response Code (QR Code) is presented using the steganographic technique of replacing the Less Significant Bit (LSB). The application as watermark is proposed where the first part calculates the parameters for inserting a QR code into another QR code based on the container version code and the hidden QR code. The second part presents the effect on the image quality of the QR code used as a bit-dependent cover used to hide the second QR code. Also is shown the metrics of image quality similarity, and visual fidelity to observe its behavior. The result indicates that the embedded QR code can be retrieved using from the LSB to the 5th bit.

Keywords: Steganography, LSB algorithm, embedding QR code.

1. Introducción

La esteganografía es el arte y la técnica de ocultar información en un medio que la hace imperceptible al observador. Esta técnica es eficaz para ocultar información que se transmite por un canal de mensajes con alta densidad de datos digitales; por ejemplo, textos, imágenes y voz [1].

Los elementos del proceso esteganográfico relacionan a un objeto portador o cubierta, que es algún medio digital y el mensaje oculto o secreto que se embeberá en el portador, creando así el llamado estego objeto, que es la mezcla del portador y del mensaje oculto y la llave o clave esteganográfica que orienta para recuperar el mensaje oculto [2].

La esteganografía es una herramienta para reducir la vulnerabilidad latente en medios digitales. Da seguridad entre emisor y receptor al incorporar información secreta y hacerla imperceptible para otros observadores. Se ha aplicado en la autenticación de documentos digitales, como: pasaportes, identificaciones de nacionalidad y licencias para conducir. En este tipo de aplicaciones, el mensaje secreto funciona como una marca de agua embebido en los documentos sensibles, siendo un recurso para validar su autenticidad [3].

Actualmente, los Quick Response Codes (códigos QR), son ampliamente usados para la identificación rápida de productos y para facilitar el acceso a información en el espacio digital. Son imágenes digitales que, al ser explorada, por la cámara de un teléfono inteligente, extraen la información codificada. Los códigos QR se utilizaron por primera vez en 1994 por Denso Wave, empresa del grupo Toyota. Desde 2011, y debido al uso general de la tecnología móvil, los códigos QR son extensamente usados como un medio de rápido acceso a la información [4].

Los códigos QR son imágenes que pueden duplicarse al copiarse a través de la captura de imagen en pantalla de un dispositivo móvil o al tomarles una fotografía, y la imagen reproducida podrá ser explorada para decodificar la información que contiene. Lo anterior hace que los códigos QR sean un medio para difundir rápida y fácilmente información. Por otro lado, hay casos donde se necesita autenticar el origen del código QR, siendo ahora la facilidad de reproducirlo y el acceso a la información vayan en contra de validar la originalidad del código QR.

Se propone así en este trabajo analizar la posibilidad de aprovechar que los códigos QR son imágenes digitales y con técnicas esteganográficas agregar una imagen digital de otro código QR como marca de agua, y así permita su validación, siendo un medio para comprobar el origen del código QR.

Este artículo presenta una serie de experimentos sobre el efecto del algoritmo esteganográficos LSB (The Least Significant Bit), para incrustar o embeber en un código QR un segundo código QR. Se analiza también el efecto en la calidad de la imagen esteganográfica para cuantificar su alteración. Existen diversos factores que impacta en la calidad de la estego imagen, entre estos la carga de datos del mensaje oculto y el algoritmo esteganográfico usado.

Para cuantificar el efecto del algoritmo esteganográfico se usa el Error Cuadratico Promedio (Mean Square Error, MSE) y la Tasa de Ruido a Señal, (Peak Signal to Noise Ratio, PSNR).

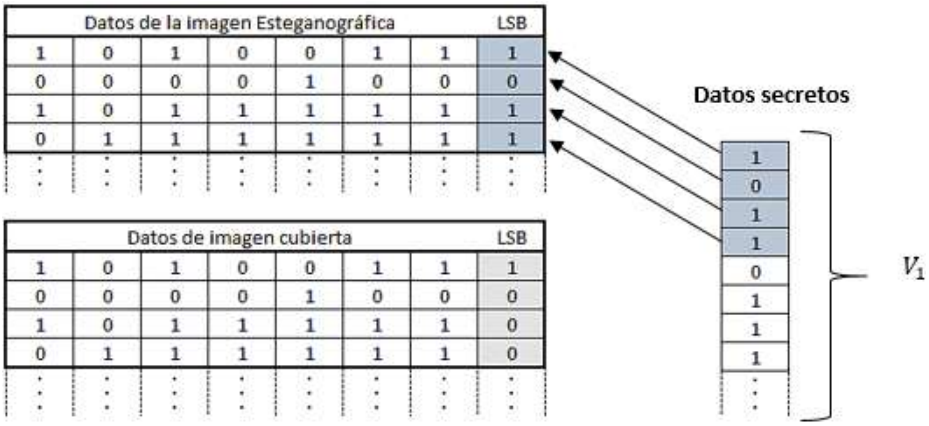


Fig. 1. Algoritmo esteganográficos Least Significant Bit.

A continuación, se presentan el método esteganográfico del bit menos significativo que se aplicará en imágenes con código QR, se dará la descripción de los códigos QR basados en la versión y sus parámetros de capacidad de información que puede contener.

1.1. Algoritmo esteganográfico the Least Significant Bit (LSB) y métricas de calidad de imagen

El algoritmo de sustitución del bit Menos Significativo (LSB) es la herramienta clásica para ocultar información en imágenes. Es una técnica del dominio espacial y se implementa al variar ligeramente el valor de los pixeles de la imagen cubierta para incorporar la información secreta [5].

En las imágenes en escala de grises cada píxel de la imagen es un elemento de una matriz, tomando valores en el intervalo de 0 a 255, siendo necesarios 8 bits por píxel. En las imágenes a color se usan 3 cadenas de N-bits/color (Red, Green, Blue), donde N es el número de bits para el color o profundidad de color (N= 8, 16 o 32 bits) [6].

Se muestra en la Fig. 1, el algoritmo LSB, donde los bits que representan a los pixeles de la imagen secreta se apilan ordenadamente en un vector V_1. Se realiza una comparación entre los bits menos significativos (LSB) de los píxeles de imagen cubierta y los bits del vector V_1. Si coinciden los bits, no se modifica el bit LSB de la imagen cubierta; en caso contrario, se sustituye el bit LSB de la imagen cubierta por el valor del bit de la imagen secreta. Al utilizar este método la información de la imagen secreta queda embebida en la información de la imagen cubierta, generando la estego imagen que es la imagen cubierta con la imagen secreta incorporada.

Para imágenes en tonos de la imagen secreta afecta al bit LSB de los pixeles de la imagen cubierta que sean necesarios para la cantidad de bits de la imagen secreta. En imágenes a color (RGB), los bits de la imagen secreta se reparten en los bits LSB de cada canal de color de la imagen cubierta [7].

El algoritmo LSB tiene la característica de ser un algoritmo reversible lo que permite recuperar el mensaje oculto de manera exacta, donde la llave será la información del algoritmo LSB y el tamaño de la imagen secreta. Para medir el efecto en la imagen estego se han propuesto diversos índices como referencia al cambio en la imagen estego.

Cualidades esteganográficas: En la esteganografía se definen tres cualidades: Capacidad, Robustez y la Imperceptibilidad. Al realizar el proceso esteganográfico se busca balancear estás cualidades. La Capacidad mide la cantidad de información que puede ser incrustada en un medio esteganográfico o carga útil, se mide en bits por píxel (bpp). La Robustez es la medida de la resistencia a los ataques realizados por sistemas de estegoanálisis.

La Imperceptibilidad mide el grado de indetectabilidad de la información secreta por un observador. Para valorar el efecto esteganográfico en imágenes como medio de ocultación de la información, se suelen utilizar índices que permiten valorar el grado de cambio en el medio de manera que el proceso genere estego objetos con el mínimo de distorsión, con la máxima carga y robustez. En este trabajo se calculan los Índice de Calidad de Imagen (IQAM), el Error Cuadrático Medio (MSE) la Relación de Señal a Ruido (PSNR) y la Fidelidad de la Información Visual (VIF) [8, 9].

Los algoritmos esteganográficos deben de mantener un equilibrio es sus parámetros, que permitan mantener la eficiencia y seguridad de estos y lograr que, al aumentar la imperceptibilidad no se sacrifique la robustez o al aumentar la robustez no se limite la capacidad.

Fidelidad de la información Visual (Visual Information Fidelity): Este modelo retoma algunos conceptos de la teoría de la información como el modelado de escenas naturales (NSS) y el sistema visual humano (HVS). Para su aplicación en esteganografía, toma la imagen de portada (*C*) para cuantificar su información, y la imagen estego (*S*) para saber la cantidad de información que puede ser extraída. Al combinarlas se puede medir la degradación que puede sufrir la imagen estego [10]. Este índice (ir a las referencias para los detalles) se representa por medio de la ecuación (1):

$$VIFF(S_k, \mathrm{C_b}) = \frac{\sum_K \sum_b \log_2 \left(1 + \frac{g_{k,b}^2 * \left(\sigma_{\mathrm{k,b}}\right)^2}{\left(\left(\sigma_{k,b}^d\right)^2 - g_{k,b}^2 * \left(\sigma_{k,b}^r\right)^2 + {\sigma_N}^2\right)} \right)}{\sum_k \sum_b \log_2 \left(1 + \frac{\left(\sigma_{k,b}^r\right)^2}{{\sigma_N}^2} \right)}. \quad (1)$$

Error Cuadrático Medio (Mean Square Error): Para medir el error que hay entre dos conjuntos de datos y el cálculo se realiza promediando la intensidad al cuadrado de la imagen original (*C*) y los pixeles de la estego imagen (*S*) donde *M* y *N* son filas y columnas respectivamente [6]. Este cálculo servirá también en la métrica PSNR la cual está definida por la ecuación 2:

$$MSE = \frac{1}{M\ xN} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(C_{i_j} - S_{i_j} \right)^2. \quad (1)$$

Un MSE igual a 0 indica que las imágenes son idénticas.

Métrica Relación Señal a Ruido Pico (PSNR): Esta métrica se usa para observar las distorsiones que existen entre la imagen original y la imagen estego, además puede indicar si existe un grado de distorsión mayor al momento de realizar la compresión en una imagen y se define por la siguiente ecuación (3):

$$PSNR = 10xlog_{10}\left(\frac{255^2}{MSE}\right), \tag{2}$$

donde las unidades de PSNR son decibelios (dB). Para la ecuación (3) se toma 255 por ser la intensidad máxima para un píxel. La clasificación PSNR para una imagen se realiza con el siguiente criterio: $PSNR$ < 30 dB es no aceptable, $PSNR$ entre 30 a 40 dB es aceptable y $PSNR$ > 40 dB es muy buena [6].

Índice de Calidad de Imagen Universal (UIQI): La métrica UIQI tiene la capacidad para medir la pérdida de información ocurrida durante los procesos que impliquen la degradación de la imagen tomando como base tres índices: pérdida de correlación, distorsión de luminancia y distorsión de contraste. El rango de evaluación en la calidad de imagen es dinámico, va de -1 a 1, entre más cercano es a 1 mejor será la calidad de la imagen evaluada. En la ecuación (4) se define esta métrica [11]:

$$\text{UIQI} = \frac{4\sigma\text{CS}}{\sigma^2 c + \sigma^2 s} \times \frac{\bar{C}\,\bar{S}}{C^{-2}\,S^{-2}}. \tag{4}$$

Índice de Semejanza Estructural (SSIM): La métrica SSIM [8], determina la similitud que existe entre una imagen estego y la original y se determina por la ecuación (5):

$$SSIM(s,y) = \frac{\left(2\mu_x\mu_y + c_1\right)\left(2\sigma_{x_y} + c_2\right)}{(\mu_x{}^2 + \mu_y{}^2 + c_1)(\sigma_x{}^2 + \sigma_y{}^2 + c_2)}. \tag{5}$$

Cuando c1 = 0 y c2 = 0, SSIM se vuelve igual a Q como un caso especial. Mientras el resultado de SSMI se acerque a 1, el rendimiento será mejor.

1.2. Código de respuesta rápida, QR (Quick Response Code)

Los códigos Quick Response (QR) fueron creados por la compañía Denso Wave Inc. para facilitar el acceso instantáneo de información al codificarla para que sea recuperada por medios ópticos. Están normados en el estándar ISO/ EEC18004:2006 y es de acceso libre.

El código QR es una imagen de dos dimensiones que codifica información por medio de puntos oscuros sobre un fondo claro. Los patrones de imagen codifican ciertas estructuras que facilitan la identificación de parámetros para la interpretación, almacenamiento y corrección de errores en la información [12].

En la Tabla 1 se resumen el tipo de formato (numéricos, alfanuméricos, binario y Kanji), capacidad límite y caracteres que admite el código QR para su implementación en el Modelo 1 y 2 que son los más frecuentes.

Tabla 1. Tipos Datos y capacidad límite que admite el código QR [13].

Formato	Capacidad de datos	Caracteres
Numérico	7089 caracteres	0-9
Alfanumérico	4296 caracteres	0-9, A-Z (mayúsculas) espacio $ % ‘+ -, /:
Binario	2953 bytes	Codificación por defecto: ISO 8859-1 (QRC 2005)
Kanji	1817 caracteres	Desplazamiento JIS X 0208

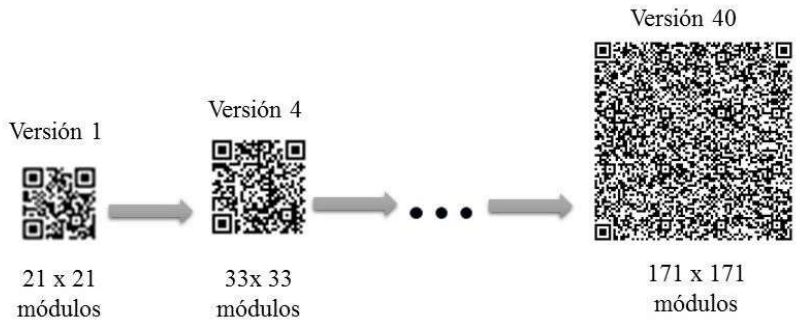


Fig. 2. Ejemplo de diferentes versiones código QR [4].

La versión para un código QR depende del tamaño de la matriz de puntos de imagen (módulos). El Modelo 1, va de la versión 1 hasta la versión 14; al Modelo 2 pertenecen de la versión 15 hasta la 40. La versión corresponde a la dimensión de la matriz del código QR y está relacionado con su capacidad de caracteres de información que soporta. Las versiones van de $n \times n$, donde $n = 17 + 4 \cdot i$, donde i=1, …, 40. Así que la versión 1, es un QR de tamaño 21×21; la versión 2, es un QR de tamaño 25×25, respectivamente.

Otro factor que está relacionado con la capacidad de datos de un código QR, es el nivel del factor de corrección de errores (ECC). Aquí se usa una codificación tipo Reed-Solomon y se especifican cuatro niveles: L (7%), M (15%), Q (20%) y H (30%), entre más alto el nivel de corrección de errores, mayor es la capacidad para recuperar la información por deterioro de la imagen, pero reduce la capacidad de datos de información.

El tamaño menor de los códigos QR corresponde a la versión 1, de 21×21 módulos, se le suman 2 módulos de margen a los lados, dando 25 × 25 módulos del código QR, cada módulo es de 4 píxeles, por lo que la imagen tiene una dimensión de 100 × 100 píxeles. Al imprimirlo a 100 pixeles por cm (100 ppcm.) el código QR será de 1 cm de ancho y 1 cm de alto.

Para la distancia de detección se recomienda la regla de 1:10, tal que la distancia de detección de un código de versión 1 es de al menos 10 cm. En la Fig. 2, se observa el código QR en versión 1 la cual es la versión mínima, al ir aumentando su capacidad van también aumentando sus módulos y su complejidad, como se observa en la versión 40 [4].

Tabla 2. Relación entre Imagen oculta e imagen cubierta.

Versión código QR oculto	Módulos	Módulos + margen: N	$2\sqrt{2}N$	Módulos sin margen	Módulos por versión	Versión de la imagen cubierta
1	21	25	71	67	69	13
2	25	29	83	79	81	16
3	29	33	94	90	93	19
4	33	37	105	101	101	21
5	37	41	116	112	117	25
6	41	45	128	124	125	27
7	45	49	139	135	137	30
8	49	53	150	146	149	33
9	53	57	162	158	161	36
10	57	61	173	169	169	38
11	61	65	184	180	-	-

1.3. Revisión de la literatura sobre el tema esteganografía y códigos QR

Los métodos esteganográficos se han utlizado para embeber información en códigos QR, como por ejemplo en [14] se presenta un resumen de diferentes trabajos relacionados sobre el tema, mostrando como se utiliza un código QR como imagen de portada y otro código QR oculto embebido y encriptado por medio de una clave secreta.

La lectura del código QR oculto se lleva a cabo por medio de un lector modificado. Además, presenta el uso de las técnicas esteganográficas en el dominio de frecuencia por medio de la Transformada Wavelet Discreta Haar, debido a que han mostrado robustez a ataques de estegoanálisis. Otra técnica utilizada es la LSB en su forma de sustitución o secuencial, por mostrar una gran capacidad de carga útil.

Así mismo presenta formas para encriptar la información utilizando métodos tipo AES (Estándar Avanzado de Encriptación) y por el método de tipo RSA (Rivest, Shamir y Adleman). Masoud Alajmi et. al. en [15] proponen un sistema utilizando un código QR como portada con información común y encriptación AES o RSA, para una imagen oculta. Utilizan la reversibilidad del algoritmo LSB y modifican el mensaje en código ASCII a binario para realizar el proceso esteganográfico con el objetivo de que al ser atacado el sistema el mensaje secreto sea confundido con ruido.

2. Desarrollo

2.1. Dimensiones para embeber un código QR en otro código QR.

La esteganografía ha tenido un interés particular como medio para ocultar él envió de información por medios sin que despierten sospecha. Está claro que existe la posibilidad de que un mensaje oculto se vea como una falla de seguridad; sin embargo, podría ser un medio para transmitir información confidencial. Entonces es un debate permanente el temor que tienen algunos agentes a lo que se les oculta y el derecho a la confidencialidad de la comunicación.

Tabla 3. Cálculo de formato físico para código QR.

Versión código QR oculto	Módulos más margen	Píxeles (4 p/m)	Dimensión (100 ppcm)	Módulos con imagen oculta	Píxeles	Dimensión (cm.×cm.)
1	25	100	1	73	292	2.92
2	29	116	1.16	85	340	3.4
3	33	132	1.32	97	388	3.88
4	37	148	1.48	105	420	4.2
5	41	164	1.64	121	484	4.84
6	45	180	1.8	129	516	5.16
7	49	196	1.96	141	564	5.64
8	53	212	2.12	153	612	6.12
9	57	228	2.28	165	660	6.6
10	61	244	2.44	173	692	6.92

Desde este punto de vista, el explorar la posibilidad de incrustar información en un código QR, dado su uso general en el ciberespacio, ya presenta por sí mismo un tema de interés. El aporte de este trabajo es definir la estructura de la imagen cubierta y la imagen estego utilizando códigos QR y su importancia al implementarlas en un sistema esteganográfico, ya que al realizar los cálculos de capacidad de los dos elementos se puede aprovechar de forma eficiente la carga útil.

Así al utilizar los códigos QR como elementos de un sistema esteganográfico se debe tomar en cuenta su versión, los niveles de corrección, además de la capacidad de información que es posible almacenar. También es requisito conocer el tamaño mínimo por pixeles que debe contener, para evitar que la lectura sea errónea al ocultar o extraer información, además en el caso del algoritmo LSB al medir los niveles de degradación pasando por los distintos bits que componen una imagen es posible conocer que tan eficiente puede ser el código QR en sus niveles de corrección.

Al ser el algoritmo LSB reversible en la imagen secreta se puede implementar como una marca de agua, dotando entonces a estos elementos de un nivel especial de seguridad, que pudiera aplicarse para autentificar la procedencia y autoría de distintos elementos que interactúan en el entorno social, tales como: documentos públicos y privados, medicamentos, ropa, artículos domésticos etc. Las marcas de aguas digitales buscan esconder una serie de bits dentro de un medio digital para identificar autoría, distribuidores, documentos, e imágenes, etc.

El diseño debe permitir que sea invisible, para evitar su manipulación. Al implementar una marca de agua eficaz se deben de contemplar tres parámetros: la capacidad, la robustez y la imperceptibilidad, en el presente trabajo se abordarán y se pondrán a prueba dos áreas, a) la capacidad y b) la imperceptibilidad, estas características son similares a las esteganográficas descritas en la sección 1.1 [13]. Para la implementación esteganográfica en la imagen del QR cubierta y del código QR oculto se convirtieron las imágenes a tonos de gris. Como se explicó en la sección 1.1.

El algoritmo LSB, incorpora la información del código QR oculto en los bits menos significativos de los valores de tono de gris de los píxeles código QR de cubierta.

Tabla 4. Parámetros de las imágenes de Código QR.

Imagen Oculta				**Imagen Portada**			
Version	Modulos	ECC	Alfanumerico	Versión	Módulos	ECC	Alfanumerico
1	21 x21	L	41	13	69 x 69	L	619
10	57x 57	H	174	13	69x69	H	269
				38	169 x169	H	1591

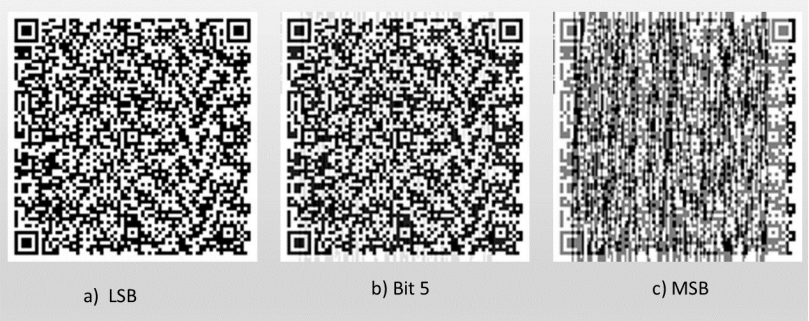


Fig. 3. Aplicación de esteganografía a imágenes con código QR.

Esto provoca que a lo más el cambio de tono de los pixeles del código QR cubierta sea de $\pm\ 1/255$, esto es menos del 0.4% en el tono del píxel, lo que hace que sea difícilmente perceptible de forma visual, ya que el ojo humano puede detectar en promedio sólo 30 tonos de gris.

Lo que implica que el cambio tonal deber ser de poco más de 8 en el cambio para el bit menos significativo para hacerlo perceptible. La imagen del código QR secreto en capacidad, es la versión 1 que cuenta con 21×21 módulos, más 2 módulos en el margen de zona de silencio, lo que da una imagen de dimensiones de 25×25 módulos, que está relacionado con su tamaño físico al usar 4x4 píxeles para cada módulo (100×100 pixeles) y una resolución de 100 píxeles por centímetro.

Para estimar el número de píxeles que se requieren de la imagen del código QR cubierta, se considera una matriz cuadrada de píxeles en el intervalo {0, 255}, es decir, equivalente a 8 bits. Se usa el bit LSB de cada 8 para guardar por sustitución un bit de información de la imagen oculta.

De forma general, si la cantidad de datos de la imagen cubierta de $N\times N$ es N^2, y el número de datos de la imagen oculta de tamaño $M\times M$ es M^2, la relación entre estos números es dada (6). Para el cálculo de la imagen cubierta se parte de la versión y el número de módulos que se ocultan, se suman dos módulos por lado de margen, se aplica la formula (6):

$$N^2 = \frac{M^2}{8} \text{lo que implica: } 2\sqrt{2}N = M\ . \tag{6}$$

El resultado es el módulo total para la imagen cubierta, se le restan 2 módulos de margen por lado y el resultado es el número de módulos mínimo para la imagen cubierta, finalmente se selecciona la versión que tenga igual o mayor número de módulos [16]. Este proceso se representa en la Tabla 2.

Tabla 5. Resultados para las métricas UIQI, SSMI, VIF y PSNR.

Conjunto A (imágenes QR versión 1 incrustada en QR versión 13 y ECC L)					**Conjunto B (imágenes QR versión 1 incrustada en QR versión 13 y ECC H)**			
Bit	UIQI	SSMI	VIF	PSNR	UIQI	SSMI	VIF	PSNR
1	0.99999	1	0.9893	51.56	1	1	0.9893	51.5367
2	0.99996	0.9999	0.9639	45.5394	1	0.9999	0.964	45.5161
3	0.99983	0.9998	0.9001	39.5188	0.9998	0.9997	0.9003	39.4955
4	0.99932	0.999	0.7901	33.4982	0.9993	0.999	0.7905	33.4749
5	0.99729	0.9958	0.6451	27.4776	0.9973	0.9957	0.6458	27.4543
6	0.98927	0.9826	0.4755	21.457	0.9892	0.9822	0.4764	21.4337
7	0.95793	0.9246	0.2864	15.4364	0.9576	0.9229	0.2868	15.4131
8	0.84256	0.6776	0.0964	9.41581	0.8414	0.6716	0.0933	9.39253

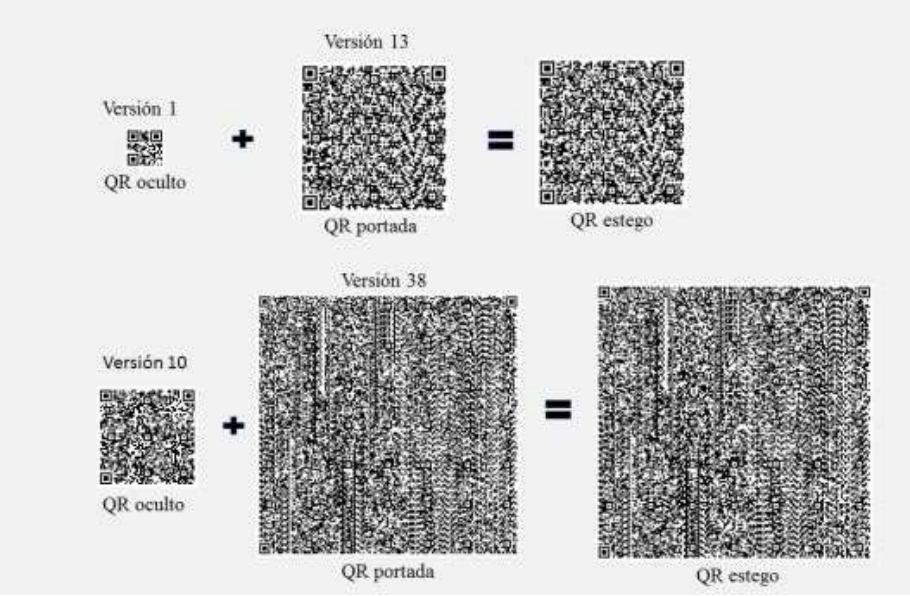


Fig. 4. Cambios en las imágenes estego del conjunto A al cambiar los bits a) LSB, b) 5to. bit y c) MSB.

Como se puede apreciar en la primera fila, si se quiere ocultar un código QR de versión 1 se necesita un código QR cubierta de versión 13. La Tabla 3 nos muestra el tamaño físico del código QR oculto y de cubierta.

Por ejemplo, de la primera línea, un código QR oculto de versión 1 de 21+4=25 módulos con una resolución de 4 píxeles por módulo, si se imprime a una resolución de 100 píxeles por cm, el código mide 1 cm por lado.

Un código QR correspondiente a una versión 13 de 73 módulos (69 módulos más 4 módulos de margen) son suficientes para incluir los datos de imagen del código QR oculto, puesto con 4 píxeles por módulo dan 292 píxeles que a una resolución de 100 ppcm. tendrá un tamaño de 2.92 cm × 2.92 cm.

Tabla 5. Resultados para las métricas UIQI, SSMI, VIF y PSNR.

Conjunto A (imágenes QR versión 1 incrustada en QR versión 13 y ECC L)					**Conjunto B (imágenes QR versión 1 incrustada en QR versión 13 y ECC H)**			
Bit	UIQI	SSMI	VIF	PSNR	UIQI	SSMI	VIF	PSNR
1	0.99999	1	0.9893	51.56	1	1	0.9893	51.5367
2	0.99996	0.9999	0.9639	45.5394	1	0.9999	0.964	45.5161
3	0.99983	0.9998	0.9001	39.5188	0.9998	0.9997	0.9003	39.4955
4	0.99932	0.999	0.7901	33.4982	0.9993	0.999	0.7905	33.4749
5	0.99729	0.9958	0.6451	27.4776	0.9973	0.9957	0.6458	27.4543
6	0.98927	0.9826	0.4755	21.457	0.9892	0.9822	0.4764	21.4337
7	0.95793	0.9246	0.2864	15.4364	0.9576	0.9229	0.2868	15.4131
8	0.84256	0.6776	0.0964	9.41581	0.8414	0.6716	0.0933	9.39253

2.2. Calidad de imagen del uso del algoritmo LSB y códigos QR

Las pruebas se llevaron a cabo en imágenes con código QR con contenido codificado por textos alfanuméricos. Se eligió este tipo de información porque es comúnmente utilizada tanto en medios digitales como en impresos. El programa utilizado para la creación de las imágenes con código QR fue codificado en lenguaje Python, usando la librería QR Code [17].

Para las pruebas, se implementaron las configuraciones indicadas en la Tabla 2, lo que representa modificar el número de versión, el factor de corrección y valor de pixeles por cada cuadro. Las imágenes están en formato PNG, los datos de las imágenes creadas para el proceso esteganográfico están divididas en imágenes ocultas y de portada. Se implementó el algoritmo LSB codificado en Matlab según la descripción de la sección 2.1.

Las imágenes con código QR de la Tabla 4 muestran el efecto de usar el LSB hasta el bit más significativo (MSB) para ocultar la información. El propósito de insertar los bits de la imagen secreta en bits de mayor peso que el LSB es para medir como se altera la imagen de cubierta al insertar la imagen secreta.

La aplicación del algoritmo esteganográfico LSB se da por medio de los siguientes pasos: Primero se elige el código QR oculto y el código QR cubierta de acuerdo con los valores de la Tabla 2, para responder a las características de carga útil que se calcula en la imagen cubierta de acuerdo con la ecuación (6) y en formato PNG, este formato se utilizó para evitar perdida de datos por compresión.

En el segundo paso se realiza el proceso de reemplazo de los bits con el algoritmo LSB para crear una estego imagen que contiene la imagen oculta, pero además de usar el LSB se aplica el algoritmo esteganográfico en bits más significativos. Este proceso se realizó para crear estego imágenes para cada uno de los 8 bits que se tienen en la imagen cubierta.

En la Fig. 3 se muestra el resultado del proceso de aplicación del algoritmo LSB para dos imágenes a ocultar de acuerdo con las propiedades de la tabla 4, en las cuales para la versión 1 incrustada en versión 13, se aplicó un nivel de corrección de error bajo (ECC L). En la versión 10 incrustada en la versión 38, se usó un nivel de corrección de error alto (ECC H). El efecto en la estego imagen, en ambos casos, no afectó la decodificación del texto con el teléfono inteligente.

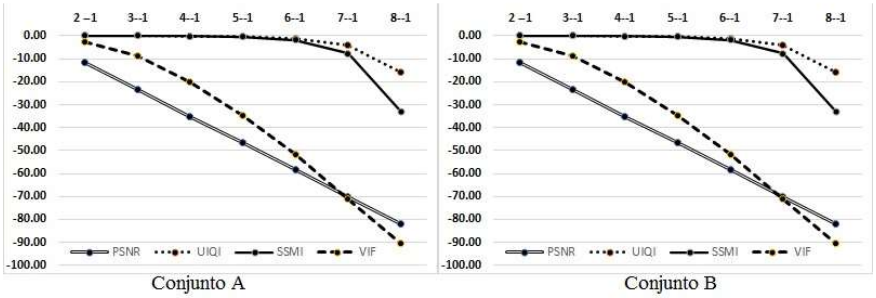


Fig. 5. Tasa porcentual relativa al LSB para el conjunto A de imágenes y B.

Al final se utiliza la llave del algoritmo LSB para recuperar la imagen oculta, recuperándose correctamente. Para conocer el grado de similitud, calidad de imagen y fidelidad visual entre la imagen de portada y la estego imagen se utilizaron las métricas propuestas en la primera sección programadas en el lenguaje Python con la librería "Sewar" que permite el cálculo PSNR, SSIM, UQUI y VIF [18].

Las imágenes que se tomaron como portada fueron las de un código QR versión 13 y la imagen oculta fue la versión 10, descritas en la Tabla 4, el proceso esteganográfico que se realizó en ellas fue desde LSB (1er. bit) hasta MSB (8vo. bit) formándose dos conjuntos de imágenes A: imágenes de un código QR versión 1 incrustada en código QR versión 13 y ECC L, y B: imágenes QR versión 1 incrustada en imagen de un código QR versión 13 y ECC H. Los cálculos de los índices para cada conjunto de imágenes se resumen en la Tabla 5.

Al aplicar el algoritmo esteganográfico desde el LSB hasta el MSB, la imagen de portada sufre varios cambios como se observa en la Fig. 4. Este comportamiento fue similar para el conjunto de imágenes A y B descritos en la Tabla 5, del 2do bit al 4to bit se empezó a mostrar degradación en la imagen en tonos visibles en gris que son perceptibles al aplicar una ampliación del 50%, y detectable en los valores de las métricas VIF y PSNR que se muestran en la Tabla 5, donde se presentan los índices de calidad entre la imagen estego y la imagen cubierta.

Notar que la tendencia a descender su valor, deduciendo que la fidelidad de la imagen se empieza a deteriorar, pero las métricas SSMI y UIQUI no muestran cambios significativos y se mantienen estables a pesar de mostrar un decremento en la calidad de la imagen cubierta como se muestra en la Fig. 4.

La aplicación del algoritmo en el 5to bit al 8vo bit (MSB), es donde se muestran cambios más significativos en la imagen estego como se observan en la Fig. 4b y 4c. Se notan marcas de la imagen oculta. Al usar el 8vo bit, la imagen estego presenta decoloración importante. Los índices VIF y PSNR son malos y la degradación de la imagen es notable. Las métricas UIQUI y SSMI muestran cambios relevantes por el uso del 7mo bit u 8vo bit, demostrando que los cambios deterioran considerablemente la capacidad de similitud y calidad de imagen.

En la Fig. 5, se muestra la tasa de cambio relativo para el conjunto A y B de imágenes estego descritas en la Tabla 5, muestran que a medida que se va cambiando el bit en la imagen cubierta, la imagen estego va perdiendo sus características de Fidelidad Visual, Similitud y Calidad de Imagen.

El cambio más significativo se observa en las métricas VIF y PSNR, mientras que en las métricas SSIM y UIQI el índice de cambio es visible a partir del 6to bit. Es notable que el comportamiento de los índices SSIM y UIQI, se asocia más a la pérdida de calidad en la imagen estego. Este resultado se interpreta como que en este tipo de imágenes podría incrustarse más de una imagen de código QR, al observar que podrían usarse además del bit LSB ya que la percepción en la imagen estego ocurre hasta el 6to bit.

Esto podría incrementar la capacidad de carga de la imagen cubierta. También se observa que la propiedad ECC de los códigos QR de cada conjunto a pesar de ser diferente no interfiere directamente en los índices de la métrica ya que el valor es similar para ambos conjuntos.

3. Conclusiones

El proceso esteganográfico llevado a cabo en las imágenes de portada con código QR, fue posible al usar los parámetros calculados e indicados en la Tabla 4. Al aplicar el algoritmo LSB en la imagen de cubierta y producir la imagen estego, pasó completamente desapercibida y por los cálculos obtenidos de las métricas de calidad de imagen, se nota que el cambio poco perceptible.

Los cambios importantes en la imagen estego se mostraron al usar el algoritmo esteganográfico en el 5to bit 5 al 8vo bit, donde el parámetro de imperceptibilidad se reduce, haciendo que el proceso esteganográfico sea detectable. La lectura de la imagen estego, la cual contiene información del código QR, fue posible en una distancia de 10 veces la dimensión física del código QR. Al conocer las proporciones que debe contener una imagen con código QR para poder ocultarla dentro de otra, permite que pueda pasar desapercibida e indicar que podría usarse más de un bit y así aumentar la cantidad de información que podría contener la imagen de portada.

Al observar la Fig. 5, fue interesante notar que los índices SSIM y UIQI mostraron como la calidad de la imagen estego no es afectada en una escala importante al usar los bits del 1 (LSB) a 6to bit de la imagen estego. Al ser el algoritmo LSB reversible el proceso de la recuperación del código QR oculto se logró satisfactoriamente en todos los casos. Al recuperar el QR podría usarse como como una marca de agua y aplicarse para autentificar códigos QR. Como trabajo futuro se buscará probar la robustez de las imágenes estego con código QR y así, conocer la resistencia ante ataques de estegoanálisis.

Agradecimientos. Al CONACYT por el apoyo a la Maestría en Ciencias de la Computación de la Universidad Autónoma del Estado de México.

Referencias

1. Yahya, A.: Steganography techniques for digital images. Palapye.Springer (2019)
2. Caballero, H., Muños, V., Ramos, M., Romero, M.: Steganography method through fractal dimension and lsb algorithm: A new perspective on RGB images. Tecnología Educativa Revista CONAIC, vol. 6, no. 1, pp. 25–30 (2019)

3. Centurión, A., Soria, A., Moreno, E.: Un algoritmo esteganográfico vinculado a los cuadrados mágicos. REDEL, Revista Granmense de Desarrollo Local, vol. 3, no. 4, pp. 225–238 (2019)
4. Denso Wave Incorpored.QRCode®. Essentials (2021) https://www.qrcode.com/en/about/version.html. [Último acceso: 1 abril 2021]
5. Muñoz, A.: Privacidad y ocultacion de informacion digital Esteganografía. Protegiendo y atacando redes informáticas, RA-Ma, pp. 35–38 (2017)
6. Mahdi, M., Mohd, M. S., Abass, F., Sabah, M., Salman, H.: Performance evaluation measurement of image steganography techniques with analysis of LSB based on variation image formats. International Journal of Engineering & Technology, vol. 7, no. 4, pp. 3505–3514 (2018)
7. Mahdi, M., Mohd, M., Abass, F., Sabah, M., Al-Wan, A., Amir, N.: An extensive analysis and conduct comparative based on statistical attach of LSB substitution and LSB matching. International Journal of Engineering & Technology, vol. 7, no. 4, pp. 4008–4023 (2018)
8. Jawad, I., Premaratne, P., Vial, P. J., Halloran, B.: Comprehensive survey of image steganography: Techniques, evaluations, and trends in future research. Neurocomputing, vol. 335, pp. 299–326 (2019)
9. Muhammad, K., Ahmad, J., Jan, Z., Sajjad, M., Rehman, N. U.: CISSKA-LSB: Color image steganography using stego key-directed adaptive LSB substitution method. Multimed Tools and Applications, vol. 76, no. 6, pp. 8597–8626 (2017)
10. Pramanik, S., Singh, R., Ghosh, R.: Application of bi-orthogonal wavelet transform and genetic algorithm in image steganography. Multimedia Tools and Applications, pp. 1–20 (2020)
11. Guzmán, Y., Pérez, E., Centurión, A.: Un algoritmo esteganográfico adaptativo para lograr mayor indetectabilidad. Lecturas Matemáticas, vol. 41, no. 2, pp.149–164 (2020)
12. Castro-Acuña, N., Leguizamón-Páez, M., Mora-Lancheros, A.: Análisis de métodos y técnicas existentes para minimizar agujeros de seguridad al usar códigos QR. Revista UIS Ingenierías, vol. 18, no. 4, pp. 157–172 (2019)
13. Padrón, A., Prieto, R., Treviño, C.: Clave óptica privada mediante un código QR cifrado. Sistemas, Cibernética e Informática, vol. 17, no. 2, pp. 5–15 (2020)
14. Remya, P.: A review on stegnography in QR codes. International Research Journal of Engineering and Technology (IRJET), vol. 5, no. 5, pp. 4294–4296 (2018)
15. Alajmi, M., Elashry, I., El-Sayed, H., Faragallah, O.: Steganography of encrypted messages inside. IEEE Access, vol. 8, pp. 27861–27873 (2020)
16. Luque, J.: Códigos QR. Manual formativo de ACTA, vol. 63, pp. 9–28 (2012)
17. Loop, L.: The python package index (PyPI), pure python QR code generator. (2021) Available: https://pypi.org/project/qrcode/. [Último acceso: 12 abril 2021]
18. Khalel, A.: The python package index (PyPI). Project Sewar, (2021) Available: https://pypi.org/project/sewar/. [Último acceso: 7 abril 2021]

ISSN 1870-4069

Diseño de descriptores mediante evolución gramaticalpara el reconocimiento de imágenes de expresiones faciales

Manuel Alejandro Torres Fonseca[1], Valentín Calzada Ledesma[2], Manuel Ornelas Rodríguez[1], Alfonso Rojas Domínguez[1], Juan Martín Carpio Valadez[1], Héctor José Puga Soberanes[1]

[1] Tecnológico Nacional de México, Instituto Tecnológico de León, León, Guanajuato, México

[2] Tecnológico Nacional de México, Instituto Tecnológico Superior de Purísima del Rincón, División de Ingeniería Informática, Guanajuato, México

321ctorres@gmail.com, 14240803@leon.tecnm.mx

Resumen. El reconocimiento de expresiones faciales representa una importante tarea en las áreas de visión por computadora y reconocimiento de patrones. La obtención de características suficientemente discriminativas para reconocer diferentes emociones resulta ser una etapa de suma dificultad. En este trabajo se presenta una metodología basada en Evolución Gramatical e Histograma de Gradientes Orientados para la generación automática de descriptores capaces de caracterizar adecuadamente imágenes de expresiones faciales. Este método se aplicó a la base de datos JAFFE, obteniendo un porcentaje de clasificación promedio del 97.78 %.

Palabras clave: Reconocimiento de expresiones faciales, descriptores de imágenes, evolución gramatical, histograma de gradientes orientados.

Descriptors Design Using Grammatical Evolution for Facial Expression Images Recognition

Abstract. In the area of Web application development there are methodologies as UWE (UML Web Engineering) [1] and W2000 [2] among others, however, it is common practice for developers skip their use and perform in empirical way rather than orderly the development of the applications; when a methodology is used, usually its models are built based on the Unified Modelling Language (UML), which implies higher level of knowledge thereof for proper application of methodologies based on it. Therefore, in this paper are proposed a series of artifacts based on the use of graphs and set theory to design the conceptual modeling of Web applications to achieve the desired results reliably and expeditiously by the developer.

Keywords: Face emotion recognition, image descriptors, grammatical evolution, histogram of oriented gradients.

1. Introducción

Dentro de la comunicación humana, las expresiones faciales toman un rol muy importante al momento de entablar una conversación; un gran porcentaje del mensaje que se quiere comunicar se muestra a través de estas expresiones, por lo que se da un indicio del contexto de la información que se está transmitiendo, así como también del estado de ánimo de la persona.

Actualmente, el estudio formal de las emociones se ha incrementado y existen una gran cantidad de autores que han contribuido en esta área de estudio. Uno de los aportes más importantes fue realizado por Ekman, en su libro titulado: "The nature of emotion: Fundamental questions", en donde se propuso una estandarización de las emociones, formando un grupo de seis emociones básicas: enojo, miedo, disgusto, felicidad, tristeza y sorpresa [1].

La estandarización de las emociones humanas fue uno de los puntos de partida que hicieron posible el reconocimiento de expresiones faciales. Esto permitió un nuevo enfoque para la interacción humano máquina. De esta manera, ha sido posible adaptar herramientas de visión por computadora para el análisis de las emociones, lo que ha permitido el desarrollo de diversos instrumentos que ayudan, por ejemplo, a la identificación de un posible agresor, la predicción del comportamiento de personas en hospitales o instituciones psiquiátricas, sistemas de reconocimiento biométrico, sistemas de seguridad en aeropuertos, etc [2].

Cualquier sistema de reconocimiento de expresiones faciales usualmente se basa en las siguientes fases (ver Fig. 1): Adquisición de imágenes de rostros, Extracción de características y Clasificación.

La primera fase consiste en la obtención y preprocesamiento de las imágenes de los rostros o secuencias de videos, en el preprocesamiento se realiza un mejoramiento de las imágenes adquiridas previo a la extracción de características, por ejemplo, se puede segmentar y eliminar el fondo de las imágenes para únicamente tomar en cuenta zonas de interés en la imagen. Otro ejemplo es la aplicación de filtros que permitan eliminar el ruido causado por la iluminación.

La extracción de características es una de las fases más importantes dentro del sistema, usualmente se lleva a cabo a través de la aplicación de descriptores los cuales son algoritmos computacionales que extraen y empaquetan los atributos significativos de las imágenes en vectores de características. Es deseable que estos atributos caracterizados permitan una buena separabilidad entre clases (emociones), permitiendo al clasificador tener un buen desempeño.

Finalmente, en la fase de clasificación, los vectores de características se etiquetan o asocian a una clase cuyas características son similares; esto se realiza mediante los algoritmos clasificadores, los cuales utilizan un indicador (porcentaje de clasificación) que permite conocer la cantidad de vectores que fueron asociados con sus respectivas clases de manera correcta.

Existe una variedad de descriptores que pueden ser utilizados para caracterizar imágenes, por ejemplo, Local Binary Pattern (LBP), Filtros de Gabor, Histograma de Gradientes Orientados (HOG por sus siglas en inglés), entre otros. Estos algoritmos han sido diseñados por expertos en el campo para abordar un determinado tipo de problemas, sin embargo, es común observar que el rendimiento de éstos puede ser menor cuando son aplicados a otros diferentes tipos de problemas; de esta manera se

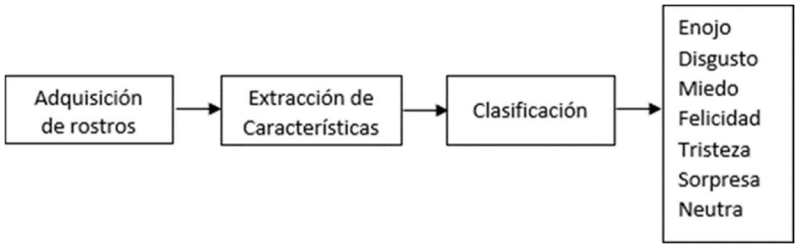


Fig. 1. Sistema de reconocimiento de expresiones faciales [3].

puede decir que no existe un descriptor perfecto. Este hecho ha motivado a la creación de metodologías para la generación automática de descriptores que se adapten a un conjunto de imágenes en particular para poder obtener un mejor desempeño en el proceso de clasificación [4].

En este artículo se propone una metodología basada en Evolución Gramatical y HOG, para la generación automática de descriptores aplicados al reconocimiento de expresiones faciales, utilizando como instancia de prueba una de las bases de datos más citadas en el estado del arte, el Japanese Female Facial Expression (JAFFE). Una Máquina de Vector Soporte es utilizada como algoritmo de clasificación.

El trabajo está organizado de la siguiente manera: la sección 2 muestra conceptos teóricos relacionados con la presente propuesta; la sección 3 describe la metodología empleada; en la sección 4 se explican los experimentos realizados y los resultados obtenidos y finalmente en la sección 5 se presentan las conclusiones.

2. Marco teórico

Evolución Gramatical (GE):es considerada una variación de la Programación Genética [10], la cual utiliza una gramática formal relacionada con el problema para generar un fenotipo o programa, el cual es codificado a partir de un arreglo numérico denominado cromosoma genotípico [10]. Además, usa un proceso de mapeo, el cual permite relacionar los elementos de la gramática con cada cromosoma para obtener un fenotipo. Un motor de búsqueda realiza el proceso evolutivo.

Gramática libre de contexto tipo Backus-Naur (BNF): es una notación formal que permite diseñar la sintaxis de un programa basándose en un conjunto de reglas de producción [10], estas reglas permiten la obtención del programa (fenotipo) tomando en cuenta símbolos divididos en terminales y no terminales. Los elementos que conforman la gramática son expresados mediante la tupla $\{N, T, P, S\}$, donde N representa los símbolos no terminales (indicados en la gramática entre signos '<' y '>'), T indica los símbolos terminales, P es el conjunto de reglas de producción y S es el símbolo inicial que indica la primera regla de producción que se aplica [11].

Proceso de Mapeo: se encarga de convertir un cromosoma genotípico en un fenotipo basándose en los elementos que conforman la gramática BNF (N, T, P, S); cada elemento del cromosoma recibe el nombre de codón. El algoritmo Depth-First (DF) fue usado como proceso de mapeo, el cual utiliza una regla de derivación que recorre los símbolos no terminales de la gramática hasta encontrar símbolos terminales. En la Fig. 2 se puede observar la regla de derivación y el proceso de mapeo, el cual

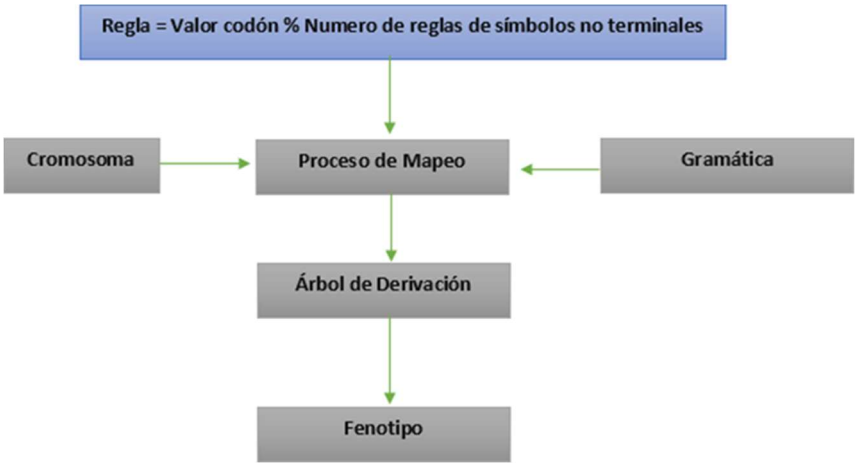


Fig. 2. Diagrama del proceso de mapeo [11].

recibe un codón para aplicar la regla a la gramática e ir formado el árbol de derivación hasta obtener el fenotipo.

A continuación, se muestra un ejemplo de este proceso de mapeo, utilizando la gramática BNF de la Fig. 3, la cual se encarga de la creación de operaciones entre números complejos. En la gramática el símbolo inicial *S* es: <*NumComplejo*>, los terminales *T* son: i, +, -, *, /, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, los símbolos no terminales *N* son: <*oper*>, <*op*>, <*signo*>, <*OpReal*>, <*numero*> y el conjunto de reglas de producción *P* se muestra en la Fig. 3.

El proceso de mapeo DF comienza con el símbolo inicial y procede a sustituir el símbolo no terminal localizado más a la izquierda, usando la regla de derivación basada en el operador módulo, hasta que encuentra un símbolo terminal. Enseguida se desplaza hacia la derecha para sustituir el siguiente no terminal. Este proceso continúa hasta que todos los símbolos no terminales del fenotipo generado son reemplazados por símbolos terminales.

En el ejemplo mostrado en la Fig. 4, el primer codón del cromosoma es 15 y el símbolo no terminal inicial tiene solo una regla de producción; aplicando la regla de derivación se obtiene $15\%1 = 0$, por lo que se selecciona la producción 0: *(<oper>)<op>(<oper>)*. El siguiente codón es 3 y el símbolo no terminal más a la izquierda *(<oper>)* tiene también una producción válida, dando como resultado $3\%1 = 0$, seleccionando <*signo*><*OpReal*><*signo*><*OpReal*>*i*.

El siguiente codón es 31 y el símbolo no terminal más a la izquierda *(<signo>)* tiene dos producciones válidas, dando $31\%2 = 1$ y seleccionando "-". Como se mencionó anteriormente, el proceso de mapeo finaliza cuando ya no existen símbolos no terminales en el fenotipo.

Motor de Búsqueda: consiste en un algoritmo de optimización capaz de encontrar soluciones adecuadas dentro de un determinado espacio de búsqueda. En este trabajo se empleó el algoritmo de Evolución Diferencial (DE por sus siglas en inglés). Para obtener una solución óptima, DE evoluciona un conjunto de soluciones candidatas con el fin de encontrar la solución mejor adaptada al problema. A cada solución se les

```
<NumComplejo> ::==  (<oper>)<op>(<oper>)      (0)
<oper> ::== <signo> <OpReal><signo> <OpReal> i     (0)
<signo> ::==  + |            (0)
                -            (1)
<OpReal> ::==<OpReal><numero>|           (0)
                <numero>                 (1)
<numero> ::==1 |         (0)
              2 |         (1)
              3 |         (2)
              4 |         (3)
              5 |         (4)
              6 |         (5)
              7 |         (6)
              8 |         (7)
              9 |         (8)
              0          (9)
<op> ::== + |        (0)
          - |        (1)
          * |        (2)
          /          (3)
```

Fig. 3. Ejemplo de Gramática BNF.

Cromosoma	15	3	31	25	61	48	7	5	12	16	3	17	36	22	41	18

(<oper>) <op> (<oper>)	15%1=0
(<signo> <OpReal> <signo> <OpReal> i) <op> (<oper>)	3%1=0
(- <OpReal> <signo> <OpReal> i) <op> (<oper>)	31%2=1
(- <numero> <signo> <OpReal> i) <op> (<oper>)	25%2=1
(- 2 <signo> <OpReal> i) <op> (<oper>)	61%10=1
(- 2 + <OpReal> i) <op> (<oper>)	48%2=0
(- 2 + <numero> i) <op> (<oper>)	7%2=1
(- 2 + 6 i) <op> (<oper>)	5%10=5
(- 2 + 6 i) + (<oper>)	12%4=0
(- 2 + 6 i) + (<signo> <OpReal> <signo> <OpReal> i)	16%1=0
(- 2 + 6 i) + (- <OpReal> <signo> <OpReal> i)	3%2=1
(- 2+ 6 i)+ (- <numero><signo> <OpReal> i)	17%2=1
(- 2 + 6 i) + (- 7 <signo> <OpReal> i)	36%10=6
(- 2 + 6 i) + (- 7 + <OpReal> i)	22%2=0
(- 2 + 6 i) + (- 7 + <numero> i)	41%2=1
(- 2 + 6 i) + (- 7 + 9 i)	18%10=8

Fig. 4. Proceso de mapeo DF para obtener operaciones de números complejos.

conoce como individuo y al conjunto de individuos se les denomina población [12]. En la Fig. 5 se muestra el algoritmo de DE.

Histograma de Gradientes Orientados (HOG): es un descriptor que utiliza la magnitud y la orientación del gradiente en porciones localizadas de una imagen denominadas "celdas"; estas cantidades son concatenadas construyendo un histograma de una dimensión tomando en cuenta los niveles de gris de los pixeles y usando las ecuaciones (1) y (2) [5, 13]:

F=coeficiente de mutación ϵ [0.4,0.9]
C=coeficiente de cruza ϵ [0.1,1]
Inicializar población de soluciones candidatas $\{x_i\}$ para *i* desde 1 hasta *N*
Repetir mientras sea diferente(criterio de paro)
  Repetir por cada individuo x_i para *i* desde 1 hasta *N*
    r1 ← entero aleatorio ϵ [1,*N*] : *r1* ≠ *i*
    r2 ← entero aleatorio ϵ [1,*N*] : *r2* ≠ {*i* , *r1*}
    r3 ← entero aleatorio ϵ [1,*N*] : *r3* ≠ {*i* , *r1*, *r3*}
    $v_i \leftarrow x_{r1} + F(x_{r2} - x_{r3})$ vector mutado
    $\mathcal{J}_r \leftarrow$ entero aleatorio ϵ $[1, n]$
    Repetir por cada dimensión para *j* desde 1 hasta *n*
      $r_{cj} \leftarrow$ numero aleatorio ϵ [0,1]
      Si ($r_{cj} < C$) o (j == $\mathcal{J}$)
        $v_{ij} \leftarrow v_{ij}$
      De otro modo
        $v_{ij} \leftarrow x_{ij}$
      Termina condicional "Si"
    Siguiente dimensión
  Siguiente individuo
  Repetir por cada población desde *i* igual a 1 hasta *N*
    Si $f(v_i) < f(x_i)$
      $x_i \leftarrow v_i$
  Siguiente población
Siguiente generación

Fig. 5. Pseudocódigo de Evolución Diferencial [12].

$$Magnitud: |\Delta f| = \sqrt{G_x^2 + G_y^2}\,, \tag{1}$$

$$Orientación: \ \theta = \tan^{-1}\left(\frac{G_y}{G_x}\right), \tag{2}$$

donde G_x corresponde al gradiente en el eje horizontal y G_y es el gradiente vertical.

La información es acumulada en un número N de bins (cantidad de elementos a tomar en cuenta por histograma) para posteriormente crear el histograma (ver Fig. 6); el vector de características se forma al concatenar los histogramas creados.

3. Metodología

En esta sección se explica el método propuesto basado en GE para la caracterización automática de imágenes de expresiones faciales. En la Fig. 7 se muestra el diagrama de dicha metodología.

Las entradas requeridas por GE son: el conjunto de imágenes de la base de datos JAFFE y un algoritmo clasificador (en este trabajo se empleó una Máquina de Vector Soporte). Inicialmente se crea una población de individuos (cromosomas) con una determinada dimensión y con valores numéricos enteros aleatorios creados a partir de una distribución uniforme. Por medio de la gramática BNF y el proceso de mapeo Depth-First, los individuos se transforman a su forma fenotípica y su calidad es evaluada por una función de aptitud.

Luego, los operadores de mutación, cruza y selecciones pertenecientes a DE son aplicados a la población para obtener nuevos individuos que puedan mejorar su desempeño en el problema. Este proceso se repite hasta que algún criterio de paro se cumpla (en este trabajo se utilizó un determinado número de generaciones).

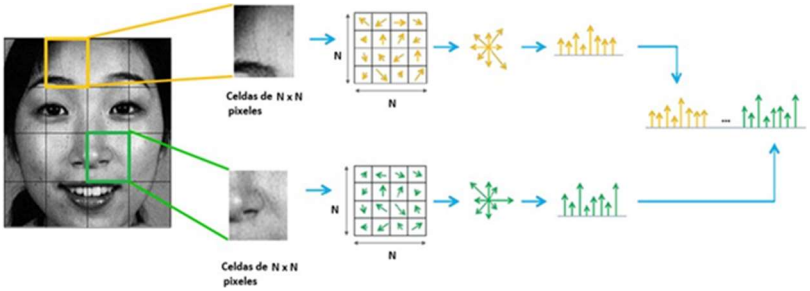


Fig. 6. Caracterización de una imagen con HOG.

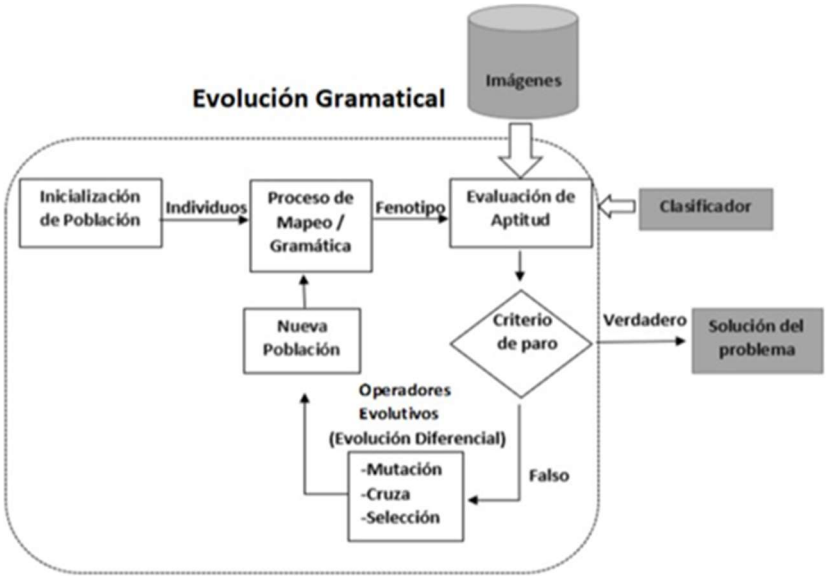


Fig. 7. Diagrama de la metodología propuesta [4].

Gramática BNF. Los descriptores generados por GE se forman por la combinación de un conjunto de operadores básicos agrupados en filtros pasa-alta, pasa-baja, direccionales, Max-Pooling, escala de grises y operadores aritméticos [4]. Estos operadores constituyen los símbolos terminales de la gramática BNF. En la Tabla 1 se muestra una descripción de estos operadores y sus correspondientes símbolos empleados en la gramática.

Los símbolos no terminales empleados en la gramática BNF fueron los siguientes:

N={*<Start>*,*<MP>*,*<Expr>*,*<Filter>*,*<Gau>*,*<Lap>*,*<GFB>*,*<Arith>*,*<Op>*, *<Terminal>* }

Las reglas de producción se muestran en la Tabla 2, las cuales son separadas por el símbolo "|". Por lo tanto, un símbolo no terminal puede tener una o varias reglas de producción, las cuales se enumeran para el proceso de mapeo. El símbolo inicial es *<Start>*.

Función de aptitud. Una vez que se tiene un fenotipo (descriptor generado por GE), se requiere evaluar su calidad mediante una función de aptitud. Para ello se aplica el

Tabla 1. Operadores básicos utilizados de la Gramática BNF.

Tipo de operador	Nombre	Símbolo terminal en la Gramática
Pasa-alta	Laplaciano	Lap
Pasa-alta	Laplaciano de la Gaussiana	LapG1, LapG2
Pasa-alta	Derivada de la Gaussiana	GauDX, GauDY
Pasa-baja	Gaussiana	Gau1, Gau2
Pasa-baja	Media	AverF
Pasa-baja	Mediana	MedianF
Direccional	Banco de Filtros de Gabor	GFB0, GFB45, GFB90, GFB135
Max-Pooling	Max Pooling	MP2, MP4, MP6, MP8, MP10
Histograma	Ecualización del Histograma	HEq
Operador aritmético	Valor Absoluto de una imagen	AbsV
Operador aritmético	Cuadrado de una imagen	Sqr
Operador aritmético	Raíz cuadrada de una imagen	Sqrt
Operador aritmético	Logaritmo base 2	Log
Operador aritmético	Multiplicar una imagen por 0.5	T0.5
Operador aritmético	Sumar 2 imágenes	ADD
Operador aritmético	Restar 2 imágenes	SUB
Operador aritmético	Resta absoluta de 2 imágenes	ASUB
Operador aritmético	Multiplicar 2 imágenes	MUL
Operador aritmético	Dividir 2 imágenes	DIV
Escala de grises	Obtener imagen a escala de grises	lg

descriptor a la base de datos JAFFE y se obtiene un nuevo conjunto de imágenes procesadas del mismo tamaño que la base de datos original.

Luego, para cada imagen procesada (del nuevo conjunto de imágenes) se obtiene un vector de características mediante HOG, el cual es etiquetado de acuerdo a la clase a la que pertenece.

Finalmente, el conjunto de vectores es clasificado por una Máquina de Vector Soporte (SVM) y el porcentaje de clasificación obtenido representa la aptitud del fenotipo. El valor de aptitud de los fenotipos más aptos se aproxima al 100 % por lo que a la función se le considera como una función de maximización.

Un ejemplo de la aplicación de un descriptor generado por GE a una imagen de la base de datos JAFFE se muestra en la Fig. 8. El descriptor generado es "*lg , GauDY , lg , SUB , Gau2 , MP6*" que mediante notación postfija se va aplicando paso por paso y de izquierda a derecha a la imagen de prueba, obteniendo una imagen procesada. A

Tabla 2. Reglas de producción utilizadas en la Gramática BNF [4].

Símbolo no terminal	Reglas de producción
<*Start*>	<*Expr*><*MP*> (0)
<*Expr*>	<*Expr*><*Expr*><*Op*> (0)| <*Expr*><*Filter*> (1) | <*Terminal*> (2)
<*Op*>	*ADD* (0) | *SUB* (1)| *ASUB* (2) | *MUL* (3)| *DIV* (4)
<*Filter*>	<*Gau*> (0) | <*Lap*> (1) | <*GFB*> (2) |<*Arith*> (3)
<*Gau*>	*Gau1* (0) | *Gau2* (1) | *GauDX* (2) | *GauDY* (3)
<*Lap*>	*LapG1* (0) | *LapG2* (1)| *Lap* (2)
<*GFB*>	*GFB0* (0) | *GFB45* (1) | *GFB90* (2)| *GFB135* (3)
<*Arith*>	*AverF* (0) | *MedianF* (1)| *HEq* (2)| *AbsV*(3) | *Sqr* (4)| *Sqrt* (5)| *Log* (6)| *T0.5* (7)
<*MP*>	*MP2* (0) | *MP4* (1)| *MP6* (2)| *MP8* (3)| *MP10* (4)
<*Terminal*>	*lg* (0)

la imagen resultante se le aplica HOG para obtener un vector de características. Este proceso se repite para cada una de las imágenes de la base de datos JAFFE.

4. Experimentación y resultados

En esta sección se muestra la experimentación realizada y los resultados obtenidos de la misma.

Base de datos. Japanese Female Facial Expresions (JAFFE) fue la base de datos utilizada en este trabajo [9]. Contiene 213 imágenes de rostros de 10 mujeres japonesas, agrupadas en 6 expresiones faciales básicas además de una expresión neutra, quedando distribuidas como sigue: 30 imágenes neutrales, 30 imágenes de enojo, 29 imágenes de disgusto, 32 imágenes de miedo, 31 imágenes de alegría, 31 imágenes de tristeza y 30 imágenes de sorpresa. Cada imagen tiene una dimensión de 256 × 256 pixeles.

Experimentación. A continuación, se muestran los valores de los parámetros utilizados en los diferentes algoritmos involucrados en la generación automática de descriptores, así como del algoritmo clasificador.

Población de individuos: El tamaño de la población fue de 20 cromosomas y cada uno de ellos estuvo formado por 20 códons con valores enteros en el rango de [0, 255]. Número de generaciones del proceso evolutivo = 80. Algoritmo HOG: Tamaño de celdas = 32 x 32 pixeles y tamaño de histogramas = 5 bins. Algoritmo clasificador (SVM): El algoritmo SVM se implementó utilizando la librería Waikato Environment for Knowledge Analysis (WEKA) empleando su configuración original (Kernel Lineal de grado 1, valor C =1, tolerancia de 0.001).

En el proceso de clasificación se utilizó validación cruzada con $K = 10$ pliegues para obtener un porcentaje de clasificación promedio como valor de aptitud de cada fenotipo generado por GE. Para probar la metodología propuesta y tener consistencia en los resultados obtenidos, se realizaron 35 experimentos de manera independiente. En cada

Tabla 3. Comparación de resultados del dataset JAFFE.

Autor	**Metodología**	**Porcentajes de Clasificación**
Eng. Et. al.[5]	HOG , SVM	88.89 %
Yang Et. al. [6]	Banco de Filtros de Gabor + Descriptor Local de Weber, KNN	92.77 %
Pitaloka Et. al. [7]	Deep Learning	97.06 %
Melaugh Et. al. [8]	Red Neuronal Convolucional	76.56 %
Nuestra propuesta	Evolución Gramatical+ HOG, SVM	97.78 %

experimento se generó inicialmente de manera aleatoria una población de cromosomas, la cual fue modificada a través del proceso evolutivo. En la última generación, se seleccionó al cromosoma más apto (con mayor porcentaje de clasificación), el cual fue el representante de ese experimento. Los parámetros de DE, HOG y SVM se mantuvieron sin cambios a través de todos los experimentos realizados.

Resultados. El porcentaje de clasificación de nuestra propuesta se presenta como el promedio de los representantes de los 35 experimentos realizados en forma independiente y se compara en la Tabla 3 con resultados del estado de arte.

En esta tabla se puede observar que nuestra metodología usando GE obtuvo resultados semejantes a los de Deep Learning [7]. La realización de un análisis estadístico para comparar el rendimiento de esta propuesta con respecto a las del estado de arte no fue factible debido a que cada metodología propone un diseño experimental diferente, sin embargo, se obtuvo información acerca de los resultados que se han obtenido en relación al reconocimiento de expresiones faciales empleando el dataset JAFFE.

Finalmente, en la Fig. 9 se muestra la evolución del mejor valor de aptitud calculado en cada generación para un experimento, utilizando los valores de parámetros reportados al inicio de esta sección.

5. Conclusiones

En este artículo se propuso una metodología basada en Evolución Gramatical e Histograma de Gradientes Orientados para la generación automática de descriptores aplicados al reconocimiento de imágenes de expresiones faciales utilizando la base de datos JAFFE. Evolución Gramatical utilizó una gramática en la forma Backus-Naur, Evolución Diferencial como motor de búsqueda y el proceso de mapeo Depth-First para generar descriptores sintácticamente correctos que se aplicaron a las imágenes de la base de datos. Posteriormente las imágenes procesadas fueron caracterizadas utilizando el Histograma de Gradientes Orientados y los vectores obtenidos se clasificaron

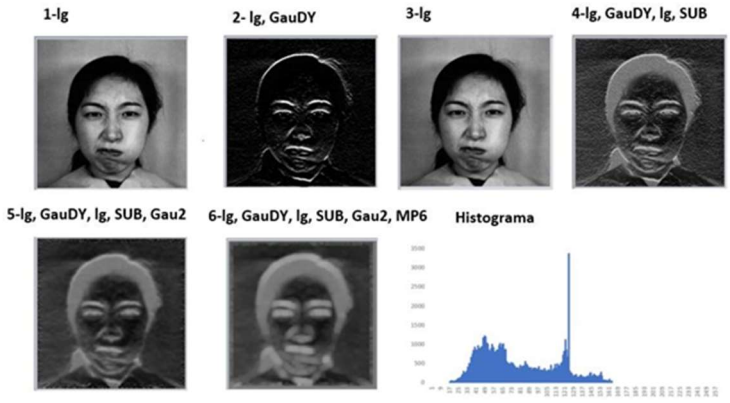


Fig. 8. Ejemplo del proceso de caracterización de una imagen.

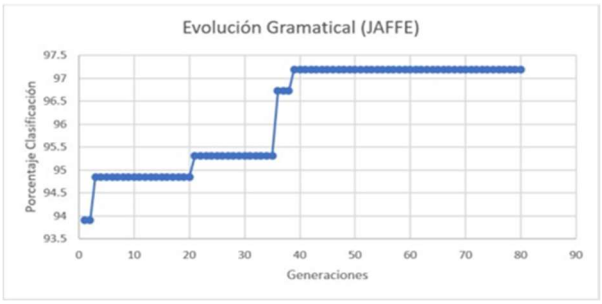


Fig. 9. Gráfica de evolución del mejor valor de aptitud por generación.

mediante una Máquina de Vector Soporte. Los resultados obtenidos mostraron que, para el caso de la base de datos JAFFE, la metodología propuesta presentó un desempeño comparable con metodologías reportadas en el estado de arte e incluso superó los resultados de algunas de ellas. Adicionalmente, esta metodología no requiere conocimiento a priori del problema a resolver ni la intervención de un experto.

Como trabajo a futuro, se desea aplicar la metodología a otras bases de datos del estado del arte, como es el caso de Extended Cohn-Kanade (CK+) o Karolinska Directed Emotional Faces (KDEF). Además, se realizará un análisis comparativo de diferentes algoritmos metaheurísticos que puedan ser usados como motor de búsqueda en Evolución Gramatical.

Agradecimientos. Los autores agradecen al Tecnológico Nacional de México el apoyo proporcionado. M. A. Torres Fonseca y A. Rojas Domínguez agradecen a CONACYT el apoyo mediante la beca para estudios de posgrado y la beca de investigación CATEDRAS-2598, respectivamente.

Referencias

1. Ekman, P. E., Davidson, R. J.: The nature of emotion: Fundamental questions. Oxford University Press, pp. 20-31 (1994)

2. Cornejo, J. Y. R., Pedrini, H.: Emotion recognition from occluded facial expressions using weber local descriptor. In: 25Th international conference on systems, signals and image processing , IEEE, pp. 1-5 (2018) doi: 10.1109/IWSSIP.2018.8439631
3. Kola, D. G. R., Samayamantula, S. K.: A novel approach for facial expression recognition using local binary pattern with adaptive window. Multimedia Tools and Applications, pp. 1-20 (2020) doi: 10.1007/s11042-020-09663-2
4. Calzada-Ledesma, V., Puga-Soberanes, H. J., Ornelas-Rodriguez, M., Rojas- Dominguez, A., Carpio-Valadez, J. M., Espinal, A., Sotelo-Figueroa, M. A.: Evolutionary design of problem-adapted image descriptors for texture classification. IEEE, vol 6, pp. 40450-40462 (2018) doi: 10.1109/ACCESS.2018.2858660
5. Eng, S. K., Ali, H., Cheah, A. Y., Chong, Y. F.: Facial expression recognition in JAFFE and KDEF datasets using histogram of oriented gradients and support vector machine. In: IOP Conference series: Materials science and engineering, IOP Publishing, vol. 705 (2019) doi: 10.1088/1757-899X/705/1/012031
6. Yang, J., Li, M., Zhang, L., Han, S., Wang, X., Wang, J.: Face expression recognition using gabor features and a novel Weber local descriptor. In: Chinese Conference on Biometric Recognition, Springer, Cham, pp. 265- 274 (2018) doi: 10.1007/978-3-319-97909-0_29
7. Pitaloka, D. A., Wulandari, A., Basaruddin, T., Liliana, D. Y.: Enhancing CNN with preprocessing stage in automatic emotion recognition. Procedia computer science, vol. 116, pp. 523–529 (2017) doi: 10.1016/j.procs.2017.10.038
8. Melaugh, R., Siddique, N., Coleman, S., Yogarajah, P.: Facial expression recognition on partial facial sections. In: 11th International Symposium on Image and Signal Processing and Analysis, IEEE, pp. 193-197 (2019)
9. Lyons, M. Kamachi, M. Gyoba J.: Japanese female facial expression database of digital images. (1997) https://zenodo.org/record/3451524#.YHCk3-gzZPY
10. Moore, J. H., Sipper, M.: Grammatical evolution strategies for bioinformatics and systems genomics. Handbook of Grammatical Evolution, Springer, pp. 395-405 (2018) doi:10.10 07/978-3-319-78717-6_16
11. Quiroz-Ramírez, O., Espinal, A., Ornelas-Rodríguez, M., Rojas-Domínguez, A., Sánchez, D., Puga-Soberanes, H., Carpio, M., Mancilla Espinoza, L. Ortíz-López, J.: Partially-connected artificial neural networks developed by grammatical evolution for pattern recognition problems. Fuzzy Logic Augmentation of Neural and Optimization Algorithms: Theoretical Aspects and Real Applications, Studies in Computational Intelligence Springer, vol. 749, pp. 99-112 (2018) doi: 10.1007/978-3-319-71008-2_9
12. Simon, D.: Evolutionary optimization algorithms. John Wiley & Sons, pp. 293–305 (2013)
13. Manas, K. B.: Computer vision and image processing. Taylor and Francis Group, pp.198–200 (2019)

ISSN 1870-4069

Implementación de un algoritmo de aprendizaje por refuerzo en ambientes virtuales, orientado a robótica móvil

Sergio Isahí Garrido Castañeda[1], Gabriel Sepúlveda Cervantes[1], Eduardo Vega Alvarado[1], Edgar Alfredo Portilla Flores[1], Miguel Ángel Garrido Castañeda[2]

[1] Instituto Politécnico Nacional, Centro de Innovación y Desarrollo Tecnológico en Cómputo, México

[2] Instituto Politécnico Nacional, Unidad Profesional Interdisciplinaria en Tecnologías Avanzadas, México

sgarridoc2000@alumno.ipn.mx, {gsepulvedac, evega, aportilla}ipn.mx, kurogarri@gmail.com

Resumen. En este trabajo se presenta un sistema de entrenamiento para robots móviles utilizando aprendizaje de máquina (*Machine Learning*) implementado en un ambiente virtual. En los últimos años, el uso de técnicas de inteligencia artificial para resolver problemas del mundo real ha ido en aumento. Las aplicaciones de estas herramientas abarcan una amplia gama de disciplinas, destacando la solución de problemas de clasificación y/u optimización. En el ámbito de la robótica móvil, esto ha abierto el horizonte a nuevos paradigmas para el control de las variables que rigen estos sistemas, siendo el aprendizaje por refuerzo uno de los modelos de aprendizaje de máquina que puede adaptarse mejor a este fin. Por otro lado, el desarrollo de ambientes virtuales en donde se modelan las características físicas de un agente sujeto a un proceso de aprendizaje permite tanto el entrenamiento del agente como la transferencia del entrenamiento al mundo real, teniendo como beneficios la simulación del comportamiento del agente en diversos ambientes, el entrenamiento evitando desgaste físico, y la realización de pruebas sin riesgos relacionados a daños del hardware o incluso a los seres vivos. Asimismo, el desempeño de los equipos de cómputo actuales permite realizar el entrenamiento de manera eficiente en periodos de tiempo cortos. Los resultados obtenidos muestran la flexibilidad del esquema propuesto, el cual permite repetir exitosamente el aprendizaje modificando las condiciones y las actividades a realizar por el agente.

Palabras clave: Aprendizaje por refuerzo, ambiente virtual, agente, robótica móvil.

Implementation of a Reinforcement Learning Algorithm in Virtual Environments, Oriented to Mobile Robotics

Abstract. In this work, a training system for mobile robots using machine learning implemented on a virtual environment is presented. In recent years, the

use of AI techniques for solving real world problems has increased. The application of these tools covers a wide variety of areas, highlighting the solution of classification and/or optimization problems. In the area of mobile robotics, this development has opened new paradigms to control the variables that govern these systems, and the reinforcement learning is one of the models of machine learning that can be best adapted for this task. Additionally, the development of virtual environments where the physical characteristics of an agent in a learning process are modeled, allows the training of the agent and, simultaneously, the transference of the training to the real world, having as benefits the simulation of the agent behavior in diverse environments, the training without physical wear of the system, and the test of the system avoiding damage to its hardware or even to the living beings related to its operation. The training can be carried out efficiently in a short time period because of the performance of current computing equipment. The results of this development show its flexibility to successfully repeat the learning with the capability to modify the conditions and the activities executed by the agent.

Keywords: Reinforcement learning, virtual environment, agent, mobile robotics.

1. Introducción

El aprendizaje automático o de máquina (*Machine Learning*) describe las técnicas con las cuales un sistema aprende y generaliza su comportamiento con base en datos. Puede clasificarse en tres categorías: supervisado, no supervisado y por refuerzo. En la primera se aprende mediante el uso de un supervisor externo que ya cuenta con el conocimiento de lo que se desea hacer, mientras que en la segunda no hay tal apoyo. Por su parte, en el aprendizaje por refuerzo existen dos elementos principales: un agente y un ambiente.

Partiendo de la premisa de que el agente está inmerso en el ambiente y tiene la capacidad de comprender su entorno, el aprendizaje se da mediante un sistema que cuantifica que tan bien el agente realiza su función; el aprender a caminar, percatarse que el fuego quema, etc., son ejemplos de esto. Así, se pueden modelar fielmente sistemas en los que se requiera realizar un entrenamiento que, después de un número determinado de eventos**,** traiga como resultado un aprendizaje.

Fue a finales de la década de los 80s que el aprendizaje por refuerzo comenzó a atraer el interés de la comunidad científica [1]. Sus aplicaciones en el ámbito de la investigación son variadas y van desde sistemas capaces de obtener puntajes jamás alcanzados por humanos en videojuegos [2, 3], algoritmos de sugerencia de publicidad con base al comportamiento de usuarios en la web [4], e incluso la creación de una inteligencia artificial capaz de vencer al campeón mundial de Go después de solamente unas pocas horas de entrenamiento [5]. Si bien estos ejemplos pudieran no parecer de alto impacto, demuestran los alcances de esta modalidad de la mano de unidades y herramientas computacionales cada vez más potentes.

Dentro de los sistemas de aprendizaje por refuerzo que cruzan la barrera del mundo digital al mundo real, se han hecho avances en el ámbito de la robótica, donde se han implementado dos variantes. En la primera, el ambiente del aprendizaje es el mundo real: cada uno de los eventos en los que el agente realiza una decisión que, con base a la política establecida desencadena una acción, se lleva a cabo en la realidad. Se han

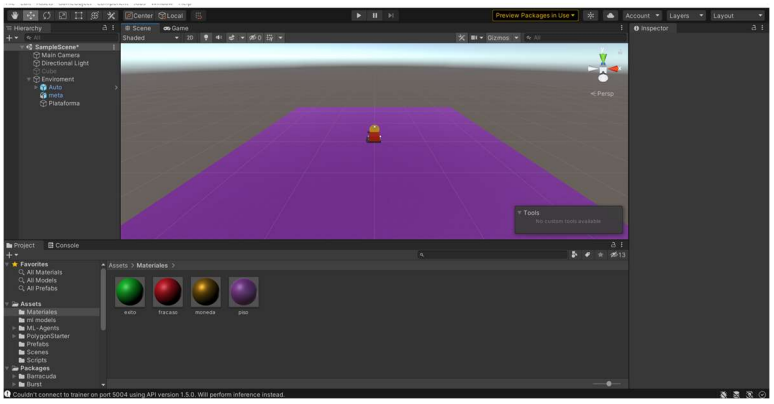


Fig. 1. Interfaz gráfica de Unity 3D.

realizado estudios de esta variante en sistemas robóticos donde el agente aprende a caminar y se observa su progresión al finalizar cada evento de aprendizaje [6], en sistemas orientados a la capacidad de adaptación a cambios de su estructura de locomoción [7], o sistemas de robótica de manipuladores para el lanzamiento de objetos considerando su estructura física y peso [8].

La segunda variante modela computacionalmente el mundo real en el que el agente está inmerso, para simular cada una de las iteraciones del algoritmo de aprendizaje y enviar la información del entrenamiento al agente físico, sin que éste deba realizar las acciones en donde la recompensa no fue maximizada. Ejemplo de esto es el aprendizaje por refuerzo orientado a sistemas, en robots terrestres o aéreos no comunicados entre sí, capaces de esquivar obstáculos en su entorno [9], y su evolución a sistemas capaces de detectar seres vivos y predecir sus movimientos, para una adecuada convivencia entre ellos y el ser humano en ambientes diversos [10, 11].

Existen diversos enfoques para analizar sistemas robóticos colaborativos, con al menos dos individuos entrenados para cumplir una tarea de forma cooperativa. En uno de ellos, los agentes trabajan de manera síncrona para compenetrarse y seguir una estrategia que maximice la recompensa obtenida, cumpliendo la tarea para la que fueron entrenados. En contraste, existe el modelo en el que varios agentes, a partir de estrategias individuales simples y sin alguna sincronía entre sí, pueden unificar su información y comportamiento individual con el fin de formular una política de alto nivel que permita la detección automática de cambios en el entorno, y a su vez una acción de respuesta para adaptarse efectivamente a la situación actual y a posibles cambios futuros.

Esto ha servido para plantear entornos donde un conjunto de robots, separados en equipos, compiten en la realización de una tarea. En [12] se propone que, con base en una serie de estrategias analizadas por cada uno de los individuos de un equipo buscando siempre la maximización de la recompensa, se agrupen con el fin de colaborar efectivamente en contra de sus adversarios.

Un caso particular del aprendizaje por refuerzo es en el que varios agentes interactúan en el mismo ambiente (*Multi-Agent Reinforcement Learning*) siguiendo la misma pauta de buscar maximizar la recompensa en todo momento, independientemente de la política a seguir en el proceso de entrenamiento [13]. En este

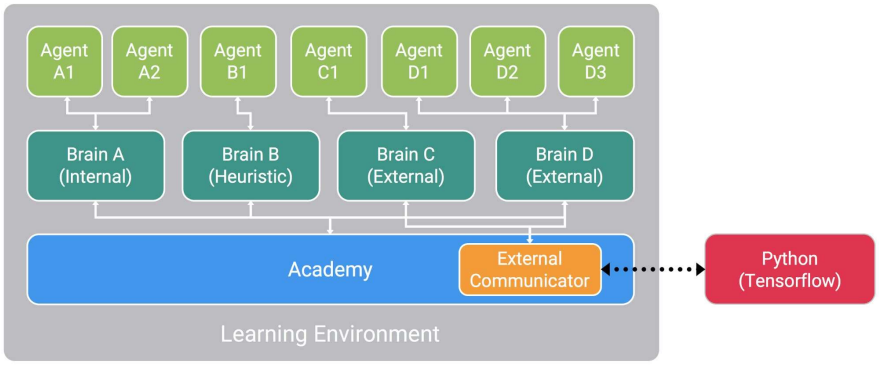


Fig. 2. Flujo de trabajo en un entrenamiento con ML-Agents.

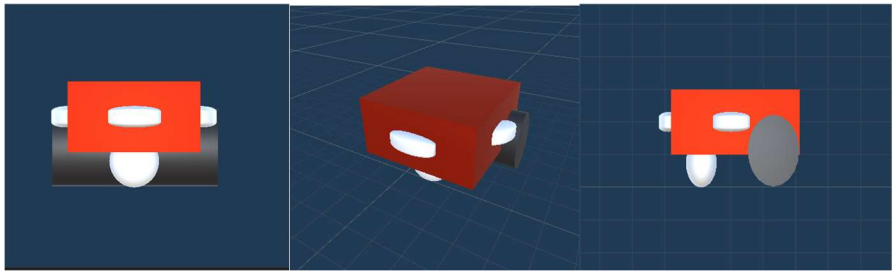


Fig. 3. Vista frontal, de perspectiva y lateral del modelo 3D del agente.

trabajo se presenta una implementación de aprendizaje por refuerzo enfocado a un agente con las características de un vehículo terrestre con ruedas de tracción diferencial, la metodología para la realización de su entrenamiento y a su vez, las acciones que puede realizar para su entrenamiento en un ambiente virtual que permite simular el sistema en su totalidad.

La organización del documento es la siguiente: en la Sección 2 se describen de manera general los parámetros del aprendizaje por refuerzo, su terminología y notación; en la Sección 3 se muestran las herramientas computacionales utilizadas para la implementación del entrenamiento del agente, del ambiente virtual y de la integración de ambos; en la Sección 4 se presenta el caso de estudio a resolver y el desarrollo de la solución; en la Sección 5 se analizan los resultados obtenidos y su validación; finalmente, la Sección 6 aborda las conclusiones y los trabajos a futuro.

2. Parámetros del aprendizaje por refuerzo

Como se mencionó anteriormente, en el aprendizaje por refuerzo se plantea cuál debe ser el comportamiento de un agente dentro de un ambiente, con el objetivo de maximizar una recompensa. El agente, sin importar su naturaleza, se enfrenta al problema de decidir las acciones a realizar, por lo que se estudian las repercusiones de dichas acciones y se proporciona al agente el aprendizaje para lograr optimizarlas. Para

la implementación del aprendizaje por refuerzo se proponen los parámetros de modelado que se detallan a continuación.

2.1. Procesos de decisión de Markov

Los procesos de decisión de Markov (MDP) son una abstracción de los componentes de la toma de decisiones. Partiendo de la existencia de un agente interactuando en un ambiente, las interacciones se dan en forma de acciones derivadas de la toma de decisiones del agente hechas de manera secuencial en cada paso de tiempo. El agente obtiene una representación del estado del ambiente y, de acuerdo con ella, selecciona y realiza una acción dentro de un conjunto finito.

La acción cambia el estado del ambiente en el siguiente paso de tiempo y el agente recibe una recompensa de acuerdo con que tan buena fue la decisión tomada. Una trayectoria es la secuencia de la toma de decisión para pasar al estado siguiente con la recompensa correspondiente al estado anterior. El agente trata de maximizar tanto la recompensa del estado siguiente como la recompensa acumulada a lo largo de la trayectoria.

En un MDP se denomina S al conjunto finito de estados, mientras que A es el conjunto finito de acciones que el agente puede seleccionar, que traen como consecuencia que obtenga una recompensa perteneciente al conjunto R. El proceso anterior se ejecuta en un estado de tiempo del conjunto $t = 0,1,2, ...$, en el cual el agente recibe al estado $S_t \in S$ y ejecuta una acción $A_t \in A$, obteniendo así el par (S_t, A_t) y la recompensa $R_{t+1} \in R$; posteriormente se pasa al tiempo $t + 1$, donde el ambiente pasa al estado $S_{t+1} \in S$. Lo anterior se resume en la expresión (1):

$$f(S_t, A_t) = R_{t+1}. \tag{1}$$

2.2. Retorno

Dado que el objetivo de un agente es maximizar las recompensas acumulativas, todas sus decisiones se toman orientadas a este fin. Las recompensas se calculan mediante el retorno esperado en un plazo de tiempo determinado, donde se define a G como el retorno en el momento t, tal como se expresa en la Ec. (2):

$$G_t = R_{t+1} + R_{t+2} + R_{t+3} + R_T, \tag{2}$$

Un episodio es una porción de la trayectoria, siempre y cuando tenga un paso de tiempo final, que termina en el tiempo T y el siguiente estado pasa a ser uno con condiciones iguales a las de $t = 0$ o algún otro dentro de un conjunto de estados posibles. En particular, el episodio siguiente es independiente del estado final del episodio que le antecedió.

Las tareas que no tienen definido un tiempo final T son tareas continuas en las que $T = \infty$, y en consecuencia $G = \infty$. Para solucionar este problema es necesario hacer que el agente considere el retorno esperado, dándole un peso mayor a las recompensas inmediatas obtenidas en cada paso de tiempo, pero con un descuento.

La tasa de descuento γ puede tomar un valor entre 0 y 1, y es el mecanismo mediante el cual se descontarán las recompensas futuras y a su vez se determinará el valor presente de dichas recompensas, como se indica en la Ec. (3). Se puede observar que,

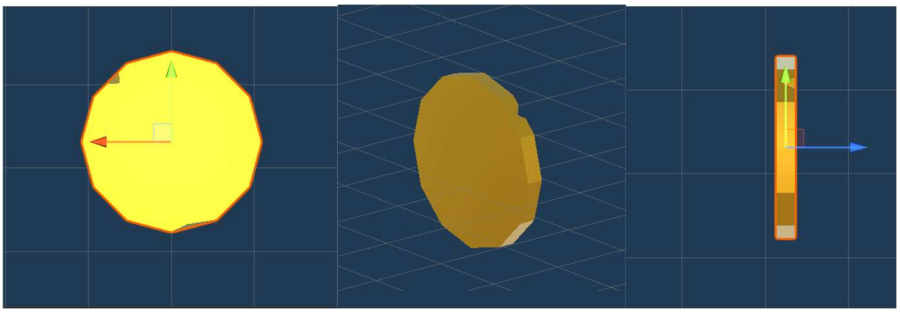


Fig. 4. Vista frontal, de perspectiva y lateral del modelo 3D del agente.

Fig. 5. Ambiente virtual para el entrenamiento.

al introducir la tasa de descuento, el agente pasa en automático a considerar las recompensas inmediatas para decidir las acciones que realizará, mientras que las recompensas futuras tienden a tener un menor peso mientras mayor sea el paso del tiempo, lo cual se expresa en la Ec. (4):

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \\ = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}, \tag{3}$$

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \gamma^3 R_{t+4} + \\ = R_{t+1} + \gamma(R_{t+2} + \gamma R_{t+3} + \gamma^2 R_{t+4} +) \\ = R_{t+1} + \gamma G_{t+1}\,. \tag{4}$$

3. Herramientas

Existen diversas plataformas para llevar el modelado de un sistema de aprendizaje automático al plano computacional. Una de ellas es TensorFlow, la cual cuenta con herramientas y bibliotecas que permiten crear y desplegar fácilmente aplicaciones basadas en diversos modelos de aprendizaje [14]; lo anterior, mediante la utilización de un solo grafo de flujo de datos que representa todos los cálculos y el estado del algoritmo de aprendizaje automático, lo que incluye también a las operaciones matemáticas individuales, los parámetros del sistema y sus reglas de actualización [15].

Unity 3D (*Unity Technologies*), cuya interfaz se muestra en la Figura 1, es una herramienta capaz de crear entornos virtuales que reflejen las condiciones exactas del ambiente de entrenamiento, por medio de un motor de desarrollo de plataformas interactivas.

Para la simulación de fenómenos físicos cuenta con los motores Nvidia PhysX o Havock Physics, además de tener la ventaja de ser flexible a la adecuación de motores de otros grupos de desarrolladores, tales como Bullets y MuJoCo [16]. Para el desarrollo de IA existe la interfaz de programación de aplicaciones ML-Agents, para utilizar entornos virtuales realizados tanto en Unity 3D como en ambientes de entrenamiento para sistemas de aprendizaje automático desarrollados previamente [17].

4. Caso de estudio

Un problema común en robótica móvil es el seguimiento o búsqueda de objetos. Esta tarea en primera instancia puede parecer simple, pero su trasfondo tiene una cantidad considerable de parámetros que se deben de tomar en cuenta si se busca modelar su comportamiento con técnicas convencionales, ya que por lo regular estos enfoques no generalizan su comportamiento en todos los casos con base a sus entradas.

4.1. Descripción

En este trabajo se busca implementar un algoritmo basado en aprendizaje por refuerzo mediante PPO (*Proximal Policity Optimization*) para solucionar el caso de estudio, haciendo uso de un ambiente virtual en el que se modelen los componentes requeridos para realizar el aprendizaje de la toma de decisiones del agente. El agente por entrenar es el modelo de un vehículo terrestre de tracción diferencial, el cual tiene como tarea alcanzar una meta en forma del modelo de una moneda.

4.2. Implementación

Para la implementar la solución al caso de estudio seleccionado se hizo uso de Unity 3D con el módulo ML-Agents incorporado, el cual a su vez utiliza en sus capas internas la biblioteca TensorFlow. En Unity 3D se modelan los componentes del ambiente y mediante código en C# se crean las instancias necesarias tales como observaciones, acciones, asignaciones de recompensas y características de los episodios de cada una de las iteraciones del entrenamiento. ML-Agents crea una comunicación entre el ambiente virtual con las configuraciones de hiper-parámetros y TensorFlow, que se encarga de la parte de entrenamiento y aprendizaje.

Así, el ambiente proporciona los datos para el entrenamiento y ML-Agents hace un puente de comunicación, adecuando dichos datos para que TensorFlow los utilice en sus cálculos, cuyo resultado se regresa al ambiente para la adquisición de nuevos datos (ver Figura 2).

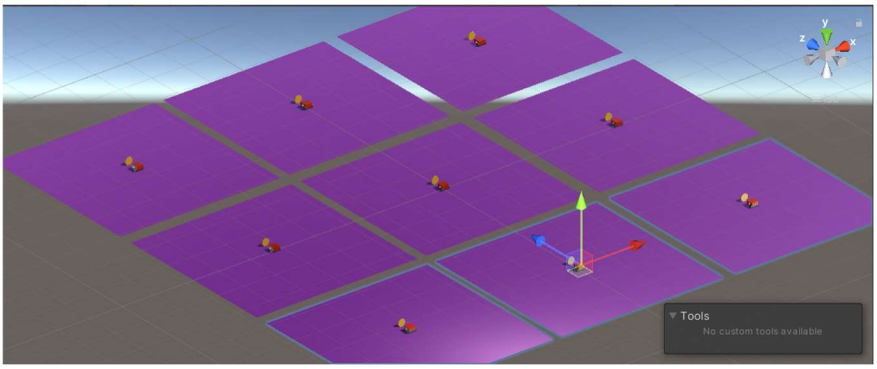


Fig. 6. Configuración para la paralelización del entrenamiento.

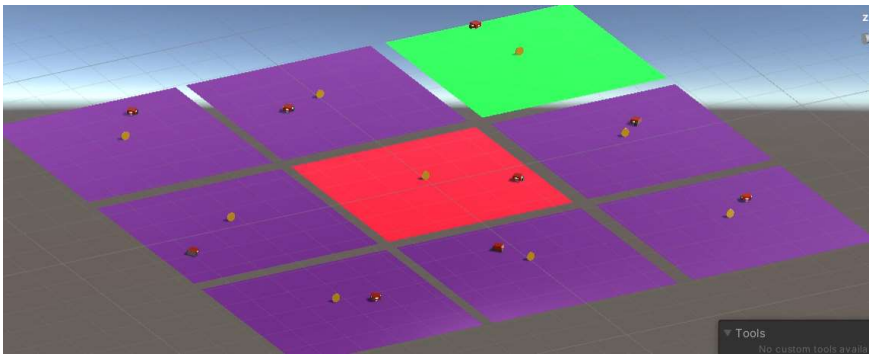


Fig. 7. Entrenamiento en ejecución.

```
2021-04-09 08:45:01 INFO [stats.py:180] agent_config. Step: 12000. Time Elapsed: 27.074 s. Mean Reward: -0.519. Std of Reward: 0.588. Training.
2021-04-09 08:45:45 INFO [stats.py:180] agent_config. Step: 24000. Time Elapsed: 71.402 s. Mean Reward: -0.421. Std of Reward: 0.682. Training.
2021-04-09 08:46:29 INFO [stats.py:180] agent_config. Step: 36000. Time Elapsed: 115.077 s. Mean Reward: -0.059. Std of Reward: 0.900. Training.
2021-04-09 08:47:13 INFO [stats.py:180] agent_config. Step: 48000. Time Elapsed: 158.518 s. Mean Reward: 0.490. Std of Reward: 1.003. Training.
2021-04-09 08:47:57 INFO [stats.py:180] agent_config. Step: 60000. Time Elapsed: 202.715 s. Mean Reward: 0.768. Std of Reward: 0.762. Training.
2021-04-09 08:48:41 INFO [stats.py:180] agent_config. Step: 72000. Time Elapsed: 246.966 s. Mean Reward: 0.926. Std of Reward: 0.516. Training.
2021-04-09 08:49:26 INFO [stats.py:180] agent_config. Step: 84000. Time Elapsed: 291.547 s. Mean Reward: 0.934. Std of Reward: 0.350. Training.
2021-04-09 08:50:11 INFO [stats.py:180] agent_config. Step: 96000. Time Elapsed: 336.699 s. Mean Reward: 0.944. Std of Reward: 0.222. Training.
2021-04-09 08:50:57 INFO [stats.py:180] agent_config. Step: 108000. Time Elapsed: 382.522 s. Mean Reward: 0.936. Std of Reward: 0.174. Training.
2021-04-09 08:51:43 INFO [stats.py:180] agent_config. Step: 120000. Time Elapsed: 428.809 s. Mean Reward: 0.945. Std of Reward: 0.076. Training.
2021-04-09 08:52:30 INFO [stats.py:180] agent_config. Step: 132000. Time Elapsed: 475.757 s. Mean Reward: 0.942. Std of Reward: 0.108. Training.
2021-04-09 08:53:18 INFO [stats.py:180] agent_config. Step: 144000. Time Elapsed: 523.795 s. Mean Reward: 0.939. Std of Reward: 0.101. Training.
2021-04-09 08:54:05 INFO [stats.py:180] agent_config. Step: 156000. Time Elapsed: 571.414 s. Mean Reward: 0.944. Std of Reward: 0.081. Training.
2021-04-09 08:54:53 INFO [stats.py:180] agent_config. Step: 168000. Time Elapsed: 619.096 s. Mean Reward: 0.946. Std of Reward: 0.080. Training.
2021-04-09 08:55:42 INFO [stats.py:180] agent_config. Step: 180000. Time Elapsed: 668.234 s. Mean Reward: 0.949. Std of Reward: 0.055. Training.
2021-04-09 08:56:31 INFO [stats.py:180] agent_config. Step: 192000. Time Elapsed: 716.577 s. Mean Reward: 0.952. Std of Reward: 0.028. Training.
2021-04-09 08:57:19 INFO [stats.py:180] agent_config. Step: 204000. Time Elapsed: 765.226 s. Mean Reward: 0.949. Std of Reward: 0.100. Training.
2021-04-09 08:58:08 INFO [stats.py:180] agent_config. Step: 216000. Time Elapsed: 814.469 s. Mean Reward: 0.901. Std of Reward: 0.319. Training.
2021-04-09 08:58:58 INFO [stats.py:180] agent_config. Step: 228000. Time Elapsed: 863.884 s. Mean Reward: 0.934. Std of Reward: 0.206. Training.
2021-04-09 08:59:47 INFO [stats.py:180] agent_config. Step: 240000. Time Elapsed: 913.316 s. Mean Reward: 0.951. Std of Reward: 0.108. Training.
2021-04-09 09:00:38 INFO [stats.py:180] agent_config. Step: 252000. Time Elapsed: 963.661 s. Mean Reward: 0.954. Std of Reward: 0.064. Training.
2021-04-09 09:01:28 INFO [stats.py:180] agent_config. Step: 264000. Time Elapsed: 1013.534 s. Mean Reward: 0.958. Std of Reward: 0.018. Training.
2021-04-09 09:02:18 INFO [stats.py:180] agent_config. Step: 276000. Time Elapsed: 1063.522 s. Mean Reward: 0.958. Std of Reward: 0.044. Training.
```

Fig. 8. Información del entrenamiento.

4.3. Ambiente virtual

El ambiente se compone de una sola escena, la cual es el entorno del modelo de aprendizaje. Para el diseño de este entorno se utilizaron tres componentes: una plataforma que delimita el espacio del ambiente, un vehículo con ruedas (ver Figura 3) que fungirá como el agente, y una moneda que será el objetivo por alcanzar. El agente

tiene la forma de un robot móvil de tracción diferencial; en adición al diseño se consideró un componente *RigidBody*, el cual permite que se le apliquen fuerzas, y un componente *BoxCollider*, con el cual es posible detectar si colisionó con otro objeto.

Referente al diseño de la meta, ésta tiene forma de moneda (ver Figura 4), y al igual que el modelo del agente, cuenta con los componentes *RigidBody* y *BoxCollider*.

Se utilizaron cinco parámetros de observación y dos acciones como componentes para la implementación del algoritmo PPO. Las cinco observaciones se relacionan con la distancia del agente a la meta y su posición sobre el plano *XZ*, mientras que las acciones corresponden a la transformación de la posición del agente sobre ese mismo plano. A su vez, en cada episodio del entrenamiento el agente tendrá una posición aleatoria sobre el plano, mientras que la meta tendrá una posición fija en el centro de la plataforma. En la Figura 5 se muestra el diseño final del ambiente de entrenamiento.

5. Recompensas

Para este entrenamiento se estableció que el agente obtiene una recompensa igual a 1 cuando logra hacer contacto con la meta; adicionalmente, con esta acción se termina el episodio en ejecución, mientras que en cada estado de tiempo en el que el agente realiza un movimiento sobre el plano, si la distancia a la meta es menor que en el estado anterior, se le premia con una recompensa de 0.01. Con respecto a las recompensas negativas, estas se obtienen si la distancia del agente a la meta es mayor que en el estado anterior, teniendo un valor de -0.01. El episodio se considera como fracaso cuando el agente abandona la plataforma, y finaliza para iniciar el siguiente.

6. Entrenamiento

Con el fin de llevar a cabo el entrenamiento en un tiempo menor, se colocaron en la escena nueve ambientes con las mismas características y parámetros tal como se muestra en la Figura 6; esto se debe a que cada uno de los episodios que son llevados a cabo durante el entrenamiento aporta al resultado final.

Una vez comenzado el entrenamiento se tienen indicadores visuales sobre la plataforma, que informan si en el episodio anterior el agente alcanzó la meta o salió de la plataforma. Un episodio en curso se muestra en color morado y un episodio exitoso se marca con color verde, mientras que un fracaso se indica con color rojo, como se observa en la Figura 7.

Durante el entrenamiento es posible visualizar por consola (ver Figura 8) el tiempo transcurrido y la ganancia promedio obtenida en cada conjunto de episodios realizados. Dado que la recompensa acumulada máxima es aproximadamente 1, se tiene que el entrenamiento se está desarrollando correctamente si la recompensa promedio tiende a esa cifra con el paso de los conjuntos de episodios.

7. Resultados

Para efectos de prueba del sistema propuesto se ejecutaron 400,000 episodios; de las observaciones realizadas se aprecia que con el entrenamiento el agente se comportó

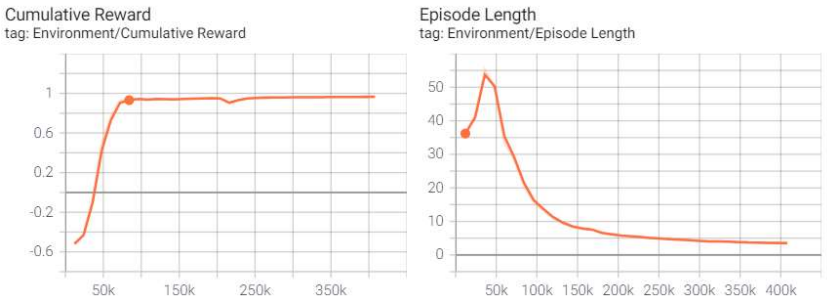


Fig. 9. Gráficas de recompensa acumulada y de longitud de episodio.

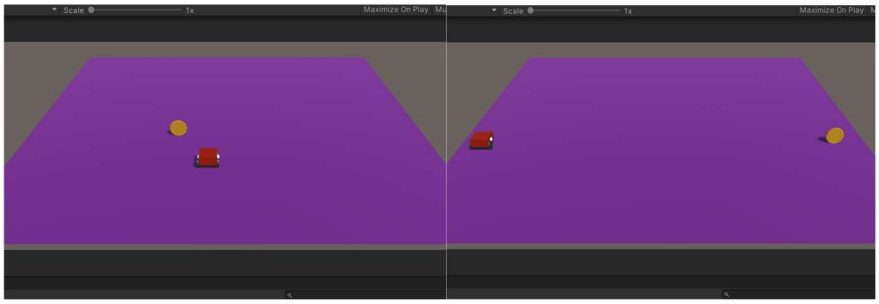


Fig. 10. Posiciones aleatorias del agente y la meta.

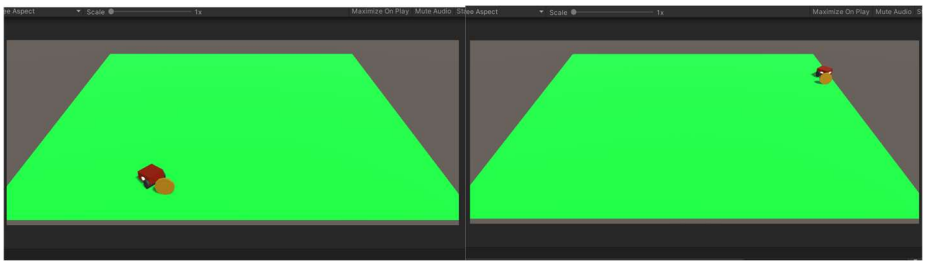


Fig. 11. Validación del entrenamiento con posición de meta aleatoria.

adecuadamente en el ambiente en el que su posición inicial es aleatoria y la posición de la meta a alcanzar es fija. De ahí se infiere que el agente aprendió correctamente ya que, en los casos de prueba, alcanzó el estado en el que obtiene una recompensa máxima.

Mediante la herramienta TensorBoard se obtuvieron las gráficas que se muestran en la Figura 9:

- Ganancia acumulada durante cada uno de los episodios, la cual debe de tender a uno en el caso de que el agente realice bien su labor.
- Longitud de episodio, la cual debe indicar que conforme avanzan los episodios de entrenamiento en los que la recompensa promedio comienza a tender a 1, la longitud debe ir en decremento.

Para la validación del entrenamiento se realizó el cambio de la posición de la moneda, para de determinar si el agente generaliza bien los datos obtenidos de las

observaciones en estados que no se encontraban presentes durante el entrenamiento, por lo que ahora tanto la posición del agente como la de la meta son completamente aleatorias (ver Figura 10).

Como se observa en la Figura 11, en este caso el agente también realiza correctamente la labor para la cual fue entrenado.

8. Conclusiones y trabajo a futuro

Cada modelo de aprendizaje automático cuenta con características que lo hacen adecuado para la solución de grupos de problemas, ya sean de clasificación u optimización, existiendo diversas herramientas para su desarrollo e implementación. En el caso del aprendizaje por refuerzo es necesario modelar un ambiente con una cantidad finita de características, donde el uso de motores de videojuegos como Unity 3D proporciona funcionalidades que flexibilizan y facilitan las tareas de diseño y establecimiento de características, permitiendo la creación de ambientes que pueden ser utilizados para el entrenamiento de agentes que busquen la maximización de una recompensa.

Por otro lado, bibliotecas como Tensor Flow hacen factible la tarea de programar y establecer los hiper-parámetros tanto del ambiente como de los agentes, con el fin de realizar su entrenamiento. Gracias a la unión de estos dos entes es posible realizar de forma sumamente gráfica la implementación de nuevas ideas de modelos de aprendizaje.

Como trabajo a futuro, se propone realizar la implementación del algoritmo de aprendizaje por refuerzo en un agente en el mundo real, en forma de un robot móvil con características similares a las del ambiente virtual, contando con sensores de proximidad y conectividad WiFi para el envío y recepción de datos, con el fin de que el entrenamiento en el mundo virtual se vea reflejado directamente en el mundo real.

De igual manera, se pretende ampliar dicho modelo para incluir características relacionadas con aspectos tales como la cinemática y dinámica del móvil.

Así mismo, se contempla la implementación de sistemas multi-agente que compartan el mismo ambiente virtual, con el fin de generar políticas que permitan el aprendizaje de los agentes para que puedan trabajar en conjunto en la realización de tareas específicas, en un esquema de trabajo colaborativo.

Referencias

1. Kaelbling, L., Littman, M., Moore, A.: Reinforcement learning: A survey. Journal of Artificial Intelligence Research, vol. 4 (1996) doi: 10.1613/jair.301
2. Machado, M., Bellmare, M., Talvitie, E., Veness, J.: Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents. Journal of Artificial Itelligence Research, vol. 61 (2017) doi: 10.1613/jair.5699
3. Mnlh, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I.: Playing atari with deep reinforcement learning. Deepmind Technologies (2013) doi: 10.48550/arXiv.1312.5602
4. Li, L., Chu, W., Langford, J., Schapire, R.: A contextual-bandit approach to personalized news article recommendation. Yahoo!, Labs- Dept of Computer Science, Princeton University, pp. 661–670 (2012) doi: 10.1145/1772690.1772758

5. Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Juang, A.: Mastering the game of Go without human knowledge. Nature 550 (2017) doi:10.1038/nature24270
6. Haarjona, T., Ha, S., Zhou, A., Tan, J., Tuker, G.: Learning to walk via deep reinforcement Learning. google brain, Berkeley Artificial Intelligence Research (2019) doi: 10.48550/arXiv.1812.11103
7. Chen, Y., Liu, M., Everett, M., How, J.: Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning. In: Proceedings of IEEE International Conference on Robotics and Automation (2017) doi: 10.1109/ICRA.2017.7989037
8. Ha, S., Kim, J., Yamaane K.: Automated deep reinforcement learning environment for hardware of a modular legged robot. In Proceedings of 15th International Conference on Ubiquitous Robots (UR), pp. 348–354 (2018) doi:10.1109/URAI.2018.8442201
9. Zen, A., Song, S., Lee, J., Rodriguez, A.: Tossingbot learning to throw arbitrary objects with residual physics. IEEE Transactions on Robotics, (2020) doi: 10.1109/TRO.2020.2988642
10. Chen, M., Everett, M., Liu, J.: How socially aware motion planning with deep reinforcement learning. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (2017) doi: 10.1109/IROS.2017.8202312
11. Everett, M., Chen, Y., How, J.: Collision avoidance in pedestrian-rich environments with deep reinforcement learning. In IEEE Access, vol. 9, pp. 10357–10377 (2020) doi:10.1109/ACCESS.2021.3050338
12. Hoang, T., Xiao, Y., Sivaakumar, K., Howl, J. P.: Near-optimal adversarial policy switching for decentralized asynchronous multi-agent systems. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 6373–6380 (2018) doi:10.1109/ICRA.2018.8460485
13. Busoniu, L., Babuska, R., de Schutter B.: Multi-agent reinforcement learning: An overview. In: Srinivasan D., Jain L.C. Innovations in Multi-Agent Systems and Applications – 1, Studies in Computational Intelligence, Springer, vol. 310, pp. 183–221 (2010) doi:10.1007/978-3-642-14435-6_7
14. Tensor Flow Official website, https://www.tensorflow.org/
15. Abadi, M., Barhman, P., Chen, J., Chen, Z., Davis, A., Dean, J.: tensorflow a system for large-scale machine Learning. In: Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (2016)
16. Juliani, A., Berges, V., Teng, E., Cohen, A.: Unity a general platform for intelligent agents (2018) doi: 10.48550/arXiv.1809.02627
17. Sepúlveda, G., Vega, E., Portilla E.: Machine learning para robots, del entrenamiento virtual a la tarea real. Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI, vol. 7 (2019) doi:10.29057/icbi.v7iEspecial.4785

ISSN 1870-4069

Evaluación de modelos DNN para la detección de objetos en dispositivos de cómputo en la frontera

Alberto Pacheco[1], Ever A. Flores[2], Edgar Trujillo[1]

[1] TecNM / Instituto Tecnológico de Chihuahua,
Posgrado e Investigación, Chihuahua,
México

Robert Bosch GmbH, Guadalajara,
México

{alberto.pg, edgar.tp}@chihuahua.tecnm.mx,
[2] Ever.FloresAvila@mx.bosch.com

Resumen. Dado el creciente interés en aplicar aprendizaje automático dentro del Internet de las Cosas, TinyML emerge para resolver problemas asociados con la adaptación de modelos de redes neuronales profundas para correr en dispositivos pequeños de bajo costo. En este trabajo, se enlistan algunos dispositivos de la frontera con potencial para ser empleados en TinyML. Se presenta una revisión de la literatura de trabajos relacionados sobre el tema a tratar. El trabajo consistió en abordar el problema de evaluar y seleccionar un modelo DNN para la detección de objetos que sea capaz de correr en diversos dispositivos de cómputo en la frontera con procesadores ARM de bajo costo y escasa memoria. Se logró correr un modelo DNN en todos los dispositivos de prueba; se detectó un rendimiento 40x superior en dispositivos ARMv8 respecto a ARMv7, y es posible mejorar su rendimiento hasta un factor 10x usando aceleradores neuronales. Para ejecutar estos modelos DNN en dispositivos pequeños se recomienda aplicar técnicas de optimización para reducir su tamaño y complejidad.

Palabras clave: Redes neuronales profundas, internet de las cosas, TinyML.

Benchmarking of Edge Computing for Object Detection DNN Models

Abstract. Given the growing interest in Machine Learning applied for IoT, TinyML arises as a new research area with the goal of performing on-device inferencing. This work reviews some potential inference computing edge devices. A literature review of recent works is also introduced. The present work carried out an object detection DNN model benchmark for diverse low-cost, low-memory ARM-based devices. The experimental results showed that one DNN model was able to run on all testing devices, where ARMv8-based test devices outperformed ARMv7 by 40 times, while DNN accelerators can boost

performance up to 10 times. To perform on-device inferences in smaller devices, it is also recommended to apply a compression technique.

Keywords: DNN, IoT, TinyML.

1. Introducción

Explosión de datos y dispositivos. Según [1-3] se proyecta para 2021 sea triplicado con respecto a 2018, el tráfico de red proveniente de la nube (*cloud computing*), así como la incorporación para 2030, de 20 a 50 mil millones de dispositivos IoT (*Internet of Things*), mismos que, aparte del creciente número de *smartphones*, se estima incrementen hasta 10 veces la cantidad de datos en Internet, y que el mercado global IoT pase de 16 a 186 mil millones de dólares durante el período 2016-2023.

Retos actuales. Al considerar este crecimiento, tanto de datos como dispositivos IoT, es probable que resulte insuficiente la infraestructura de telecomunicaciones, derivando en mayores latencias de red e insuficiencias en la capacidad de almacenamiento y procesamiento, privacidad y seguridad. Por ello, de acuerdo con [4-8], en el futuro inmediato, entre los principales retos para IoT figuran: ¿cómo extraer información contextual analizando una gran cantidad de datos proveniente de entornos físicos dinámicos y ruidosos? y ¿cómo procesar datos con algoritmos cada vez más inteligentes de una forma eficiente, segura y sustentable?

Estado del arte. El cómputo en la frontera (*edge computing*) ofrece la posibilidad de procesar información de forma distribuida y en tiempo real, con pequeños dispositivos alimentados por baterías, mismos que adquieren datos directamente de sus propios sensores. El cómputo en la frontera ofrece una forma muy eficiente en términos de consumo de energía, latencia y privacidad sin necesidad de saturar la red [4-6]. Por otra parte, las modelos de redes neuronales profundas (*deep neural network*, DNN) son algoritmos muy eficaces para el reconocimiento de patrones en información vasta, compleja y ruidosa.

Al construir y aplicar modelos DNN existen dos etapas: el entrenamiento y la inferencia (ejecución del modelo DNN). TinyML representa el estado del arte en IoT [7-10], al potenciar las capacidades del cómputo en la frontera para la etapa de inferencia dentro de dispositivos pequeños de bajo costo y memoria reducida, posibilitando el procesamiento de datos de manera inmediata y privada, evitando así, la latencia y congestionamiento de la red, cubriendo además, un amplio espectro de aplicaciones IoT, tales como: eHealth, smart spaces, transporte inteligente, industria 4.0 y agricultura de precisión, entre otros [13-14, 29-30].

Planteamiento del problema. El presente trabajo consiste en evaluar y seleccionar un modelo DNN para detección de objetos que sea capaz de correr en diversos dispositivos con procesadores ARM de bajo costo y escasa memoria, esperando obtener un rendimiento de 2-5 FPS para soportar la viabilidad de su aplicación [16]. La Tabla 1 plantea la arquitectura de un sistema TinyML orientado a la detección de objetos en secuencias de vídeo (a), usando modelos DNN pre-entrenados basados en redes convolucionales (b), para la etapa de inferencia (c), sin esquemas de simplificación o reducción de modelo DNN (d), usando distintos frameworks ML para comparar el rendimiento del modelo DNN corriendo en diferentes dispositivos y plataformas (f-h).

Tabla 1. Sistema TinyML basado en redes neuronales.

Dimensión	Características	Opciones
Aplicaciones	a) Visión por computadora	Detección de objetos
Modelos y arquitecturas	b) Modelos y arquitecturas de redes neuronales	CNN, FF, RNN, GAN, LSTM, SOM, SAE, SVM, ART, etc
	c) Entrenamiento e inferencia de modelos	Supervisado, no supervisado. Prueba, validación, aplicación.
	d) Optimización de modelo DNN	Cuantización, poda, SVD, compresión, destilación, etc.
	e) Frameworks y formatos	Keras, Caffe, TensorFlow, Pytorch, CoreML, MXNet, OpenVino, etc.
Hardware	f) Computadora (procesadores)	CPU, GPU, TPU.
	g) Sistema operativo	Linux, Windows, macOS, iOS, RT.
	h) Acelerador neuronal	FPGA, DSP, NPU, acelerador DNN (Movidius, Coral), etc.

El presente trabajo es la segunda de tres etapas de un proyecto concluido recientemente [15-16], para la cual se establecieron los siguientes subproblemas:

Comparar el rendimiento de una red neuronal DNN entrenada para la detección de objetos a partir de imágenes de vídeo, cuya etapa de inferencia sea verificada en distintos dispositivos.

- Determinar el flujo de trabajo necesario para portar y habilitar modelos DNN pre-entrenados (deployment) en diversos dispositivos.
- Elegir un modelo DNN para detección de objetos con base en los criterios de eficiencia y portabilidad para el ecosistema TinyML heterogéneo.
- Caracterizar y determinar cuál configuración ofrece el mejor rendimiento para el modelo DNN elegido.

2. Trabajos relacionados

La ejecución optimizada de modelos ML en dispositivos de bajo costo es un campo emergente de desarrollo e investigación. En 2015, DeepEar [13], un trabajo pionero, ejecutó un modelo DNN para reconocimiento de voz en un DSP de baja potencia. El consorcio de benchmarks para sistemas embebidos (EEMBC), liberó en 2019 la primera prueba TinyML denominada MLMark [17]. Como revelan [18-19] y eventos como TinyML Summit 2019-2021 [20], existe un creciente número de fabricantes y desarrolladores de dispositivos y aplicaciones para TinyML.

A continuación, revisamos la evidencia existente en la literatura de investigación. Se realizó una búsqueda en marzo de 2021 en la biblioteca digital de ACM, en la colección *The ACM Guide to Computing Literature* (2,923,187 registros) buscando dentro de los resúmenes (*abstract*) de "*research articles*", con una expresión de búsqueda para

Tabla 2. Arquitecturas para realizar inferencias ML [14].

	Cloud ML	**Mobile ML**	**Tiny ML**
Memoria (activación)	>16GB	4GB	200-400Kb
Memoria (pesos DNN)	>TB	<256GB	<1MB

trabajos sobre inferencia ML/CN/DL o TinyML en dispositivos móviles, edge, fog, IoT, embebidos con CPUs ARM, NPUs, FPGAs con/sin técnicas de compresión de modelos DNN, excluyendo cómputo en la nube, big data, TPUs, etc. para 2010-2021 [21]:

```
[[inferenc* OR optim*] AND [ml OR cnn OR dl OR ai OR "machine learning" OR tinyml
OR "deep learning"] AND [edge OR fog OR "mobile device" OR "on-device" OR fpga
OR arm OR smartphone OR npu OR iot OR embed*] AND NOT [cloud OR "large-scale"
OR cluster OR server* OR "big data" OR tpu] AND [Publication Date: (01/01/2010
TO 12/31/2021)]
```

Se encontraron 506 artículos (Tabla 6), distribuidos como sigue: 327 en *Proceedings* y 153 en *Journals* y revistas periódicas, destacando *ACM Trans. on Embedded Computing Systems* con 11 artículos y *Proc. ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* con 9 artículos. Entre las principales organizaciones editoriales se encuentran: ACM con 323, IEEE con 101 y Elsevier con 26 artículos.

Dentro del Top 10 de las organizaciones participantes destacan: 2 universidades chinas (Pekín y Tsinghua con 26), 3 universidades americanas (Stanford, Illinois y California con 19), 2 empresas americanas (Intel y Microsoft con 17), 2 universidades de Singapur (Nayuang y Singapur con 12) y una universidad inglesa (8 Cambridge).

También, en la misma fecha, se realizó la misma búsqueda en IEEEXplore (5,271,145 ítems disponibles), donde se encontraron 1,817 artículos (Tabla 6) con 1,352 artículos de Proceedings, 443 de Journals y 22 de Magazines IEEE entre los años 2010 y 2021, usando la misma expresión de búsqueda [22].

Destacan IEEE Access (135), seguido de IEEE Internet of Things Journal (31). Entre los países con más artículos aparecen: 23 artículos americanos, 17 chinos, 8 ingleses y 4 de Singapur. Como muestra la Tabla 6, el promedio de crecimiento anual es alrededor de 30% para 2019-2020 considerando ambas (IEEE y ACM).

En cuanto a los artículos con un mayor número de citas por año en IEEE más citados se encuentran [23-25] con 130, 88 y 83 citas/año, respectivamente. Los más citados en ACM DL son [26-28] con 130, 38 y 38 citas/año, respectivamente.

Trabajos relacionados. En síntesis, existen tres categorías de trabajos relacionados con la ejecución de modelos DNN en dispositivos pequeños y limitados [28-29]: 1) los modelos DNN ligeros (*small footprint DNN models*) entrenados para tareas muy simples y específicas, tales como GooLeNet, ResNet, SENet, SuffleNet; 2) la incorporación de unidades de aceleración de modelos DNN, como DeepX, DeepSense, DeepMon, ProjectionNet, QNNPack, Movidius, Coral; 3) la aplicación de diversas técnicas para re-entrenar, reducir tamaño y complejidad de modelos DNN, tales como la poda, cuantización, compresión, destilación, entre otros. El presente trabajo, se enfoca exclusivamente a las dos primeras categorías.

Tabla 3. Tiny ML en procesadores ARMv7 (32 bits).

Dispositivos (32 bits)	**CPU ARMv7 (núcleos)**	**Reloj RAM Flash**	**Consumo energía (Watts)**	**Precio (año)**
Raspberry Pi Pico	Cortex-M0+ (2 núcleos)	133 MHz 264 KB 2 MB	0.1 W	$4 (2021)
Espressif ESP32	Xtensa LX6 (2 núcleos) 0.006 GFLOPS	240 MHz 320 KB 4 MB	0.2 W	$10 (2016)
SparkFun Edge	Cortex-M4F (1 núcleo)	48 MHz 384 KB 1 MB	0.06 W	$15 (2019)
Silicon Labs SLTB004A	Cortex-M4 (1 núcleo)	40 MHz 256 KB 1 MB	0.1 W	$20 (2018)
Arduino Nano 33 BLE Sense Adafruit Circuit Playground	Cortex-M4F nRF 52840 (1 núcleo)	64 MHz 256 KB 1MB	0.2 W	$30 (2019) $20 (2018)
Adafruit Edge/Py Badge Wio Terminal	Cortex-M4F ATSAMD51 (1 núcleo)	120 MHz 192 KB 512 KB	~0.3 W	$35 (2019) $30 (2015)
SMT32F746 Discovery kit	Cortex-M7 STM32F746 (1 núcleo) 0.022 GFLOPS	216 MHz 340 KB 1 MB	~0.9 W	$50 (2015)

Tabla 4. Dispositivos para TinyML con procesadores ARMv8 (64 bits).

Dispositivos (64 bits)	**CPU ARMv8 (núcleos)**	**Reloj RAM Flash**	**Rendi-miento GFLOPs**	**Potencia (watts)**	**Precio**
Raspberry Pi 3B	Cortex-A53 (4 núcleos)	1.2 GHz 1 GB	1-4	2-5 W	$35 (2016)
Raspberry Pi 4	Cortex-A72 (4 núcleos)	1.5 GHz 2/8 GB	5-10	5-8W	$35/75 (2019)

3. Dispositivos para TinyML

En la Tabla 2 se describen las propiedades típicas de los elementos de cómputo para los principales tipos de sistemas con modelos DNN. A diferencia de los predominantes sistemas de cómputo en la nube, para el cómputo en la frontera, interesa principalmente ejecutar modelos DNN en dispositivos móviles y dispositivos IoT o embebidos que cuentan con un hardware muy limitado y de menor costo, conocidos de distintas formas: *Tiny ML, edge AI, smart IoT, on-device inferencing y edge inference* [8-11, 12-13].

Debido a que los modelos DNN poseen muchas capas y millones de parámetros de punto flotante, típicamente float32, existe un campo muy activo de investigación

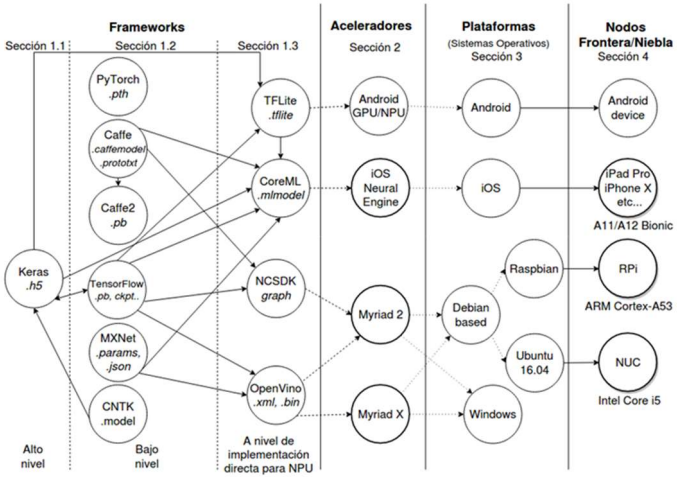


Fig. 1. Flujo de trabajo para transferir modelo DNN a los distintos dispositivos.

enfocado en la reducción y compresión de dichos modelos DNN a escalas y precisiones más acordes para dispositivos pequeños, brindando como beneficio, la posibilidad de procesar grandes cantidades de datos complejos, dinámicos e inciertos, permitiendo así procesar flujos continuos de datos adquiridos de dispositivos sensores que van desde video, micrófonos, acelerómetros, altímetros, termómetros, etc. para identificar patrones, eventos y objetos de interés directamente en la frontera de la red.

Revisión de dispositivos ARMv7 con potencial para TinyML. Como muestra la Tabla 3, a diferencia de los equipos de cómputo en la nube, existe una categoría de dispositivos embebidos (SoC) para TinyML con procesadores de 32 bits (ARMv7) que cuestan menos de $50 dlrs, corren a ~300 MHz, tienen 200-400 KB RAM y consumen miliwatts (usan pilas). A su vez, podemos segmentar esta categoría en dos partes (Tabla 3): a) los dispositivos más pequeños, con muy bajo consumo energético y más baratos (menos de $10 dlrs); b) los dispositivos con precio de $15-50 dlrs, CPU Cortex-M (1 núcleo), que pueden correr modelos DNN muy simplificados y reducidos.

Dispositivos ARMv8 para TinyML. La Tabla 4 muestra equipos de 64 bits de bajo costo (ARMv8), con sistemas operativos tipo Linux o para dispositivos móviles (smartphones), consumo de 2-8 watts, y suficiente memoria (hasta 8 GB RAM) para ejecutar modelos DNN de medianas proporciones. En la Tabla 4 solo se muestran modelos similares a los usados en el trabajo (RPi). Esta categoría está ganando gran auge y nuevos modelos en el mercado (e.g. RPi4), y el surgimiento de la nueva arquitectura ARMv9 para TinyML.

Aceleradores DNN para TinyML. La Tabla 5 muestra aceleradores de hardware para modelos DNN [31], cuya arquitectura interna y conjunto de operaciones que pueden ejecutarse en paralelo y son más específicas para ejecutar modelos DNN, acelerando hasta 10 veces más la ejecución comparada con CPUs de precio similar, consumiendo además menos energía.

Algunas de las tarjetas de desarrollo de la Tabla 5 son más sofisticadas porque incluyen varios procesadores (ARM, GPU, DSP, VPU, TPU multicore) que elevan considerablemente el costo y consumo de potencia.

Tabla 5. Tarjetas de desarrollo TinyML y aceleradores DNN.

Dispositivo	**Unidades de aceleración**	**Arquitectura CPU**	**Rendi-miento GFLOPs (FP16)**	**Consumo de energía**	**Precio (año)**
Intel Movidius (NCS)	VPU 12 cores 4 GB RAM	Ninguno	40	1 W	$75 (2017)
Intel Movidius 2 (NCS2)	VPU 16 cores 4 GB RAM	Ninguno	~100	1.5 W	$75 (2018)
Google Coral Accelerator	TPU	Ninguno	~250	2 W	$75 (2019)
NVIDIA Jetson Nano	GPU 128 cores 4 GB RAM	Cortex-A57 4 cores	500	10 W	$130 (2019)
Google Coral Dev Board	TPU GPU 1GB RAM	Cortex-A53 4 cores	~600	15 W	$150 (2019)
ASUS Tinker Edge R	TPU GPU NPU 6 GB RAM	Cortex-A72+A53 6 cores	~1,200	15W	$250 (2020)
NVIDIA Jetson TX2	GPU 256 cores 8 GB RAM	Cortex-A57 4 cores +ARMv8 2 cores	1,300	15 W	$400 (2021)
NVIDIA Jetson AGX Xavier	GPU 512 cores 32 GB RAM	ARMv8 8 cores	11, 000	50 W	$700 (2021)

4. Metodología

Para realizar exitosamente la comparación de los tiempos de inferencia en modelos DNN en diversos dispositivos fue necesario llevar a cabo un flujo de trabajo (*workflow*) para portar los modelos DNN a los formatos específicos de cada plataforma y hardware de los dispositivos de prueba, lo cual, a su vez permitió derivar una perspectiva práctica para implementar las pruebas en distintos tipos de procesadores, siguiendo para ello, la siguiente metodología:

a. Etapa de investigación: la primera etapa consistió investigar modelos DNN pre-entrenados viables de ser ejecutados en cada dispositivo de prueba, con el objetivo de obtener al menos un modelo DNN capaz de ser ejecutado en todos los dispositivos para asegurar la confiabilidad de la evaluación comparativa.

b. Etapa de implementación: en esta etapa se realizó la conversión e implementación del modelo DNN para cada dispositivo de prueba.

Tabla 6. Artículos publicados entre 2010-2020.

Año	Artículos ACM	Artículos IEEE
2010	7	22
2011	3	22
2012	9	25
2013	9	18
2014	9	26
2015	14	48
2016	44	74
2017	53	152
2018	77	282
2019	109	470
2020	143	594

c. Etapa de evaluación: finalmente, se calculó el tiempo de procesamiento de detección de objetos presentes en imágenes de video, considerando la métrica de cuadros por segundo (*frames per second*, FPS) para luego, comparar y analizar los resultados obtenidos.

5. Desarrollo

A continuación, se presentan los dispositivos de prueba considerados en esta evaluación comparativa, así como sus características principales:

- Computadora Intel NUC 6-i5SYH con i5-6260U 64-bit CPU @ 1.8 GHz, 16 GB RAM (15 watts) con Ubuntu 16.04 LTS (Linux Kernel 4.4) instalado.
- Computadora Raspberry Pi 3B (RPi), quad-core ARMv8 64-bit CPU @ 1.2GHz, 1GB RAM (5W) con Raspbian 2018 (Linux Kernel 4.9) instalado.
- Dos aceleradores DNN intel Movidius Neural Compute Stick (NCS) con VPU Myriad 2 de 1 watt y una NCS2 con VPU Myriad X de 1.5 watts.
- Un dispositivo móvil Apple iPad 6th-Gen 9.7" con iOS 12, quad-core A10 Fusion ARMv8 64-bit CPU @ 2.34 GHz y 2 GB RAM.
- Un teléfono inteligente Apple iPhone 8 con iOS 12, hexa-core A11 Bionic ARMv8 64-bit CPU @ 2.39 GHz y 2 GB RAM.

Etapa de investigación. Se seleccionaron 5 modelos DNN pre-entrenados para la detección de objetos: YOLOv3, TinyYOLOv2, TinyYOLOv3, SSD + MobileNet y SSDLite + MobileNetv2. La Figura 1 sumariza el flujo de trabajo efectuado para intercambiar el formato de un modelo DNN para soportar cada uno de los distintos sistemas operativos, dispositivos de prueba y aceleradores DNN.

Frameworks ML (sección 1). Los modelos DNN son muy dependientes del formato del framework ML en que fueron entrenados. Esto representa un problema al momento de tratar de ejecutarlos en un hardware distinto, donde dependiendo del fabricante, puede que no exista soporte para un determinado formato. Sin embargo, es posible convertir el formato de un modelo DNN, como muestra la sección 1 de la Figura 1, donde las flechas representan las conversiones de formato y los nodos indican el formato soportado por cada framework (segundo renglón de cada nodo). En la sección

Tabla 7. Resultados de rendimiento (FPS) de cada modelo DNN en cada dispositivo.

		YOLOv3	**Tiny YOLOv3**	**Tiny YOLOv2**	**SSD + MobileNet**	**SSDLite + MobileNetv2**
RPi	CPU	-	0.5	0.5	-	-
	1 NCS	0.5	3.7	2.5	9.6	6.0
	2 NCS	-	-	5.8	13.4	-
	1 NCS2	1.0	3.9	3.2	11.9	6.5
NUC	CPU	1.8	14.7	5.5	-	54.0
	1 NCS	0.7	6.5	4.7	11.3	9.0
	2 NCS	1.3	13.0	12.4	19.0	-
	1 NCS2	2.7	30.0	10.2	20.0	10.0
iOS	iPad 6G	2.5	-	5.9	-	6.0
	iPhone 8	3.0	-	27.0	-	9.0

1.1 aparece el framework ML de alto-nivel Keras, que presenta múltiples facilidades para la creación, entrenamiento y modificación de modelos DNN.

Los frameworks de la sección 1.2 son herramientas de bajo nivel con las cuales se realiza el diseño y evaluación de modelos de manera más eficiente y con mayor flexibilidad en comparación con una herramienta de alto nivel como Keras. En este punto, es posible convertir desde Keras a casi cualquier otro formato de bajo nivel pasándolo por ejemplo al formato ONNX, y viceversa, incluyendo también otros como OpenVino y CoreML.

Finalmente, en la sección 1.3 se encuentran los frameworks para instalar el modelo DNN (*deployment*) directamente en algún dispositivo o aceleradores DNN, en otras palabras, en los formatos nativos para cada equipo. Se recomienda convertir de un formato a otro de manera directa, ya que a pesar de que exista el soporte de conversión, en ocasiones los modelos DNN son diseñados con operaciones o capas muy específicas de un determinado framework ML, ocasionando que durante la conversión algunas operaciones sean modificadas y afecten la calidad o el funcionamiento del mismo.

Aceleradores DNN, sistemas operativos y dispositivos (secciones 2-4). En la sección 2 de la Figura 1 se señalan cuáles formatos son compatibles con los aceleradores DNN y cómo puede afectar además el tipo de sistema operativo instalado en el dispositivo de prueba al que se conectan por puerto USB (sección 3).

Si bien una manera de evitar los problemas de incompatibilidad de formatos es creando nuevos modelos DNN para cada caso de aplicación, pero esta opción es poco viable en la mayoría de las ocasiones debido a que si bien la arquitectura de la red neuronal se puede replicar en otro framework, sería necesario entrenar ese modelo, y con ello surge la necesidad de grandes cantidades de datos y capacidades de cómputo para entrenamiento, por ello la importancia de conseguir reutilizar modelos pre-entrenados para convertirlos a otras plataformas.

En este trabajo se realizaron las conversiones de formato indicadas hasta llegar a los dispositivos de prueba de la sección 3, exceptuando la plataforma Android que aparece como posible referencia.

Etapas de implementación. Se utilizó TensorFlow para realizar las conversiones de formatos para los equipos NUC y RPi. Adicionalmente, en estos dos dispositivos se hicieron evaluaciones agregando los aceleradores Intel Movidius, en cuyos casos se empleó OpenVino para convertir del formato de TensorFlow para los modelos DNN

ejecutados en dichos aceleradores, cuyo formato OpenVino consiste de dos archivos, uno para almacenar la información de la estructura de la red neuronal (XML) y otro para almacenar los pesos de la red (formato binario). De manera similar, para los dispositivos iOS, los modelos DNN se convirtieron a formato. MLmodel para ser procesados por medio del framework CoreML de iOS.

Etapa de evaluación. Para evaluar los modelos DNN se desarrollaron distintos scripts en Python para ejecutarlos en las computadoras de prueba (NUC, RPi) y una aplicación móvil de prueba en iOS. Las pruebas consistieron en obtener imágenes de video de una webcam, realizar el preprocesamiento correspondiente y procesar el modelo DNN para obtener las inferencias y desplegar resultados de la detección de objetos sobre la imagen de vídeo de salida en pantalla.

Para comparar resultados de rendimiento se calculó el promedio de cuadros de video (FPS) procesados por segundo para una misma muestra de video de prueba consistente en 200 cuadros de video de entrada. En la Tabla 7 se muestran los resultados obtenidos para dispositivos y modelos DNN de detección de objetos, siendo TinyYOLOv2 el único modelo que pudo ser ejecutado exitosamente en todas las siguientes combinaciones:

- Con un acelerador DNN Movidius NCS (Myriad 2).
- Con un acelerador DNN Movidius NCS2 (Myriad X).
- Con dos aceleradores DNN Movidius NCS.
- En una computadora con CPU 64 bit NUC Intel Core i5.
- En una computadora con CPU 32 bit RPi ARM Cortex-A53.

6. Resultados

Si asumimos como rendimiento mínimo aceptable para soportar una aplicación práctica para detección de objetos en video un rango ubicado entre 2 y 5 FPS [32], tenemos que, si bien, el equipo de prueba con menor capacidad (RPi ARMv8) logró ejecutar 2 de 5 modelos DNN (los modelos más pequeños TinyYOLO), su rendimiento resultó inadecuado en términos prácticos (0.5 FPS).

Sin embargo, al incorporar uno o dos aceleradores DNN en ese mismo equipo, se logró un resultado aceptable en 4/5 casos, destacando incluso que para TinyYOLOv2 (5.8 vs 5.5 FPS) superó al equipo con CPU de mayor capacidad (NUC i5) y se observa que 2 unidades NCS brindan mejores resultados que 1 unidad NCS2 para TinyYOLO.

Desafortunadamente, los modelos DNN más demandantes (YOLOv3 y MobileNet) no se lograron ejecutar en la computadora RPi usando CPU ni tampoco usando dos aceleradores DNN, debido en este último caso al alto consumo de energía que demandaban. Sin embargo, múltiples aceleradores corrieron sin problema en la computadora NUC que, además, al contar con puertos USB3.0, logrando así el mejor rendimiento de dichas unidades de aceleración, destacando que, en 4 de 5 casos, el rendimiento del acelerador NCS2 superó en alrededor de 45% al procesador de la computadora (Core i5).

Finalmente, en dispositivos móviles iOS se ejecutaron 3 modelos DNN, donde destaca que se logró el mayor rendimiento (iPhone 8) para el modelo pequeño TinyYOLOv2 (27 FPS) y en los demás modelos DNN más pesados se logró un rendimiento promedio, que sin embargo, demuestra que los dispositivos móviles más

recientes tienen buenas capacidades para ejecutar este tipo de modelos DNN, donde resulta interesante las capacidades que ofrece el procesador del iPhone 8, A11 Bionic con 6 núcleos (2 núcleos de alto rendimiento), un GPU de 3 núcleos y un nuevo procesador neuronal integrado (*neural engine*), que sin embargo, no fue posible determinar en las prueba realizadas con este dispositivo, en qué tipo de unidad de procesamiento (CPU, GPU, NPU) fueron realizadas las inferencias de los modelos DNN probados.

Por último, es posible observar algunas variaciones no proporcionales de los rendimientos entre los diferentes modelos y dispositivos (e.g. 12.5 FPS de NCS2 en NUC para TinyYOLOv2 y 54 FPS con CPU de NUC para MobileNetv2). En este trabajo no se profundizó en esos casos, aunque es útil señalar que los aceleradores están optimizados para cierto tipo de operaciones, por lo cual a veces un CPU puede ser más eficiente dependiendo de la arquitectura específica de un modelo DNN, y algo similar ocurre entre distintos tipos de CPU.

7. Conclusiones

En este trabajo se logró convertir y ejecutar 5 modelos DNN para detección de objetos en distintas combinaciones de dispositivos y aceleradores DNN. Debido a su reducido tamaño y mayor compatibilidad (tipos de capas/operaciones), el modelo TinyYOLOv2 fue el único modelo ejecutado con éxito en todos los casos de prueba.

Destaca también la importante mejora en el rendimiento que proporcionan los aceleradores DNN, sobre todo en los dispositivos de menos capacidad, brindando así una opción de bajo costo y consumo de energía. También resulta muy notable la capacidad y el nivel de desempeño que brindan los teléfonos inteligentes de alta gama para ejecutar modelos DNN y, por tanto, posibilitan una nueva generación de aplicaciones móviles donde cada vez es más posible aplicar la Inteligencia Artificial en la vida cotidiana.

Se encontró además que para ejecutar modelos DNN en dispositivos pequeños (TinyML), hay dos obstáculos importantes: a) el excesivo tamaño de la mayoría de los modelos DNN supera, por lo general, la capacidad de procesamiento y memoria disponible en dichos dispositivos; b) una de las tareas más complicadas es migrar dichos modelos de su formato original al formato de la plataforma-destino, donde las diferentes capas y operaciones del modelo que cada plataforma/framework soporta, pueden en el proceso de conversión alterar la estructura interna del modelo o terminar siendo incompatible.

Por ello, para aplicaciones que incluyan distintos dispositivos, sean móviles, embebidos o aceleradores DNN, es útil comprender primero el grado de compatibilidad que soportan, donde la Figura 1 sirvió de guía en este trabajo, al recopilar distintas rutas posibles para convertir el formato de un modelo DNN. Por otro lado, los frameworks CoreML (iOS) y OpenVino (aceleradores DNN) soportan un gran número de conversiones de formato y tipos de operaciones, elevando su grado de compatibilidad para soportar más plataformas de hardware.

Como trabajo a futuro, se considera evaluar dispositivos recientes (RPi 4, nuevos procesadores móviles y las nuevas arquitecturas RISC-V y ARMv9). Para la siguiente etapa del proyecto, existe una gran oportunidad de mejora en el rendimiento de modelos

DNN al aplicar técnicas de optimización, para podar, comprimir y acelerar los modelos antes de su despliegue, buscando así, desde otro frente, acortar la brecha entre dispositivos de la frontera y redes neuronales profundas.

Referencias

1. Pepper, R.: Cisco visual networking index: Global mobile data traffic forecast update. Cisco, White Paper, (2017) https://goo.gl/2aVofM
2. WinterGreen Research: Internet of Things (IoT) Market Shares. Strategies, Forecasts, Worldwide, 2017 to 2023, Lexington, Massachusetts, Market Analysis Report (2017) https://goo.gl/bDZDB7
3. Grand View Research: Smart home automation market size. Industry Trend Report 2014-2025, Report USA, Aug (2017) https://goo.gl/FCRteF
4. Bilal, K.: Potentials, trends, and prospects in edge technologies: Fog, cloudlet, mobile edge and micro data centers. Computer Networks, vol. 130, pp. 94–120 (2018) doi: 10.10 16/j.comnet.2017.10.002
5. Satyanarayanan, M.: The emergence of edge computing. Computer, vol. 50, no. 1, pp. 30–39 (2017) doi:10.1109/MC.2017.9
6. Iorga, A.: Fog computing conceptual model. National Institute of Standards and technology. NIST Special Publication U.S. Dept. of Commerce, pp. 500–325 (2018) doi:10.6028/NIST.SP.500-325
7. Mohammadi, M., Al-Fuqaha, A.: Enabling cognitive smart cities using big data and machine learning: Approaches and challenges. IEEE Communications Magazine, vol. 56, no. 2, pp. 94–101 (2018) doi:10.1109/MCOM.2018.1700298
8. Li, H., Ota K., Dong, M.: Learning IoT in edge: Deep learning for the internet of things with edge computing. IEEE Network, vol. 32, no. 1, pp. 96–101 (2018) doi:10.11 09/MNET.2018.1700202
9. Li, Q., Xiao, Q., Liang, Y.: Enabling high performance deep learning networks on embedded systems. In: Proceedings of IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society, pp. 8405–8410 (2017) doi:10.1109/IECON.2017.8217476
10. Verhelst, M., Moons, B.: Embedded DNN processing: Algorithmic and processor techniques bring deep learning to IoT and edge devices. IEEE Solid-State Circuits Magazine, vol. 9, no. 4, pp. 55–65 (2017)
11. Gershgorn, D.: Forget the space race. The Artificial Intelligence Race is Just Beginning, (2018) http://shamov.info/node/652
12. Georgiev, P.: Low-resource multi-task audio sensing for mobile and embedded devices via shared deep neural network representations. In: Proceedings of ACM Interact. Mob. Wearable Ubiquitous Tech, vol. 1, no. 50, pp. 1–19 (2017) doi: 10.1145/3131895
13. Lane, N. D., Bhattacharya, S., Mathur, A., Georgiev, P., Forlivesi, C., Kawsar, F.: Squeezing deep learning into mobile and embedded devices. IEEE Pervasive Computing, vol. 16, no. 3, pp. 82-88 (2017) doi:10.1109/MPRV.2017.2940968
14. Han, S.: Putting AI on a diet: TinyML and efficient deep learning. TinyML Summit Keynote (2021) doi: 10.1109/VLSI-DAT52063.2021.9427348
15. Pacheco, A., Flores, E., Sánchez, R., Almanza, S.: Smart classrooms aided by deep neural networks inference on mobile devices. In: Proceedings of IEEE International Conference on Electro/Information Technology (EIT), pp. 605–609 (2018) doi:10.1109/EIT.20 18.8500260
16. Pacheco, A., Cano, P., Flores, E., Trujillo E., Marquez, P.: A smart classroom based on deep learning and osmotic IoT computing. In: Proceedings of Congreso Internacional Innovación y Tendencias en Ingeniería, pp. 1–5 (2018) doi:10.1109/CONIITI. 2018.8587095

17. EEMBC. Benchmarks, Embedded Microprocessor Benchmark Consortium (2021) https://www.eembc.org/products/
18. Tang, S.: The AI chips list: A list of ICs and IPs for AI, ML and DL. Github repository, (2021) https://basicmi.github.io/AI-Chip/
19. Allan, A.: Benchmarking edge computing: Comparing Google, Intel and NVIDIA accelerator hardware. Medium blog, (2019) https://aallan.medium.com/benchmarking-edge-computing-ce3f13942245
20. TinyML Foundation. In: TinyML Summit conference and research symposium, online event, California, USA (2021) https://www.tinyml.org/event/summit-2021
21. ACM Search query for machine learning inference research publications from 2010 to 2021. ACM Digital Library, http://tiny.cc/comia1
22. IEEE Search query for machine learning inference research publications from 2010 to 2021. IEEEXplore Digital Library, https://aka.my/i8y
23. Li, H., Ota, K., Dong, M.: Learning IoT in edge: Deep learning for the internet of things with edge computing. IEEE Network, vol. 32, no. 1, pp. 96–101 (2018) doi:10.1109/MNET.2018.1700202
24. Jacob, B.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, USA, pp. 2704–2713 (2018) doi:10.1109/CVPR.2018.00286
25. Wang, S., Tuor, T., Salonidis, T., Leung, K. K., Makaya, C., He, T., Chan, K.: Adaptive federated learning in resource constrained edge computing systems. IEEE J. Sel. Areas in Communications, vol. 37, no. 6, pp. 1205–1221 (2019) doi:10.1109/JSAC.2019.2904348
26. Zhang, C.: Optimizing FPGA-based accelerator design for deep convolutional neural networks. In: Proceedings of ACM/SIGDA International Symposium Field-Programmable Gate Arrays. ACM, NY, USA, pp. 161–170 (2015) doi:10.1145/2684746.2689060
27. Ma, Y.: Optimizing loop operation and dataflow en FPGA acceleration of deep CNNs, In: Proceedings of 2017 ACM/SIGDA International Symposium Field-Programmable Gate Arrays, NY, USA, pp. 45–54 (2017) doi:10.1145/3020078.3021736
28. Shen, Y., Ferdman, Y., Milder, P.: Maximizing CNN accelerator efficiency through resource partitioning. SIGARCH Computer Architectures News, vol. 45, no. 2, pp. 535–547 (2017) doi:10.1145/3140659.3080221
29. Sanchez-Iborra, R., Skarmeta, A. F.: TinyML-enabled frugal smart objects: Challenges and opportunities. IEEE Circuits and Systems Magazine, vol. 20, no. 3, pp. 4–18 (2020) doi:10.1109/MCAS.2020.3005467
30. Dai, X., Spasić, I., Chapman, S., Meyer, B.: The state of the art in implementing machine learning for mobile apps: A survey. SoutheastCon, pp. 1–8 (2020) doi:10.1109/SoutheastCon44009.2020.924965
31. Xie, Y.: A brief guide of xPU for AI accelerators. ACM SIGARCH (2018) https://www.sigarch.org/?p=15093
32. Nikouei, S.: Real-time human detection as an edge service enabled by a lightweight CNN. In: IEEE International Conference on Edge Computing, pp. 125–129 (2018) doi10.1109/EDGE.2018.00025

ISSN 1870-4069

Sistema de procesamiento de IDs basado en modelos de aprendizaje profundo

Raúl Aguilar-Figueroa, Rolando Borja-Brito,
Carlos Daniel Virgilio-González

Biometría Aplicada,
Departamento de Innovación,
México

{raguilar, rborja, cvirgilio}@biometriaaplicada.com

Resumen. Una de las fases del onboarding digital, consiste en autenticar la identidad del cliente a partir de la captura de una fotografía de un documento de identidad (ID). Esta imagen del ID suele adquirirse en entornos donde factores como la orientación o la presencia de fondos, complican la correcta ejecución de tareas posteriores en el flujo del onboarding. Con la finalidad de superar estos inconvenientes, en el presente artículo se introduce un sistema para el procesamiento de IDs, el cual localiza, segmenta y alinea el ID a través de módulos construidos completamente en función del enfoque del aprendizaje profundo. Para la tarea de localización, se utiliza el algoritmo de detección YOLOv4; la segmentación del ID se realiza con la arquitectura U-Net, y la alineación del ID se logra con el sistema RotNet. Buscando incrementar el tamaño de la base de datos de IDs, se aplica una metodología de aumentación de datos que genera imágenes de IDs sintéticas. Los resultados obtenidos al evaluar los modelos de localización, segmentación y alineación con base en credenciales de elector de los Estados Unidos Mexicanos, resaltan su robustez ante las variaciones agresivas que suelen tener las imágenes de entrada, pues el modelo de localización alcanza un rendimiento de 0.9634 con la métrica *IoU*, por 0.9820 del modelo de segmentación de acuerdo a la métrica *coeficiente de dice*, mientras que los ángulos inferidos por el modelo de alineación tienen una discrepancia promedio de apenas 1.5920 con respecto a los ángulos verdaderos.

Palabras clave: IDs, onboarding digital, aprendizaje profundo, YOLOv4, U-Net, RotNet.

ID Processing System based on Deep Learning Models

Abstract. One of the phases of digital onboarding, consists of authenticating the identity of the client by capturing a photograph of an identity document (ID). This image of the ID is usually acquired in environments where factors such as orientation or the presence of backgrounds, complicate the correct execution of subsequent tasks in the onboarding flow. In order to overcome these

drawbacks, this article introduces a system for the processing of IDs, which locates, segments and aligns the ID through modules built entirely based on the deep learning approach. For the localization task, the YOLOv4 detection algorithm is used; the segmentation of the ID is done with the U-Net architecture, and the alignment of the ID is achieved with the RotNet system. Seeking to increase the size of the ID database, a data augmentation methodology capable of generating synthetic ID images is applied. The results obtained when evaluating the localization, segmentation and alignment models based on voter credentials from the United Mexican States, highlight their robustness in the face of the aggressive variations that input images usually have, since the localization model achieves a peformance of 0.9634 with the *IoU* metric, by 0.9820 of the segmentation model according to the *dice coefficient* metric, while the angles inferred by the alignment model have an average discrepancy of only 1.5920 with respect to true angles.

Keywords: IDs, digital onboarding, deep learning, YOLOv4, U-Net, RotNet.

1. Introducción

En la actualidad, debido a factores como la masificación de los dispositivos conectados a internet y la crisis sanitaria provocada por el COVID-19, la implementación de sistemas de onboarding digital se ha convertido en una necesidad prioritaria en el sector financiero y en la industria en general, ya que para incorporar o dar de alta a nuevos clientes, el procedimiento tradicional que implica que el cliente se traslade físicamente a la institución y lleve a cabo largos trámites presenciales, es reemplazado por un proceso completamente remoto y automatizado, el cual genera una reducción en los costos financieros, una mejor experiencia del usuario y la eliminación de los formularios de papel, entre otros beneficios [10, 2, 8]. Una de las fases del onboarding digital, consiste en autenticar la identidad del cliente a partir de la captura de una fotografía de un documento de identidad (ID) [16].

En este escenario, se vuelve indispensable contar con sistemas de análisis de documentos que permitan procesar la imagen del ID con el objetivo de realizar tareas posteriores en el flujo del onboarding, como la extracción de información de interés, la verificación de la pertenencia del documento a una categoría específica o la identificación de documentos falsificados [1, 7].

Estos sistemas de procesamiento de IDs, deben ser capaces de operar en entornos no controlados, ya que es probable que las fotografías de los IDs ingresados por los usuarios, contengan variaciones ocasionadas por cambios en la perspectiva, la iluminación, la orientación y la traslación.

Asimismo, es muy común que las fotografías de IDs presenten distintos fondos, siendo este un aspecto trascendental que dificulta la localización de la región correspondiente al ID, y que es causado por elementos adversos presentes en la captura realizada, tales como el bajo contraste o la similitud de las tonalidades de los colores entre el fondo y el ID [3].

Con la intención de superar los problemas previamente mencionados, en este artículo se presenta un sistema para el procesamiento y análisis de IDs que permite localizar, segmentar y alinear el ID, utilizando para cada una de estas etapas, módulos expresamente construidos con base en un área del aprendizaje máquina conocida como aprendizaje profundo.

De acuerdo a nuestro conocimiento, el sistema propuesto se destaca como el primer esfuerzo en el estado del arte del procesamiento de IDs, en el que sus módulos funcionan íntegramente con técnicas de aprendizaje profundo, y que, además, asume que los IDs han sido capturados en ambientes sin restricciones, en donde el grado de orientación del ID, su ubicación en la imagen, los fondos, la perspectiva y la iluminación, no son homogéneos y pueden cambiar dependiendo de la fotografía que capture un usuario en particular.

En la etapa de localización del ID, se utiliza el algoritmo de detección YOLOv4 [5]; para la fase de segmentación, se emplea la arquitectura U-Net [19], y en la fase de alineación, se hace uso del sistema RotNet [20]. También es importante señalar que, por primera vez en el estado del arte, en este artículo se utilizan YOLOv4 y RotNet para el procesamiento de IDs.

Otra de las contribuciones esenciales de este trabajo, es que durante el entrenamiento de los modelos de segmentación y orientación, se generan imágenes sintéticas especialmente diseñadas para asemejarse a las fotografías de los IDs capturados por los usuarios, lo cual incrementa drásticamente el tamaño de la base de datos y ayuda a disminuir el fenómeno principal que aqueja al rendimiento de los modelos de aprendizaje profundo: el sobreajuste.

El resto del artículo se organiza de la siguiente manera. En la Sección 2, se ofrece un resumen de los trabajos del estado del arte referentes al procesamiento de IDs. La teoría preliminar relacionada con el enfoque del aprendizaje profundo y los algoritmos de este tipo que forman parte del sistema de procesamiento de IDs propuesto, se presentan en la Sección 3.

La descripción del flujo de procesamiento del sistema se expone en la Sección 4. La información respecto a los resultados experimentales, se encuentra en la Sección 5. Finalmente, en la Sección 6 se brindan las conclusiones obtenidas de esta investigación, así como el trabajo futuro a realizar.

2. Trabajos relacionados

En el estado del arte, la implementación de métodos de aprendizaje profundo en los sistemas de análisis y procesamiento de IDs, es un ámbito de documentación escasa, y la mayoría de los trabajos de investigación, se concentran en abordar los inconvenientes en la captura de IDs mediante técnicas tradicionales de procesamiento de imágenes.

Fang et al. [11], propusieron un sistema de identificación de IDs de China que determina el ángulo de inclinación de los IDs por medio de la Transformada de Hough, aunque no se especifica si a través de este enfoque, la corrección en la alineación se realiza sólo si los IDs tienen un ángulo de orientación en un rango determinado o si el ángulo de inclinación es despreciable.

En el trabajo de Xu et al. [21], se presentó un proceso de reconocimiento para ambos lados de los IDs de China usados en los experimentos, el cual incluye la alineación del ID considerando ángulos de inclinación menores a 45°. Para alinear correctamente el ID, se emplea la Transformada de Hough, seguida de una transformación afín.

Posteriormente, la etapa de localización de la región del ID se lleva a cabo a través de la detección de ciertos elementos del ID mediante el algoritmo de AdaBoost, el cual es alimentado por características de tipo Haar. Vale la pena destacar que, en los dos trabajos referenciados con antelación, no se ofrece una descripción cuantitativa de los resultados experimentales alcanzados en la detección y alineación de los IDs.

En [3], Attivissimo et al. presentaron un prototipo que, entre otras operaciones, localiza IDs italianos con base en un algoritmo que ajusta iterativamente los vértices de un rectángulo de referencia hasta que estos se acoplan con los vértices del ID, el cual debe estar posicionado en la parte central de la imagen.

Los resultados experimentales indican que los vértices de los IDs fueron detectados con un tasa de precisión de 68.57 %. Castelblanco et al. [7], propusieron un sistema de análisis de IDs de Colombia que incluye una etapa de segmentación de IDs con rotaciones en el rango de [0°, 359°] y que se realiza por medio de la arquitectura U-Net.

Después, el ID es recortado y alineado usando una regresión lineal y una transformación en la perspectiva. El rendimiento del modelo de segmentación presentó una exactitud de 98.41 % y un valor en el índice de Jaccard de 0.98. Por otra parte, el método de recorte alcanzó una exactitud de 88.54 %.

3. Teoría preliminar

En esta Sección, se establece un contexto teórico pertinente para la comprensión del funcionamiento del sistema propuesto. Se describen los principios básicos del aprendizaje profundo y de un algoritmo de aprendizaje profundo denominado red neuronal convolucional (CNN, del inglés Convolutional Neural Network), así como de los modelos YOLOv4, UNet y RotNet, los cuales fueron construidos a partir de CNNs.

3.1. Aprendizaje profundo

Los algoritmos que operan bajo el paradigma del aprendizaje profundo, brindan la posibilidad de extraer características directamente de los datos crudos de entrada, sin la necesidad de aplicar técnicas de preprocesamiento, de tal forma que, mediante el enfoque del aprendizaje de representaciones y a través de arquitecturas compuestas por múltiples capas que determinan la profundidad del modelo, los métodos de aprendizaje profundo aprenden de forma automatizada capas sucesivas de representaciones cada vez más sofisticadas y significativas de los datos crudos de entrada [9].

3.2. Redes neuronales convolucionales

Una CNN es una arquitectura de red neuronal profunda que procesa los datos de entrada a través de una estrategia basada en el aprendizaje de patrones locales por medio

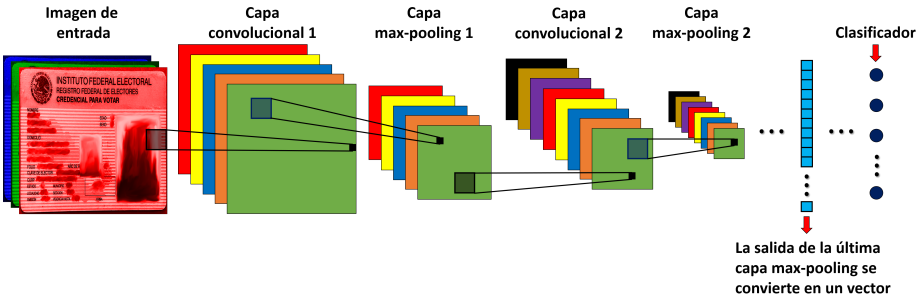


Fig. 1. Ejemplo de una arquitectura clásica de CNN.

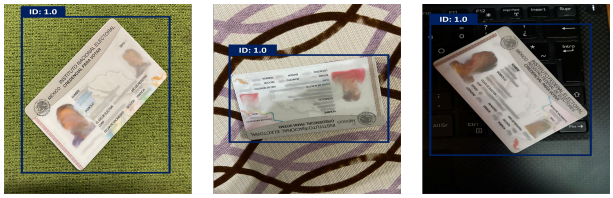


Fig. 2. YOLOv4 localiza y clasifica el objeto de interés contenido en la imagen. En estos ejemplos, se observa que los IDs localizados y delimitados mediante bounding boxes, han sido categorizados en la clase ID, con una precisión del 1.0 (100 %).

de la operación matemática de convolución. Las entradas que son ingresadas a una CNN, se estructuran en múltiples arreglos, como es el caso de las imágenes a color o de las secuencias de lenguaje [14].

Apegándose a los principios de conexiones locales, pesos compartidos, pooling y el uso de una gran cantidad de capas, las CNN's ofrecen robustez ante las variaciones y distorsiones de los datos de entrada, además de una metodología jerárquica espacial para el aprendizaje de patrones [13, 14, 9]. En la Figura 1, se muestra una arquitectura clásica de CNN.

3.3. YOLOv4

You Only Look Once (YOLO) es un sistema de detección de objetos en imágenes propuesto por Redmon et al. [17], el cual actúa como un método de regresión que opera a través de una sola CNN, prediciendo las coordenadas asociadas a la posición de objetos específicos dentro de una imagen.

Estas coordenadas inferidas por YOLO, suelen representarse por medio de cajas delimitadoras, comúnmente conocidas como *bounding boxes*. De forma simultánea a la localización de objetos, YOLO también efectúa la tarea de categorizarlos en sus clases correspondientes (Ver Figura 2). En este trabajo, se hace uso de la cuarta versión de YOLO, desarrollada por Bochkovskiy et al. [5], y denominada YOLOv4.

Fig. 3. Ante una imagen de entrada, U-Net produce su mapa de segmentación correspondiente.

3.4. U-Net

U-Net es una arquitectura de CNN que fue diseñada por Ronneberger et al. [19], para ser aplicada en problemas biomédicos relacionados con la segmentación semántica de células y estructuras neuronales.

Por segmentación semántica, se entiende aquella tarea dedicada a la clasificación de los píxeles de una imagen en ciertas categorías. Por ejemplo, en el caso de la segmentación de fotografías de IDs, los píxeles pueden asignarse ya sea a la clase ID o a la clase fondo.

La estructura de la arquitectura U-Net, que es una CNN completa que carece de capas densamente conectadas, se compone de una ruta de contracción y una ruta de expansión.

La ruta de contracción se conforma por una red neuronal convolucional típica, mientras que la ruta de expansión se constituye por un conjunto de capas convolucionales sucesivas donde se aplica la operación de convolución ascendente, que genera como salida un mapa de segmentación con la misma resolución de la imagen de entrada (Ver Figura 3).

3.5. RotNet

Desarrollado por Daniel Sáez [20], RotNet es un sistema que predice el ángulo de orientación de una imagen por medio de CNNs. En contraste con otras propuestas basadas en CNNs que abordan esta tarea desde la óptica de un problema de regresión [12, 4], Sáez enmarca la tarea de predicción del ángulo de inclinación dentro del ámbito de los algoritmos de clasificación, de tal manera que, ante una imagen de entrada que presente cierto ángulo de rotación, RotNet la clasifica en alguna de las 360 clases que representan cada uno de los posibles ángulos de orientación de dicha imagen.

Como parte de la configuración de RotNet, uno de sus aspectos medulares consiste en la aplicación de la estrategia de transferencia de aprendizaje mediante el modelo ResNet50, incrementando así la eficiencia y la eficacia del proceso de entrenamiento.

Otra de los propiedades fundamentales en el funcionamiento de RotNet, es que durante la etapa de entrenamiento, se generan imágenes sintéticas con rotaciones artificiales en el rango de $[0, 359] \in Z$, aumentando en gran medida la base de datos y refinando la capacidad de generalización del modelo ante las imágenes de prueba, lo que reduce ostensiblemente el riesgo de que el modelo se ajuste en exceso al conjunto de entrenamiento.

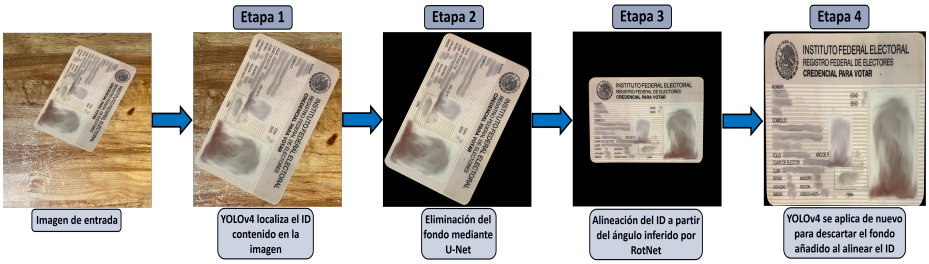


Fig. 4. Flujo de procesamiento del sistema propuesto.

4. Sistema de procesamiento de IDs

A continuación, se brinda una descripción pormenorizada de las etapas de operación del sistema de procesamiento de IDs propuesto. Este sistema funciona de acuerdo a las predicciones que se obtienen al desplegar los modelos de localización (YOLOv4), segmentación (U-Net) y orientación (RotNet). En la Figura 4, se expone el flujo de procesamiento del sistema.

4.1. Etapa 1: Localización del ID

Una vez que el sistema recibe la fotografía del ID capturada por el usuario, el primer proceso que se ejecuta es el de la localización de la región correspondiente al ID a través de YOLOv4. Dado que YOLOv4 proporciona como salida las coordenadas que permiten ubicar el ID, estas coordenadas son usadas para recortar el ID de la fotografía original mediante la función crop de la librería Pillow, de Python (Ver Sección 5 para más detalles técnicos de la implementación del sistema), lo que ocasiona que gran parte del fondo presente en la fotografía sea removido, facilitando así el entrenamiento del modelo de segmentación en la Etapa 2.

4.2. Etapa 2: Segmentación del fondo

En esta fase del sistema, U-Net recibe como entrada la imagen del ID recortado, y genera como salida su mapa de segmentación correspondiente (Ver Subsección 3.4). Luego, mediante una multiplicación a nivel de píxeles, el mapa de segmentación se aplica a la imagen del ID recortado, produciéndose así una imagen resultante de alto contraste entre el fondo y el ID, ya que la porción del fondo original que se venía acarreando desde la Etapa 1, es reemplazada por píxeles que representan el color negro, permitiendo que el ID quede resaltado.

4.3. Etapa 3: Alineación del ID

Después de que la imagen del ID recortado queda libre de fondo, el sistema procede a predecir el ángulo de inclinación del ID por medio de RotNet. Posteriormente, este ángulo es usado para alinear el ID con base en la función rotate_bound de la librería imutils, de Python.

Al alinear el ID, no sólo se cambia el ángulo de orientación del ID, sino de la imagen completa, lo que provoca que los bordes de esta imagen sean rellenados automáticamente con píxeles de color negro (Ver Figura 4).

4.4. Etapa 4: Segunda localización del ID

En la etapa final del sistema, se vuelve a localizar el ID debido a los bordes que se añaden en la etapa previa. Por lo tanto, YOLOv4 se emplea de nuevo para posicionar el ID. La ocurrencia de esta etapa tiene una condición: el ID de entrada debe tener cierta rotación, de lo contrario, basta con que se desplieguen las primeras tres etapas del sistema de procesamiento de IDs.

5. Resultados experimentales

En las Subsecciones siguientes, se detallan las características de la configuración experimental dispuesta para entrenar, validar y evaluar los modelos de localización, segmentación y alineación que componen el sistema de procesamiento de IDs propuesto, así como un análisis del funcionamiento integral del mismo.

En lo que se refiere a la implementación del sistema, esta se llevó a cabo en la plataforma de Google Colaboratory, la cual permite programar en el navegador a través del lenguaje de Python, además de que da acceso a GPUs genéricos para el entrenamiento de modelos de aprendizaje profundo.

Bajo el entorno de Google Colaboratory, los modelos de aprendizaje profundo que forman parte del sistema propuesto, se construyeron mediante Keras, que es una API de alto nivel de Tensorflow.

5.1. Bases de datos

Las imágenes de IDs usadas en este trabajo, corresponden a fotografías frontales de credenciales de elector de los Estados Unidos Mexicanos. En total, se recolectaron 2,000 imágenes con el consentimiento de los participantes, quienes realizaron las capturas con sus smartphones, siguiendo un protocolo con una única restricción: el ID debía aparecer lo más alineado posible.

Otras condiciones de captura, tales como la resolución, la iluminación, la perspectiva o los fondos, podían tener cualquier variación. Además de la adquisición de IDs, también se procedió a recolectar una base de datos de imágenes de fondos que contienen diversas texturas, objetos y fluctuaciones en la iluminación.

Las imágenes de fondos se recopilaron manualmente y también se obtuvieron de otras fuentes [6, 15], lo que permitió construir una base de datos de 26,534 fondos.

Los IDs alineados de las 2,000 imágenes recopiladas, fueron recortados manualmente, y a cada uno de ellos se les agregó un canal alfa, con la intención de que fuera posible generar imágenes sintéticas a partir de ellos, siguiendo un procedimiento que denominamos metodología de aumentación de datos: el ID recortado y con canal alfa puede rotarse libremente en el rango de $[0, 359] \in Z$, para después pegarlo en cualquier posición de un fondo, donde la resolución de este fondo se

ajusta a las dimensiones de la entrada especificada por el modelo de aprendizaje profundo en cuestión, de manera tal que, al pegar el ID, este se adapta al tamaño del fondo redimensionado.

Finalmente, la imagen puede ser sometida a alteraciones en el brillo, aunque en el caso en que el ID se coloque sobre un fondo negro, como ocurre con las imágenes de entrada que requiere RotNet (Ver Subsección 3.5), el cambio en el brillo se aplica únicamente en el ID. Usando esta metodología de aumentación de datos, es factible incrementar fácilmente el tamaño de la base de datos, ya que se puede producir una amplia gama de variaciones por rotación, traslación, iluminación y el elevado número de imágenes de fondos con que se cuenta.

El conjunto de 2,000 imágenes de IDs recortados se particionó en tres conjuntos de datos: entrenamiento, validación y prueba. En el conjunto de entrenamiento, se colocaron 1,200 imágenes, mientras que, tanto al conjunto de validación como al de prueba, se les asignaron 400 imágenes. Las imágenes de los IDs que conforman estos conjuntos, se seleccionaron de forma aleatoria.

A continuación se muestra cómo, en función de estos tres conjuntos de imágenes, a los que denotaremos como las versiones originales, se construyeron las bases de datos para el entrenamiento, validación y prueba de los modelos de aprendizaje profundo que forman parte del sistema de procesamiento de IDs.

5.2. Modelo de localización

Para entrenar de forma adecuada a YOLOv4, fue preciso construir nuevos conjuntos de entrenamiento, validación y prueba, a partir de sus versiones originales y empleando la metodología de aumentación de datos diseñada. De esta forma, se generó un conjunto de entrenamiento con 10,000 imágenes sintéticas, mientras que los conjuntos de validación y prueba, se constituyeron con 2,000 imágenes sintéticas cada uno. Estas imágenes se redimensionaron a un tamaño de 416×416.

Debido a que originalmente los IDs fueron recortados, es sencillo generar las etiquetas para el entrenamiento, validación y evaluación de YOLOv4, las cuales consisten en definir la clase y las coordenadas de la ubicación del ID. En el problema que se está abordando, sólo se tiene la clase ID, y como el ID recortado (y luego rotado) se pega sobre algún fondo, usando para ello coordenadas aleatorias, estas mismas coordenadas son las que se especifican en la etiqueta del ID en cuestión.

El entrenamiento de YOLOv4 se realizó de acuerdo a las especificaciones por default establecidas por su creador [18]. Así pues, el entrenamiento se desarrolló durante 6,000 épocas, utilizando como métricas de evaluación del rendimiento a las funciones *IoU (intersection over union)* y *mAP (mean average precision)*. El modelo con el mejor rendimiento con respecto al conjunto de validación, se alcanzó en la época 5,000, obteniendo un valor de *IoU* de 0.9721 y un 100 % en *mAP*.

Al evaluar este modelo en el conjunto de prueba, se alcanzó un valor de *IoU* de 0.9634 y un 100 % en *mAP*.

5.3. Modelo de segmentación

La arquitectura U-Net se entrenó con base en un enfoque de generación de imágenes sintéticas durante la propia etapa de entrenamiento, incrementando en gran medida el conjunto de entrenamiento y haciendo que el modelo no se sobreajustara, pues ninguna de las imágenes sintéticas era igual entre sí, ya que se diseñaron siguiendo la metodología de aumentación de datos propuesta.

Las imágenes sintéticas producidas, simulan a las imágenes que se obtienen de la Etapa 1 del sistema de procesamiento de IDs (Ver Subsección 4.1): el ID aparece localizado en el centro de la imagen y una gran porción del fondo es eliminada.

Asimismo, de forma adjunta a cada imagen sintética, se genera su etiqueta, que en este caso corresponde a la imagen de un mapa de segmentación (Ver Subsección 3.4).

En este sentido, el entrenamiento de U-Net se inició con los 1,200 IDs recortados que conforman el conjunto de entrenamiento original, y en función de estos IDs, se fueron generando imágenes sintéticas durante el entrenamiento, tal como se explicó previamente.

Por consiguiente, debido a que se requirieron 100 épocas para entrenar el modelo de segmentación, en cada una de estas épocas, los pesos del modelo se ajustaron con 1,200 imágenes de IDs sintéticas y distintas unas de otras, por lo que, al final de la época 100, el modelo terminó entrenándose con 120,000 imágenes completamente diferentes entre sí.

En relación a los conjuntos de validación y prueba, ambos se construyeron a partir de sus versiones originales, conformándose cada conjunto con 2,000 imágenes de IDs junto con sus respectivos mapas de segmentación, los cuales también se generaron sintéticamente, procurando que simularan las propiedades de las imágenes resultantes de la Etapa 1 del sistema de procesamiento de IDs. Tanto las imágenes sintéticas de IDs como sus respectivos mapas de segmentación, se adecuaron a una resolución de 512 $\times$ 512.

Para el entrenamiento de U-Net, como optimizador se usó la función Adam con una tasa de aprendizaje de 0.0001 y como función de pérdida, se empleó binary_crossentropy.

Para darle seguimiento al rendimiento del modelo, se utilizaron las métricas *accuracy* y el *coeficiente de Dice*, donde la primera aporta certeza acerca de la categorización correcta de cada píxel y la segunda proporciona información sobre la precisión en el traslape entre el mapa de segmentación objetivo y el que es inferido por el modelo.

El mejor desempeño del modelo en relación al conjunto de validación, se obtuvo en la época 94, con un *accuracy* de 98.30 % y un 0.9889 de *coeficiente de Dice* (Ver Figura 5). Después de evaluar este modelo en el conjunto de prueba, el *accuracy* obtenido fue de 97.85 %, mientras que el valor del *coeficiente de Dice* llegó a 0.9820.

5.4. Modelo de alineación

De forma similar a la estrategia aplicada para entrenar U-Net y usando nuestra metodología de aumentación de datos, para el modelo de alineación basado en el sistema RotNet también se generaron imágenes sintéticas durante el proceso de

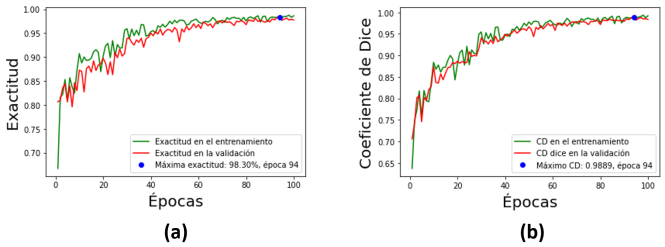


Fig. 5. (a) Gráfica de la exactitud en el entrenamiento y la validación. (b) Gráfica del coeficiente de Dice en el entrenamiento y la validación.

entrenamiento, las cuales simulan las características de las imágenes derivadas de la Etapa 2 del sistema de procesamiento de IDs (Ver Subsección 4.2), donde el ID se ubica en el centro de la imagen y el fondo que aparece es de color negro.

Al rotar la imagen de entrenamiento sintética, el ángulo de rotación se guarda como la etiqueta de dicha imagen rotada.

Por su parte, los conjuntos de validación y prueba se diseñaron partiendo de sus versiones originales, constituyéndose cada conjunto con 2,000 imágenes de IDs generadas sintéticamente junto con sus etiquetas asociadas, donde estas imágenes sintéticas tienen una estructura similar a las de las imágenes que se obtienen de la Etapa 2 del sistema de procesamiento de IDs. La resolución de las imágenes sintéticas generadas para este modelo, se ajustó a un tamaño de 224×224.

Es importante precisar que, aunque de forma predeterminada RotNet puede rotar las imágenes de entrada durante la fase de entrenamiento, no es capaz de pegar una imagen sobre otra ni de cambiar el brillo de las imágenes, por lo que fue necesario modificar su configuración original para hacer que la generación de imágenes sintéticas se adaptara a nuestra metodología de aumentación de datos, con lo que se incrementaron las capacidades iniciales del funcionamiento del sistema.

El entrenamiento del modelo de alineación se desarrolló durante 100 épocas, empleándose la función SGD (Stochastic Gradient Descent) como optimizador, con una tasa de aprendizaje de 0.01 y un momentum de 0.9.

La función categorical_crossentropy se usó para determinar la pérdida del modelo, y la función angle_error, que es una métrica exclusiva de RotNet, sirvió para monitorear la discrepancia entre los ángulos esperados y los ángulos inferidos por el modelo.

Es en la época 62 donde se obtiene el mejor rendimiento del modelo con respecto al conjunto de validación, alcanzándose un valor de angle_error de 1.5600 (Ver Figura 6).

Una vez que este modelo se evaluó en el conjunto de prueba, el angle_error que se obtuvo fue de 1.5920.

5.5. Despliegue y evaluación del sistema de procesamiento de IDs

Luego de haber entrenado los modelos de localización, segmentación y alineación, estos se usaron para poner en operación el sistema de procesamiento de IDs, tomando como entradas a las imágenes y etiquetas que conforman el conjunto de prueba del modelo de localización (Ver Subsección 5.2).

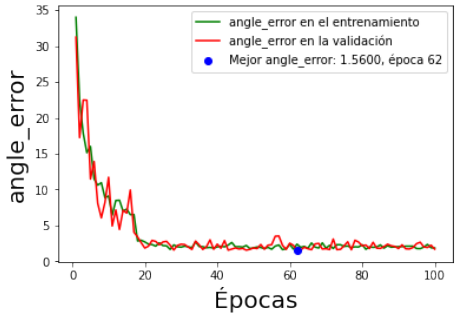


Fig. 6. Gráfica del angle_error en el entrenamiento y la validación.

En consecuencia, estas imágenes fueron procesadas a través de las 4 etapas del sistema propuesto. Como resultado de la etapa 1, el rendimiento obtenido es exactamente el mismo al que se detalla en la Subsección 5.2.

Por su parte, las imágenes recortadas derivadas de la etapa 1, fueron pasadas al modelo de segmentación de la etapa 2, donde se alcanzó un accuracy de 97.27 % y un *coeficiente de dice* de 0.9786.

Las imágenes segmentadas que provienen de la etapa 2, se sometieron al procesamiento del modelo de alineación de la etapa 3, teniendo un desempeño de 1.6015 de acuerdo a la métrica angle_error.

En la etapa 4, se ingresaron las imágenes de los IDs alineados que resultaron de la etapa 3, las cuales tienen bordes de relleno añadidos al ser rotadas, por lo que es necesario ubicar de nuevo el ID, y en esta segunda aplicación del modelo de localización, se obtiene un valor de *IoU* de 0.9659 y un 100 % en *mAP*.

Estos resultados experimentales, indican que el sistema de procesamiento de IDs propuesto en el presente artículo, ofrece un rendimiento elevado ante las variaciones agresivas que comúnmente se presentan en las fotografías de IDs que se ingresan a los sistemas de onboarding digital.

Esta cualidad del sistema propuesto, lo hace apto para ser desplegado en producción, ya que se mejora notablemente la experiencia del usuario, quien no tiene que preocuparse por alinear correctamente la imagen del ID de entrada, colocar el ID en un fondo con textura simple o realizar la captura en un ambiente con iluminación controlada.

6. Conclusiones y trabajo futuro

En este artículo, se introdujo un sistema de procesamiento de IDs basado en modelos de aprendizaje profundo que realiza tres operaciones fundamentales: localización del ID, segmentación del fondo y alineación del ID.

Para el modelo de localización, se empleó el sistema YOLOv4; la tarea de segmentación se realizó a través la arquitectura U-Net, y la corrección en la alineación del ID, se alcanzó gracias al uso del sistema RotNet.

El entrenamiento de los modelos se vio beneficiado por la adopción de una metodología de aumentación de datos que permitió incrementar sustancialmente el número de imágenes disponibles, y, en el caso particular de los modelos de segmentación y alineación, este incremento se llevó a cabo durante la etapa de entrenamiento.

Al evaluar los modelos, los resultados mostraron la viabilidad de desplegar nuestro sistema en un proceso de onboarding digital, ya que este exhibe una considerable robustez ante la mayoría de las variaciones presentes en las fotografías de IDs ingresadas por los usuarios.

Por otro lado, contemplando la posibilidad de que la fotografía del ID posea distorsiones severas en la perspectiva (p. ej. distorsiones afines o proyectivas), se pretende agregar posteriormente un módulo para la aplicación de una matriz de transformación sobre el ID, de tal forma que la imagen resultante facilite el procesamiento de la fase de alineación de nuestro sistema.

La adición de este módulo, sería asimismo benéfica para las etapas subsiguientes del onboarding digital, específicamente, para la tarea de extracción de texto, que es un problema cuyo tratamiento también se tiene planeado como trabajo futuro.

Referencias

1. Arlazarov, V. V., Bulatov, K., Chernov, T., Arlazarov, V. L.: Midv-500: A dataset for identity document analysis and recognition on mobile devices in video stream. Computer Optics, vol. 43, no. 5, pp. 818–824 (2019) doi: 10.18287/2412-6179-2019-43-5-818-824
2. ASOFOM: Digital Onboarding: Definition, characteristics and how it works (2020), https://asofom.mx/2020/12/01/onboarding-digital-solucion-practica-y-omnipresente/
3. Attivissimo, F., Giaquinto, N., Scarpetta, M., Spadavecchia, M.: An automatic reader of identity documents. In: IEEE International Conference on Systems, Man and Cybernetics, pp. 3525–3530 (2019) doi: 10.1109/smc.2019.8914438
4. Baltruschat, I. M., Saalbach, A., Heinrich, M. P., Nickisch, H., Jockel, S.: Orientation regression in hand radiographs: A transfer learning approach (2018) doi: 10.1117/12.2291620
5. Bochkovskiy, A., Wang, C. Y., Liao, H. Y. M.: Yolov4: Optimal speed and accuracy of object detection, (2020) doi: 10.48550/arXiv.2004.10934
6. Burghouts, G. J., Geusebroek, J. M.: Amsterdam Library of Textures (ALOT) (2021), https://aloi.science.uva.nl/public_alot/
7. Castelblanco, A., Solano, J., Lopez, C., Rivera, E., Tengana, L., Ochoa, M.: Machine learning techniques for identity document verification in uncontrolled environments: A case study. In: Mexican Conference on Pattern Recognition. pp. 271–281 (2020) doi: 10.1007/978-3-030-49076-8_26
8. Cath, J.: COVID-19 and the rush to Digital Onboarding. Fintrail (2020), https://www.fintrail.co.uk/news/2020/10/28/covid-19-and-the-rush-to-digital-onboarding
9. Chollet, F.: Deep Learning with Python. Manning Publications Co (2017)
10. Electronic Identification: Digital Onboarding: definition, characteristics and how it works (2021), https://www.electronicid.eu/es/blog/post/digital-onboarding-process-financial-sector/en
11. Fang, X., Fu, X., Xu, X.: ID card identification system based on image recognition. In: 12th IEEE Conference on Industrial Electronics and Applications. pp. 1488–1492 (2017) doi: 10.1109/ICIEA.2017.8283074

12. Fischer, P., Dosovitskiy, A., Brox, T.: Image orientation estimation with convolutional networks. In: German Conference on Pattern Recognition. vol. 9358, pp. 368–378 (10 2015) doi: 10.1007/978-3-319-24947-6_30
13. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE. vol. 86, pp. 2278–2324 (1998) doi: 10.1109/5.726791
14. Lecun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature, vol. 521, no. 7553, pp. 436–444 (2015) doi: 10.1038/nature14539
15. Morales, F.: Complete end-to-end training. Keras (2019), https://keras-ocr.readthedocs.io/en/stable/examples/end_to_end_training.html
16. Narayan, A.: Customer onboarding in a COVID-19 world. Refinitiv (2020), https://www.refinitiv.com/perspectives/financial-crime/customer-onboarding-in-a-locked-down-world/
17. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 779–788 (2016) doi: 10.1109/CVPR.2016.91
18. Redmon, J.: Darknet: Open source neural networks in C (2016), http://pjreddie.com/darknet/
19. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation, (2015) doi: 10.1007/978-3-319-24574-4_28
20. Sáez, D.: Rotnet. guthub (2017), https://github.com/d4nst/RotNet\#readme
21. Xu, J., Wu, X.: A system to localize and recognize texts in oriented id card images. In: IEEE International Conference on Progress in Informatics and Computing (PIC). pp. 149–153 (2018) doi: 10.1109/PIC.2018.8706303

ISSN 1870-4069

La exhibición interactiva: su proceso de diseño con base en inteligencia artificial

Sandra Rodríguez-Mondragón, Manuel Martín Clavé-Almeida,
Luis Jorge Soto-Walls, Oscar Herrera-Alcántara

Universidad Autónoma Metropolitana,
México

{sandra.rgz.mondragon, luissotowalls,
mclavealmeida}@gmail.com, oha@correo.azc.uam.mx

Resumen. La presente investigación tiene la intención de dar a conocer una metodología para el "Diseño de la exhibición interactiva", y la aplicación de inteligencia artificial (IA). Como parte inicial se abordan algunas metodologías para el diseño, se realiza una propuesta para el diseño de la exhibición interactiva y se exploran algunas áreas de aplicación de inteligencia artificial. Posteriormente se muestran algunos ejemplos prácticos de exhibición interactiva basada en técnicas de IA. Finalmente, se describe cómo interviene y que técnicas de IA son comúnmente aplicables al diseño de la exhibición interactiva.

Palabras clave: Diseño, exhibición interactiva, tecnologías, multimedia, virtual, gestión de datos, sistemas expertos y aprendizaje automático.

The Interactive Exhibiting: Your AI-Powered Design Process

Abstract. The present investigation intends to present a methodology for the "Design of the interactive exhibition", and the application of artificial intelligence (AI). As an initial part, some design methodologies are addressed, a proposal for the design of the interactive exhibition is made and some areas of application of artificial intelligence are explored. Subsequently, some practical examples of interactive exhibition based on AI techniques are shown. Finally, it describes how it intervenes and which AI techniques are commonly applicable to the design of the interactive exhibition.

Keywords: Design, interactive exhibition, technologies, multimedia, virtual, data management, expert systems and machine learning.

1. Aspectos del diseño

1.1. Proceso de diseño de productos

El diseño de productos implica un proceso que generantemente no es lineal, en el cuadro A, se presenta una breve historia (1962-1992) descriptiva de algunos de estos métodos de diseño. Un ejemplo de modelo para diseño de productos es el Modelo General del Proceso de Diseño (MGPD) [1] de la UAM – Azcapotzalco[1], que se compone de cinco fases:

1. Caso: fenómenos sociales desde la interdisciplinariedad, se deriva una propuesta para el diseño.
2. Problema: estudio del fenómeno desde los objetivos, las condiciones teóricas de una disciplina propia del diseño.
3. Hipótesis: desarrollo de la máxima cantidad de alternativas para los requerimientos del problema.
4. Proyecto; se divide en dos partes
    a. 1ra. se desarrollan planos, maquetas y simulaciones.
    b. 2da. se confrontan con lo propuesto en la hipótesis.
5. Realización: La producción material de la forma propuesta.

Otro ejemplo de modelo de proceso de diseño es el de Navarrina (1987)[2], el cual tiene una estructuración operativa de relaciones y transmisión de información entre los diversos niveles. Aquí se pueden identificar tres etapas de diseño donde es posible implementar IA: 1) modelado y parametrización; 2) calculo y; 3) mejora del producto (ver figura 1).

1.2. La exhibición interactiva

De acuerdo con Beatriz Abella [3], existen diversos tipos de exposición, dentro de esta clasificación se encuentra la interactiva:

> *Atendiendo a las categorías o caracteres desde la perspectiva del público receptor, pueden calificarse como didácticas y no didácticas. […] Y, en "otras categorías", incluye las interactivas (que pueden modificar su presentación según la percepción que el diseñador tenga de la respuesta del espectador), reactiva (la que automáticamente se pone en marcha delante del visitante), dinámica (animadas por medios mecánicos u otros), centrada en el objeto (cuando éste tiene preponderancia sobre cualquier otro medio interpretativo), sistemática (organización de los objetos según un modelo aceptado), temática (parte de una línea argumental y recurre a los objetos para ilustrar el tema) y participativa (busca involucrar al visitante a través del sentido del tacto). Las tipologías no son excluyentes y a menudo nos encontramos con exposiciones donde se combinan categorías.*[3]

[1] Manual de diseño industrial. p. 36.
[2] Navarrina, Fermín. Una metodología general para optimización estructural en diseño asistido por el ordenador". p. IV 14a
[3] Abella, Beatriz. Diseño de exposiciones. Concepto, instalación y montaje. p. 2.

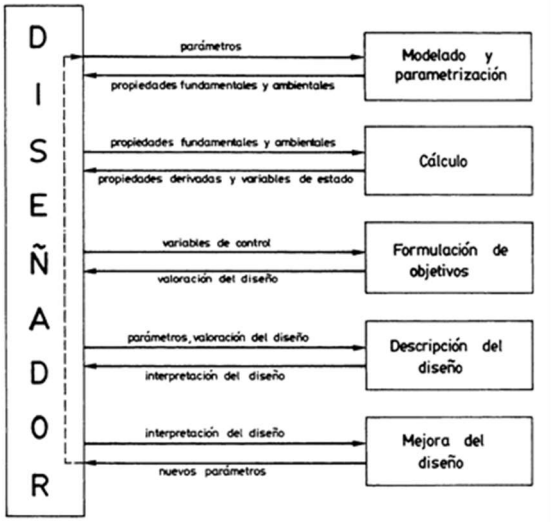


Fig. 1. Estructuración operativa del proceso de diseño. Navarrina, F., 1987. [2].

Cuadro A. Historia de los métodos del diseño (1962-1992).

Autores Representativos	Descripción
Asimow (1962)	Dos etapas: • Planeación y Morfología • Diseño detallado.
Jones (1963)	La intuición y los aspectos no-racionales tienen el mismo rol que los lógicos y los procedimientos sistemáticos.
Archer (1963),	Listas de chequeo (¡más de 229 items!), para verificar tres fases: • Análisis. • Creatividad • Ejecución.
Alger y Hays (1964)	Énfasis en la valoración de alternativas del proyecto.
Alexander (1964)	Análisis riguroso del problema. Adaptación del programa de diseño al problema específico División del problema complejo en subgrupos de problemas.
Luckman (1967)	Método AIDA, tres fases: • Análisis • Síntesis • Evaluación. No son lineales sino interactivas.
Levin (1966)	Caracterización de propiedades de sistemas. Relación causa – efecto (controlables y no controlables)
Gugelot (1963) Burdel (1976)	Información sobre necesidades del usuario. Aspectos funcionales Exploración de posibilidades funcionales Decisión Detalle: cálculos, normas, estándares. Prototipo.
Jones (1970)	No es un método, pero expone dos tendencias: Caja negra: la parte más importante del diseño se realiza en el subconsciente del diseñador, no puede ser analizada. Caja de cristal: todo el proceso se hace transparente.
Jones (1971) Alexander Tudela	Contracorriente: Los métodos de diseño destruyen la estructura mental del diseñador. Se produce una abolición de la racionalidad funcional.
Manuri (1974)	No es correcto proyectar sin método. Indica que primero se hace un estudio sobre materiales y procesos, que alimentan la generación de ideas.
Maldonado (1977) Dorfles (1977)	Deben integrarse al proceso de diseño los factores: funcionales, simbólicos o culturales, de producción.
Bonsiepe (1985)	Dos métodos: Reducción de la complejidad de Alexander. Búsqueda de analogías o Sinéctica de Gordon.
Quarante (1992)	Para cada problema hay un método. No universalidad de métodos.

Una descripción basada en la definición de Mc. Lean (1993) [4], la encontramos en la Red de Popularización de la Ciencia y la Tecnología en América y el Caribe (RPCTAC):

> *Las exhibiciones interactivas son aquellas en las cuales "el visitante puede conducir actividades, recolectar evidencia, seleccionar opciones, formar conclusiones, probar*

> *habilidades, proporcionar insumos y, de hecho, alterar una situación basada en un insumo" (Mc. Lean, 1993). Así, una buena exhibición realmente interactiva personaliza la experiencia para el visitante.*[4]

Así, las exhibiciones interactivas tienen el objetivo de que el espectador interactúe sensorialmente con cualquier tema de la muestra por medio de los sentidos: la vista (la visión), el oído (la audición), el olfato, el gusto y el tacto; parte de esta interacción está dirigida hacia una experiencia personal.

1.3. Diseño de exhibición interactiva

Si bien el proceso de diseño que se aplique depende de problema de diseño que se enfrente, en este caso se puede adoptar cualquier metodología de diseño que se desee, sin embargo, por el tipo de problemáticas que se enfrenan al diseñar una exhibición interactiva implica considerar ciertas restricciones en la toma de decisiones en el proceso de desarrollo tales como los aspectos ergonómicos, medioambientales, de sustentabilidad, además de incluir el factor de discapacidad en el usuario, así estaremos hablando de un modelo incluyente donde uno de los requerimientos principales, es la condición de discapacidad del usuario.

Regularmente los sistemas de exhibición interactiva están ligados a las Tecnologías de Información y Comunicaciones (TIC'S)[5], dado que por medio de las tecnologías es posible identificar el factor de interacción con el usuario e incluso medirlo de forma precisa.

De acuerdo con Jorge Padilla[6]:

> *La importancia de la interacción en las exhibiciones como un requisito básico para el aprendizaje y la comprensión […] Se ha propuesto que la "atractividad" y el potencial de aprendizaje en las exhibiciones interactivas se realiza con cuatro rubros:*
>
> *a. Curiosidad y motivación intrínseca*
>
> *b. Modos múltiples de aprendizaje*
>
> *c. Juego y exploración en el proceso de aprendizaje*
>
> *d. Conocimientos y modelos mentales previos del usuario*

Tomando estos preceptos se debe contemplar que difícilmente es posible satisfacer todas las necesidades de las personas con discapacidad, puesto que en cuestiones sensoriales de percepción cuando existe una discapacidad, ésta se ve disminuida o anulada, por lo que se sugiere priorizar el diseño en términos de acceso e ir ponderando el canal de comunicación, con base en el tipo de información a transmitir.

En el diseño de una exhibición interactiva, se deben contemplar, de acuerdo con Héctor Miranda[7], al menos los siguientes cuatro elementos [5]:

---

[4] Padilla, Jorge y RPCTAC. Diseño, construcción y operatividad de exhibiciones interactivas. p.1 Recuperado el 20/06/18 de: https://www.redpop.org/diseo-construccin-y-operatividad-de-exhibiciones

[5] TIC'S, nos referimos a un grupo diverso de prácticas, conocimientos y herramientas, vinculados con el consumo y la transmisión de la información y desarrollados a partir del cambio tecnológico vertiginoso que ha experimentado la humanidad en las últimas décadas, sobre todo a raíz de la aparición de Internet. Recuperado el 08/04/21 de: https://concepto.de/tics/#ixzz6rbBa3cHI

[6] Padilla, Jorge y RPCTAC. Diseño, construcción y operatividad de … op. cit. p.2

[7] Miranda, Héctor. Diseño de exhibiciones interactivas. p.5-17

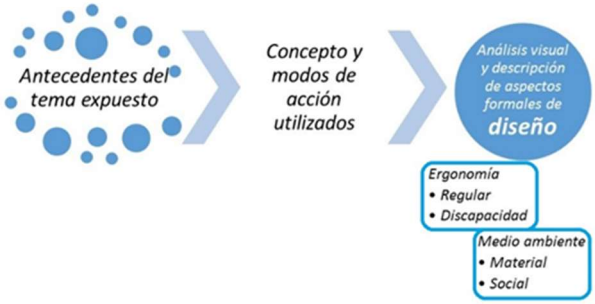


Fig. 2. Proceso de diseño de exhibición interactiva; autoría propia basada en el modelo de Héctor Miranda.

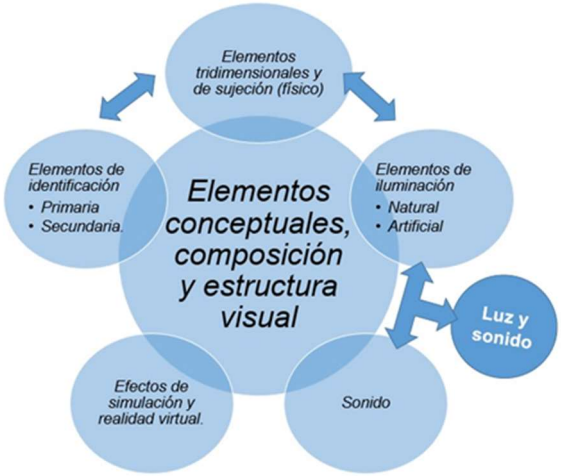


Fig. 3. Elementos conceptuales, composición y estructura visual; autoría propia basada en el modelo de Héctor Miranda.

1. *Antecedentes del tema expuesto.*
2. *Concepto museográfico y modos de acción utilizados.*
3. *Análisis visual y descripción de aspectos formales de diseño.*
4. *Elementos conceptuales, composición y estructura visual.*
    a. *Elementos tridimensionales.*
    b. *Elementos de sujeción.*
    c. *Elementos de iluminación (intensidad, filtros, interactividad con el entorno, interacción con los objetos).*
        - *Natural, de sol.*
        - *Artificial, fluorescente (focos o lámparas), luz incandescente (bombillas 120 v.), halógena, led.*
    d. *Sonido.*
    e. *Luz y sonido.*

Cuadro B. Tecnologías de la exhibición interactiva. Resumen basado en Ávila[8].

Tecnología	Función
OBJETOS TANGIBLES	Se refiere a todos aquellos elementos u objetos que el usuario puede tocar y percibir de manera precisa al interactuar con ellos.
PANTALLA TÁCTIL	Son pantallas que permiten al usuario interactuar directamente con la interfaz, sin necesidad de un elemento intermediario, su tamaño y resolución puede variar según se necesite.
PANTALLAS MÚLTIPLES	También llamado multipantalla o multi-monitor, es el uso de múltiples dispositivos de visualización para aumentar el área de visión e interacción disponible en una interfaz.
MESA INTERACTIVA	Similar a una pantalla táctil de grandes dimensiones. Se caracteriza por permitir dos o más puntos de interacción simultáneamente, así como la posibilidad de dividirse en dos o más áreas de visualización e interacción distintas.
MAPEO DE PROYECCIÓN	También conocido como vídeo *mapping*. Es el uso de proyectores para desplegar animaciones o imágenes sobre superficies reales y en ciertos casos permitir la interacción directa del usuario, simulando una superficie táctil.
REALIDAD AUMENTADA	Esta tecnología permite al usuario visualizar parte del mundo real a través de un dispositivo que genera y agrega información digital, es decir, una parte virtual aparece en la realidad. Los elementos físicos tangibles se combinan con elementos virtuales, creando una realidad aumentada en tiempo real.
REALIDAD VIRTUAL	Esta tecnología genera un entorno de escenas u objetos de apariencia real dentro de un espacio virtual. Usualmente este entorno crea en el usuario la sensación de estar inmerso en él, sin posibilidad de visualizar el mundo real. En ciertos casos, esta tecnología puede mezclarse con estímulos externos para generar una experiencia inmersiva.
REALIDADES MIXTAS	*Mixed Reality* es una variación de la Realidad Aumentada que se concentra en colocar objetos totalmente digitales que interactúen con el entorno del usuario, permitiendo a los usuarios ver y manipular dichos objetos.

f. Efectos de simulación y realidad virtual.

g. Organización efectos: ritmos, simetrías, proporciones, oposiciones, direcciones, etc. (elementos compositivos).

h. Elementos de identificación.

- *Comunicación primaria: semánticos, sintácticos, pragmáticos; visuales: puntos, líneas, planos, volumen y color.*
- *Comunicación secundaria o efectos comunicativos: miméticos, ornamentales, expresivos, emblemáticos, inventados.*

[8] Ávila, José. Diseño de una exhibición interactiva para un museo en Noruega. p. 37-47

Cuadro C. Tecnologías combinadas.

Tecnología	Función
PANTALLAS + PANTALLA TÁCTIL	En estos casos existe una separación entre el sistema de interacción directa (pantalla táctil) y la visualización de esta (pantalla sencilla).
PROYECCIÓN + PANTALLA TÁCTIL	Al igual que en la combinación de pantallas sencillas, esta combinación busca separar el sistema de interacción con el sistema de visualización, con la diferencia de que la proyección puede variar el tamaño según se necesite, además de mantener la tecnología del mapeo.
OBJETOS TANGIBLES + PROYECCIÓN	Esta es una de las soluciones más interesantes en cuanto a tecnología e interacción. El usuario es capaz de interactuar con objetos físicos para visualizar, variar o interactuar con la información proyectada.

En cuestiones de diseño, además se deben contemplar aspectos ergonómicos y medio ambientales (ver figura 2).

En el diseño de elementos de composición y estructura visual existe interacción entre elementos, así, por ejemplo, la iluminación puede operar en conjunto con el sonido generando efectos visuales de estimulación sensorial; los objetos en relación con la luz también modifican su percepción visual; esto se muestra a continuación (ver figura 3).

El Centro de Ciencias Aplicadas y Desarrollo Tecnológico (CCADT) de la Universidad Nacional Autónoma de México (UNAM), también propone un modelo metodológico para el diseño de la Experiencia del Usuario en Espacios Interactivos (EUEI) que consisten en cinco pasos: 1) Planteamiento; 2) Diseño; 3) Realización; 4) Pruebas y evaluación; 5) Cierre; 6) Retroalimentación y mantenimiento (ver figura 4).

2. Recursos

2.1. Tecnologías

De acuerdo con Ávila [7] alguna de las tecnologías que se aplican a la exhibición interactiva son las que se muestran a continuación (ver cuadro B). De estas tecnologías también se puede realizar la producción de la exhibición interactiva con una combinación de ellas y algunos ejemplos se muestran a continuación (ver cuadro C).

2.2. Áreas de trabajo de IA

Dado que la exhibición interactiva está basada en interfaces que hacen uso de bases de datos y gráficos, en el proceso de diseño de estas se pueden intervenir a partir de determinar el o los canales de comunicación con el usuario. Como ya se mencionó la interacción se puede generar por medios físicos mecánicos, robotizados y/o tecnológicos tales como las pantallas táctiles o una combinación de ellos.

Ahora bien, para identificar la forma de intervenir los sistemas de exhibición interactiva, cabe revisar algunas áreas de trabajo de IA, y es conveniente aclarar que

Cuadro D. Áreas de trabajo de IA.

Básicas	Específicas
Representación del conocimiento	Planificación de tareas
	Tratamiento del Lenguaje Natural
	Razonamiento Automático
Resolución de problemas, búsqueda	Sistemas Basados en el Conocimiento
	Percepción
	Aprendizaje Automático
	Agentes autónomos

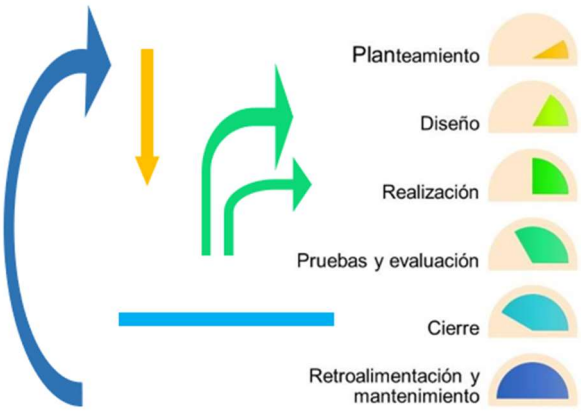


Fig. 4. Modelo metodológico para el diseño de la Experiencia del Usuario en Espacios Interactivos; autoría propia basado en G. De la Cruz [6].

ello no es limitativo para los desarrolladores, pero puede dar la pauta para iniciar el proceso creativo (ver cuadro D).

3. Casos

3.1. Muro multimedia

Este proyecto desarrollado por Hoch Drei GmbH & Co.KG para Herrenknecht AG en 2016 (ver figura 5).

> *En un "muro multimedia" interactivo de 3,4 metros de ancho y 1,3 metros de alto, los visitantes pueden acceder a información sobre más de 300 proyectos diferentes de Herrenknecht AG. De esta manera, el muro multimedia presenta la impresionante gama de proyectos de la empresa. Detrás de cada una de las imágenes de vista previa en constante movimiento hay más imágenes del proyecto, una descripción textual y una hoja de datos de las máquinas utilizadas en el proyecto. El usuario puede abrir esta*

Fig. 5. Muro de medios de Herrenknecht.también se pueden abrir como un proyecto cuando los tocas.[10]

Fig. 6. Estación interactiva Herrenknecht.

información de forma intuitiva simplemente tocando las imágenes de vista previa en una de las cinco pantallas y usando gestos. Al mismo tiempo, la exhibición también funciona como una herramienta de presentación para guías. Para ello, la guía utiliza una aplicación para iPad desarrollada para este fin con el fin de buscar proyectos específicos y llamarlos en la pared. Además, la exhibición se muestra con animaciones para efectos de larga distancia en toda el área de la pantalla en modo inactivo. Por ejemplo, con un salvapantallas de texto o una selección de los los 5 proyectos principales que también se pueden abrir como un proyecto cuando los tocas.[9]

[10] Muro de medios de Herrenknecht. Recuperado el 02/04/21 de de https://www.17k.de/projekt/herrenknecht-media-wall/

Cuadro E. Fragmento de Modelos organizacionales [9].

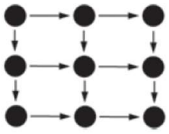
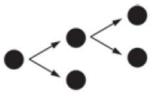
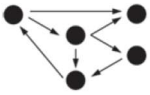
Modelo	Diagrama	Gráficos comunes
Lineal		Lista, gráfico de una sola variable
Tabular		Hoja de cálculo, lista de varias columnas, tabla ordenable, radial. Tabla, gráfica multi-Y, otras gráficas multivariables
Jerárquico		Árbol, listas en cascada, tabla de árbol, mapa de árbol, radial. Tabla, gráfico dirigido
Red de interconexiones		Gráfico dirigido, diagrama de flujo, tabla radial

Fig. 7. XHolo Asistente Virtual.

3.2. Estación interactiva

Este proyecto desarrollado por Hoch Drei GmbH & Co.KG para Herrenknecht AG en 2019, consta de varios módulos interactivos que incluyen: mesa, pantallas táctiles, túnel de simulación y proyector 3D (ver figura 6).

3.3. XHolo Asistente Virtual

Asistente virtual en realidad aumentada, es un personaje generado digitalmente que facilita información mediante voz, reconoce un lenguaje básico y es capaz de interactuar con la persona en diferentes idiomas. Desarrollado por Digalix en 2016, para el estand de Suez en la primera edición del salón *iWater* [8] (ver figura 7).

Como se puede observar en los ejemplos, el potencial de estas aplicaciones es incalculable, puesto que entre otras se pueden contar diversos sectores de las actividades económicas tales como:

- Turístico: hoteles y restaurantes.
- Educación: remota, a distancia y modelos híbridos.
- Exposiciones: comerciales y proyectos de difusión cultural.
- Corporativo: espacios en salas y presentaciones.
- Salud: información, atención, rehabilitación, etc.
- Servicios: desarrollo de software y eventos multimedia.

Por lo antes mencionado, es recomendable identificar una metodología que facilite procesar la información e identificar cómo es que la IA puede apoyar en estos desarrollos.

4. Herramientas de IA

4.1. Gestión de datos

Por medio de IA se puede desde catalogar hasta el análisis de datos, algunas formas de realizar estas operaciones, entre otras se encuentran:

Automatización del proceso de datos por medio de *Deep Learning* (aprendizaje profundo) y en ello es posible: verificar la calidad de la información, integración de datos heredados, desarrollar reglas para la gestión automatizada de datos

En estos procesos, de acuerdo con Tidwell [9], en el manejo de la información visual:

> *Los buenos gráficos de información interactiva ofrecen a los usuarios respuestas a estas preguntas:*
>
> - *¿Cómo se organizan estos datos?*
> - *¿Qué está relacionado con qué?*
> - *¿Cómo puedo explorar estos datos?*
> - *¿Puedo reorganizar estos datos para verlos de manera diferente?*
> - *¿Cómo puedo ver sólo los datos que necesito?*
> - *¿Cuáles son los valores de datos específicos?*

De la misma forma Tidwell sugiere entre otros algunos modelos organizacionales de datos que son de utilidad en el manejo de datos para la visualización en interfaces interactivas (ver Cuadro E).

Así, en la exhibición, la exploración de los datos puede ocurrir por medio de desplazamiento, *zoom*[11], abrir y cerrar puntos de interés y profundizar en puntos de interés, siendo estos modos los más comunes en dispositivos tales como mesas, tabletas y pantallas táctiles.

[11] *Zoom*: Objetivo de distancia focal variable, que mantiene enfocada la imagen al variar la distancia focal. Efecto de acercamiento o alejamiento de la imagen obtenido mediante este objetivo.

4.2. Sistemas expertos

Por otro lado, los sistemas expertos, son la base del desarrollo de los asistentes virtuales puesto que recopilan y simulan el pensamiento de expertos humanos en un área específica del conocimiento.

> *Estos sistemas son capaces de procesar y memorizar información, aprender y razonar en situaciones determinísticas e inciertas, comunicarse con humanos y/o sistemas expertos, hacer decisiones apropiadas y explicar el porqué estas decisiones han sido tomadas [10].*[12]

4.3. Aprendizaje automático (*Machine Learning*)

El aprendizaje automático es una herramienta de gran utilidad en la exhibición interactiva. En coincidencia con Pascual [11]:

> *Normalmente este aprendizaje automático suele ser de dos tipos: supervisado o no supervisado. En el primer caso hay un humano que le dice lo que hace bien o mal. En el no supervisado, es la propia IA la que tiene que aprender a descubrir lo que hace bien y lo que hace mal, en función de unas reglas.*[13]

Una de sus principales aplicaciones es la de reconocer o identificar las preferencias de los visitantes a las exhibiciones y configurar el material exhibido tomando en cuenta estas tendencias.

También es importante mencionar que una de las principales aplicaciones de IA, excediendo su potencial en la exhibición interactiva es en la operación de museos y salas de exhibición donde se identifican aplicaciones para hacer más eficientes sus operaciones y reducir costos. De acuerdo con Michaels [12]:

> *En la forma en que medimos y pronosticamos los comportamientos de los visitantes, en la forma cómo funcionan los sistemas de seguridad y en la forma cómo se administra la energía y otros recursos. (La inteligencia artificial) debería permitir la realización de ahorro en la gestión de nuestros edificios. Esas son a menudo las mayores fuentes individuales de costos operativos en los museos, y el nivel de eficiencia impulsada por la inteligencia artificial podrían transformarse bajo modelos comerciales operativos.*[14]

Así, gracias a la velocidad de procesamiento de información, la inteligencia artificial es una herramienta de gran utilidad para el diseño de la exhibición interactiva.

5. Conclusiones

La inteligencia artificial es una herramienta de gran utilidad en el diseño de exhibición interactiva, se puede decir que la exhibición interactiva es una antes de IA y otra completamente diferente después de IA, porque gracias a IA es posible no sólo hacer personalizada la experiencia del usuario o asistente, sino que permite cerrar el círculo de calidad con la evaluación del usuario.

---

[12] Sosa, María Del Carmen. *Inteligencia artificial en la gestión financiera empresarial.* p.158

[13] Pascual, Juan. *Inteligencia artificial: qué es, cómo funciona y para qué se está utilizando.*

[14] Michaels, Chris. *Cómo utilizan los museos la inteligencia artificial.*

La exhibición interactiva, puede mantenerse en constante evolución y perfeccionamientos gracias a IA, no sólo en los casos que emplean agentes autónomos, si no por medio del almacenamiento de información que los visitantes proporcionan durante su visita, ello permite identificar patrones de comportamiento en los recorridos, captar emociones o reacciones dentro de la muestra e incluso esta interacción permite que los visitantes puedan enriquecer la muestra con observaciones y comentarios.

Otro de los beneficios de utilizar IA en el diseño de la exhibición interactiva, es que ésta permite interactuar por más canales de comunicación que una exhibición tradicional y ello da acceso a personas con discapacidad; por ejemplo para las personas invidentes, se puede hacer accesible la muestra por medio de contenidos de audio; o para personas con debilidad visual, es posible modificar el tamaño de las imágenes o datos proyectados, y esto también es aplicable a la información textual que se puede manipular para que sea más fácil de visualizar.

Por otro lado, las ayudas auditivas pueden desarrollarse en varios idiomas y esto permite que más personas tengan acceso a la información en su propia lengua.

Uno de los beneficios derivados del uso de IA en el diseño de la exhibición interactiva es la programación del consumo de energía de los dispositivos empleados y reducir costos. La mayoría de este tipo de exhibiciones empelan recursos visuales desarrollados en dispositivos tales como pantallas o proyectores, y gracias a IA, es posible programar el comportamiento de estos dispositivos de acuerdo con el flujo de actividad en la exhibición y hacer un mejor uso de dichos recursos.

Referencias

1. Bernal, Roberto, R.: Tesis Propuesta de un modelo de proceso de diseño industrial apoyado en las nuevas tecnologías de la información y su aplicación a un caso de estudio. UAM, Azcapotzalco (2007)
2. Navarrina Martínez, F.: Tesis doctoral Una metodología general para optimización estructural en diseño asistido por el ordenador. Escuela técnica superior de ingenieros de caminos, canales y puertos de Barcelona, España (1987)
3. Abella, B.: Diseño de exposiciones. Concepto, instalación y montaje, pp. 27 (2013) http://fido.palermo.edu/servicios_dyc/blog/docentes/trabajos/14058_47095.pdf
4. Padilla, J.: Diseño, construcción y operatividad de exhibiciones interactivas (2018) https://www.red pop.org/diseo-construccin-y-operatividad-de-exhibiciones
5. Miranda, H.: Diseño de exhibiciones interactivas. pp. 75 (2009) http://martinelli2008.blogspot.mx/
6. De la Cruz, G., Eslava, A. L., Castañeda, R.: Diseño de la experiencia del usuario para espacios interactivos de aprendizaje no formal. Research in Computing Science, vol. 89, pp. 62 (2015)
7. Ávila, J.: Tesis: Diseño de una exhibición interactiva para un museo en Noruega. Instituto Tecnológico de Costa Rica (2019)
8. Digalix.: https://www.digalix.com/es/asistente-virtual-realidad-aumentada-xholo/
9. Tidwell, J.: Designing Interfaces. O'Reilly Media, Inc. Canadá (2010) http://bedford-computing.co.uk/learning/wp-content/uploads/2016/07/Livro-Designing-Interfaces-2nd-Edition-2010.pdf

10. Sosa, M. C.: Inteligencia artificial en la gestión financiera empresarial. Pensamiento & Gestión, Universidad del Norte, Barranquilla, Colombia, núm. 23, pp. 153-186 (2007) https://www.redalyc.org/articulo.oa?id=64602307
11. Pascual, J. A.: Inteligencia artificial: Qué es, cómo funciona y para qué se utiliza en la actualidad. Computer Hoy (2019) https://computerhoy.com/reportajes/tecnologia/inteligencia-artificial-469917#tipos
12. Lauren, Styx.: How are museums using artificial intelligence, and is AI the future of museums? The Museum Next, EVE Museos e Innovación (2019) https://evemuseografia.com/2020/02/24/como-utilizan-los-museos-la-inteligencia-artificial/

ISSN 1870-4069

Desarrollo de una base de datos para el reconocimiento de la lengua de señas mexicana

Kenneth Mejía-Pérez, Diana-Margarita Córdova-Esparza,
Erika del-Río-Magaña

Universidad Autónoma de Querétaro,
Facultad de Informática,
México

{diana.cordova, erika.delrio}@uaq.mx,
ickennethmp@gmail.com

Resumen. El reconocimiento automático de la lengua de señas involucra tareas fundamentales para su desarrollo, por ejemplo la adquisición de información para el entrenamiento y validación de resultados de los diversos clasificadores basados en técnicas de Inteligencia Artificial. En el presente artículo se describe la metodología utilizada para el desarrollo de una base de datos que puede aplicarse en el reconocimiento de la lengua de señas mexicana. Esta base de datos consiste en la captura y etiquetado de las 27 letras del abecedario, de las cuales 6 son señas dinámicas (con movimiento) y 21 estáticas (sin movimiento), se añade también un análisis fonológico para cada una de las señas. Las señas fueron adquiridas mediante el sensor Kinect V2, se obtuvieron un total de 540 grupos de datos, los cuales corresponden a 27 señas con 20 repeticiones cada una.

Palabras clave: Base de datos, LSM, reconocimiento automático.

Development of a Database for Mexican Sign Language Recognition

Abstract. Automatic recognition of sign language involves fundamental tasks for its development, such as acquiring information for training and validating the results of various classifiers based on Artificial Intelligence techniques. This article describes the methodology used to develop a database that can be applied in the recognition of Mexican Sign Language. This database consists of capturing and labeling the 27 letters of the alphabet, of which 6 are dynamic signs (with movement) and 21 are static signs (without movement). A phonological analysis was also added for each of the signs. The signs were acquired using the Kinect V2 sensor, obtaining a total of 540 data groups, which correspond to 27 signs with 20 repetitions each.

Keywords: Database, Mexican sign language, automatic recognition.

1. Introducción

La comunicación es una herramienta fundamental para el acceso a la información, la participación social y el desarrollo de la vida en comunidad. Entre las diferentes formas de comunicación, se destaca la expresión oral como la más común e importante, en consecuencia como se menciona en [26] si esta vía de comunicación se ve suprimida por condiciones médicas se imposibilita la realización social, influyendo en otros aspectos de la vida personal, como: el desarrollo educativo, profesional y humano.

Las personas pertenecientes a la comunidad sorda presentan una dificultad para la comunicación oral en una o ambas direcciones (receptores y emisores), por tal motivo la comunidad ha desarrollado su propia lengua, es decir la lengua de señas.

La lengua de señas mexicana (LSM) ha sido objeto de estudio desde los años 80 en áreas de ciencias sociales. Sin embargo, se comenzó a popularizar en las áreas de computación y electrónica a principios del siglo XXI, esto debido a la creciente necesidad de utilizar la tecnología para dar soporte a través de traductores automáticos [20, 16] y para apoyo en la enseñanza y el aprendizaje [1, 22, 3], entre otras aplicaciones.

Por otro lado, existen trabajos basados en visión por computadora, los cuales emplean cámaras y algoritmos para realizar la detección y clasificación de señas pertenecientes a la lengua de señas, estos trabajos pueden dividirse en dos grandes grupos:

a) Propuestas que utilizan cámaras de color (RGB) como medio de adquisición de datos, por ejemplo los trabajos [28, 5, 17],

b) Trabajos que utilizan cámaras de color y profundidad (RGB-D) como lo son [8, 29, 9].

En ambos grupos se busca lograr la meta de crear un traductor de la lengua de señas que sea capaz de reconocer y clasificar las señas de forma eficiente, algunas aplicaciones prácticas de este tipo de sistemas de reconocimiento se ejemplifican en los trabajos desarrollados en [33, 14].

Este artículo como parte de un proyecto de investigación para el reconocimiento automático de la LSM propone la creación de una base de datos que contenga información necesaria para la clasificación e interpretación de las señas.

1.1. Trabajos relacionados

En la literatura se encuentran diversas bases de datos que se utilizan para el reconocimiento automático del lenguaje de señas en diferentes países del mundo. Por ejemplo: Alemania (DGS) [32, 7], Estados Unidos de América (ASL) [13, 25, 18, 11, 6], Arabia (ArASL) [10], Argentina (LSA) [24], México (LSM) [21, 15], Polonia (PJM)[12], Turquía (TID) [19], entre otros.

Estas bases de datos contienen datos sobre personas expresándose en la respectiva lengua de señas de su país, generalmente esta información se presenta en forma de videos o imágenes RGB, y en algunas ocasiones incluyen algún tipo de dato adicional,

Tabla 1. Bases de datos para el reconocimiento de la lengua de señas.

Nombre de la base de datos y referencia	Lenguaje de señas	Categorías (Señas Distintas)	Total de capturas	Perspectiva	Dispositivo de adquisición de datos	Tipos de datos adquiridos
RWTH-PHOENIX-Weathe [7]	Alemán	1,200	45,760 ejemplos	Personas viendo hacia el frente	Cámara de color	Videos de color resolución 210×260
PSL Kinect 30 [12]	Polaco	30	300 videos	Persona viendo hacia el frente	Kinect	Nubes de puntos videos de profundidad 320×240 px
PSL TOF 84 [12]	Polaco	84	1680 videos	Personas con ropa negra de frente a la cámara	Cámara ToF	Video de 176 144 fotogramas 50 fotogramas/segundo Las nubes de puntos varían según el ejemplo
SIGNUM [32]	Alemán	825 (45 señas básicas, 780 oraciones)	33,210 secuencias de video 5,970,450 imágenes aprox. 55.3 h de video	Personas viendo hacia el frente	Cámara de color	Video de color Resolución 775×578 Imagen de color 24 bpp
LSA64: An Argentinian Sign Language Dataset [24]	Argentino	64	3,200 videos	Cuerpo de frente y uso de guantes de color	Cámara RGB	Videos de color Resolución 1920×1080
American Sign Language Lexicon Video Dataset (ASLLVD) [18]	Americano	Mayor a 3300	Más de 9800 videos	Cuerpo completo de frente y perfil, enfoque en ambas manos	Cámara RGB	Videos de color 640×480 pixeles 2020 fotogramas por video
MS-ASL [11]	Americano	1,000	25,513 videos	Personas viendo hacia el frente	Obtención por recortes de videos públicos	Procesamiento para ajustar a 24×24 pixeles
Rivas dataset [21]	Mexicano	10	3,000 imágenes	Enfoque en una mano	Kinect V1	Imágenes de color 115×151 pixeles
Base de datos [15]	Mexicano	21	6,300 imágenes	Enfoque en una mano	Cámara RGB	Imágenes de color 20×20 pixeles
ArASLdataset [10]	Árabe	32	54,049 imágenes	Enfoque en una mano	Cámara RGB	Imágenes en escala de grises 64×64 pixeles
BosphorusSign22k [19]	Turco	744	22,542 videos	Personas viendo hacia el frente	Kinect V2	Videos de color 1920×1080 px Videos de profundidad 30 FPS 512×424 pixeles Información de la posición de los cuerpos
How2Sign [6]	Americano	Mas de 16,000 palabras	35,000 (83 horas de video)	Personas de frente y perfil	Cámara RGB y sensor de profundidad	Videos de color resolución 1280×720
WLASL [13]	Americano	2,000 palabras	21,083 videos	Personas viendo hacia el frente	Cámaras de profundidad y de color	Videos y poses del esqueleto en 2D
ASL-LEX 2.0 [25]	Americano	2,723	2,723 videos	Persona viendo hacia el frente	Cámara de color	Archivos en formato webm, sin información

como etiquetas o distancias cuando se tienen imágenes de profundidad adquiridas con cámaras RGB-D.

En referencia a la lengua de señas mexicana, existen dos bases de datos de libre acceso, la primera de ellas es la base de datos descrita en [21], la cual consiste en 10 repeticiones de imágenes de profundidad de gestos manuales estáticos de los números del 0 al 9, captadas por un sensor Kinect con una resolución de 115×151 pixeles, para cada una de las clases se incluyen 300 repeticiones, es decir, se obtuvieron un total de 3,000 imágenes, cada imagen está etiquetada con la clase a la que pertenece y un ejemplo.

La segunda base de datos pertenece a [8], formada por 21 clases distintas, con 300 repeticiones por clase dando un total de 6,300 imágenes con una resolución de 20 x 20 pixeles, al igual que la base de datos presentada en [21] ésta se enfoca en gestos manuales estáticos focalizados en una sola mano, la diferencia principal es que esta base de datos consta de 21 señas estáticas del alfabeto y las imágenes fueron tomadas por una cámara de color.

Por otra parte, la mayoría de las bases de datos de acceso público encontradas en la literatura están enfocadas en la lengua de señas americana, entre las cuales se destacan: WLASL [13] y ASL-LEX 2.0 [25] ya que las categorías superan las 2,000 clases, en

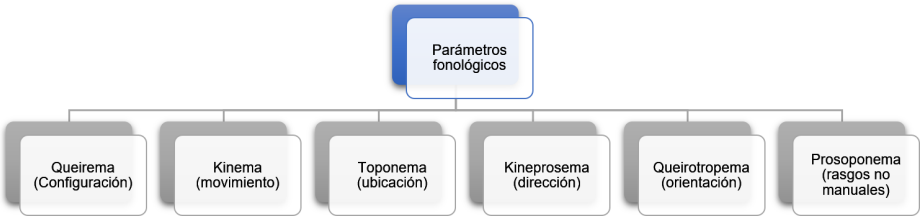


Fig. 1. Parámetros fonológicos de la lengua de señas.

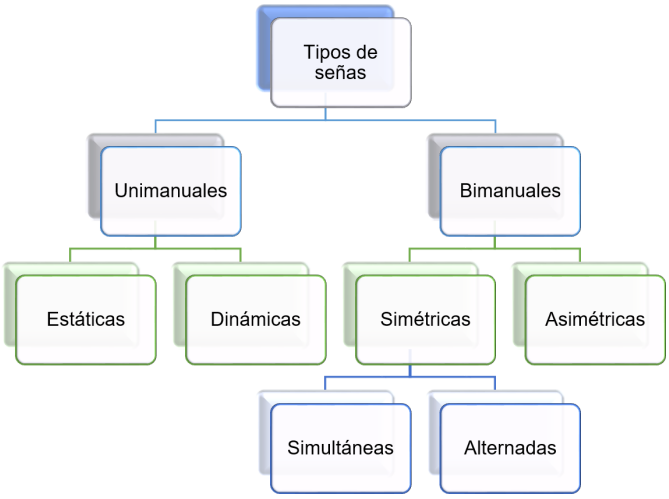


Fig. 2. Tipos de señas.

el caso de WLASL cada una de las palabras fue replicada alrededor de 10 veces (con 119 señantes distintos) dando un total de 21,083 videos de señas dinámicas, los videos constan de personas frente a la cámara realizando las señas y los datos guardados son imágenes de color y puntos 2D del esqueleto humano, mientras que ASL-LEX 2.0 consta de un único video para cada una de las señas.

A continuación, en la Tabla 1 se muestra una comparativa de las bases de datos utilizadas en el reconocimiento automático de la lengua de señas de diferentes países. Como se puede observar en la tabla, la mayoría de las bases de datos pertenecen al lenguaje de señas americano, también se puede notar que las bases de datos difieren en la extensión de sus categorías; desde 10 hasta más de 2,000 categorías.

2. Parámetros fonológicos de la lengua de señas

La lengua de señas como cualquier otro idioma tiene su propia estructura fonológica. Sin embargo, el término fonología hace referencia a sonidos verbales [4], por consecuencia este concepto no describe adecuadamente el estudio de la lengua de

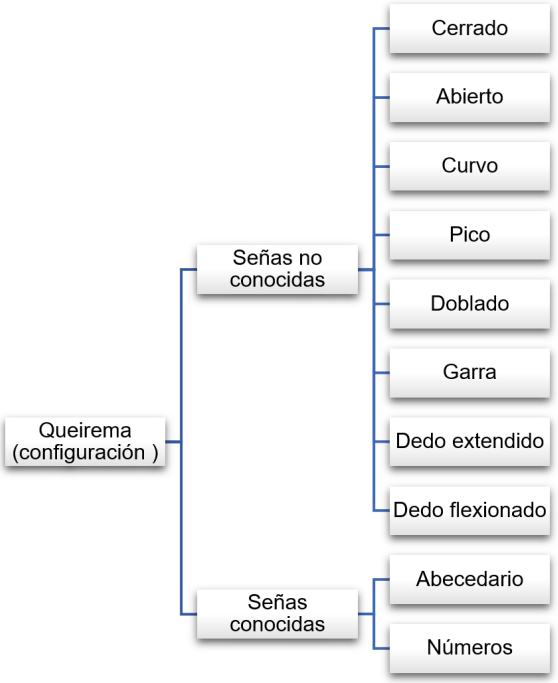


Fig. 3. Configuración de las señas.

señas. Es por ello que la fonología de la lengua de señas es conocida como querología y los fonemas como queremas [31].

La querología se utiliza para describir los morfemas o unidades mínimas de la lengua de señas [27], las cuales en conjunto construyen cada una de las señas del lenguaje; cambiar alguno de estos parámetros en cualquiera de las señas podría cambiar su significado. Los parámetros fonológicos de la lengua de señas se pueden expresar en seis componentes principales [23], estos parámetros son los siguientes (ver Figura 1):

1. Queirema (configuración): Configuración manual de cada seña.
2. Toponema (ubicación): Ubicación en relación al cuerpo.
3. Kineprosema (dirección): Dirección del movimiento de las manos.
4. Queirotropema (orientación): Orientación de la mano con respecto al cuerpo.
5. Prosoponema (rasgos no manuales): Todos aquellos rasgos que no utilizan las manos, principalmente movimiento corporal y expresiones faciales.
6. Kinema (movimiento): Tipo de movimiento de las manos (circular, zig zag, lineal, etc.).

La lengua de señas puede clasificarse por el uso de una o dos manos, esto se conoce como el uso de señas unimanuales o bimanuales respectivamente como se indica en [2]. Cada señante tiene una mano base y una dominante, éstas pueden alternarse entre

Fig. 4. Ejemplo del uso de la configuración de Y en la palabra jugar.

Fig. 5. Ejemplo del cambio de configuración en la seña del número 100.

izquierda y derecha y no pierde el sentido de la palabra o el significado, las señas unimanuales se pueden categorizar como estáticas y dinámicas.

Las señas estáticas son aquellas que no requieren de movimiento para interpretarse, mientras que las dinámicas si; las señas bimanuales son dinámicas es decir, requieren el movimiento de la mano dominante, o incluso de la mano base, estas señas a su vez pueden ser simétricas o asimétricas, esto se atribuye a que ambas manos tengan o no la misma configuración, a su vez las señas pueden contener movimiento simultáneo o alternado; las señas simultáneas son aquellas que tienen un movimiento inversamente proporcional, como si se tratara de un espejo.

Por otra parte, las señas alternadas tienen movimientos idénticos inversos. Esta clasificación se puede apreciar en la Figura 2, en donde la clasificación se encuentra en 4 niveles, en el nivel más alto se comienza la división por los tipos de señas, en el segundo nivel se divide en unimanuales y bimanuales, es decir, realizadas con 1 o 2 manos, en el siguiente nivel se finaliza la clasificación de señas unimanuales como estáticas y dinámicas. Por otra parte, las señas bimanuales se clasifican como

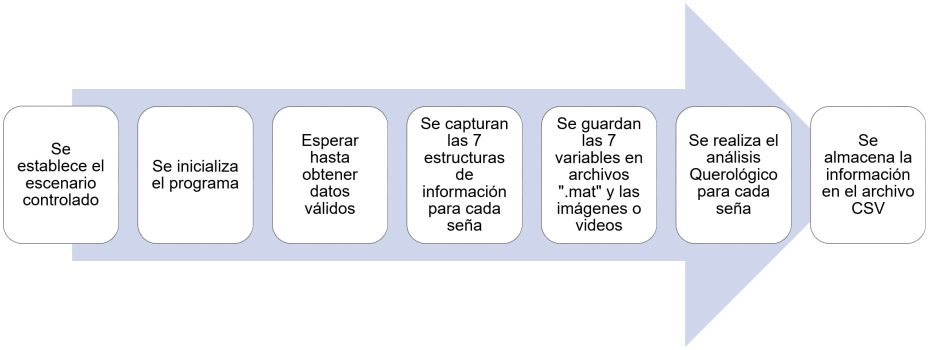


Fig. 6. Secuencia de pasos para la captura y almacenamiento de las señas.

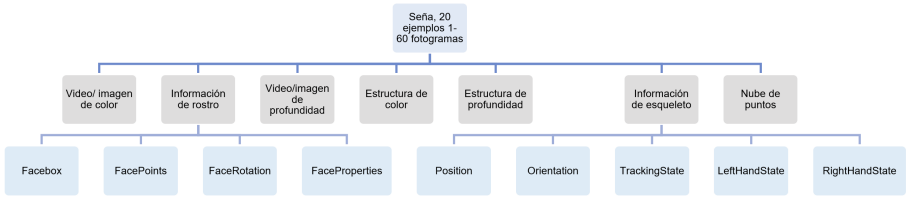


Fig. 7. Ejemplo de la colección de datos obtenida para una seña.

simétricas y asimétricas, finalmente en el cuarto nivel se encuentran las señas simétricas simultáneas y alternadas.

La forma que adoptan las manos al realizar una seña es llamada configuración, las manos pueden tener más de una configuracion a lo largo de una sola seña, además estas configuraciones pueden ser o no ser iguales en ambas manos [21], en la Figura 3 se muestra una clasificación sobre las descripciones que se pueden realizar en una configuración, por ejemplo, en la palabra “jugar” ambas manos adquieren la configuración de la letra “Y”, esta seña también incluye un movimiento circular alternado como se observa en la Figura 4, actualmente no se encuentra estandarizada una cantidad finita de señas para su estudio.

Sin embargo, se pueden contar más de cien configuraciones, de las cuales se destacan el abecedario y los números naturales con algunas variaciones. Estas variaciones no siempre tienen un nombre específico, para describir nuevas configuraciones se puede partir de la idea de una mano con los dedos completamente erguidos verticalmente (una configuración de “B” con el pulgar extendido verticalmente).

Para describir la seña se pueden considerar seis posiciones: cerrada, abierta, curva, garra, doblado, pico, con dedos extendidos o flexionados, como se indica en la Figura 3. Las señas dinámicas (con movimiento) pueden tener uno o varios cambios de configuración mientras se realiza; un ejemplo es el caso del número 100, que además de tener un movimiento horizontal cambia la postura del dedo índice (ver Figura 5). La

Algoritmo 1: Obtención de la colección de datos para cada una de las señas

Entrada: (Nombre_de_Seña, No._fotogramas)
Salida : Colección de datos → (vidColor, vidDepth, EstColor, EstDepth, EstFace, EstSkeleton, EstPC)

```
Procedure ;
for i ← 1 to 20 do
    Crear carpetas;
    Crear variables locales;
    while datos_disponibles==0 do
        Esperar a disponibilidad de los datos;
    end
    for i ← 1 to 20 do
        Mostrar vista previa del video durante 5 fotogramas;
        Mostrar vista previa del video en canal 1 durante 5 fotogramas;
        Mostrar vista previa del video en canal 2 durante 5 fotogramas;
        Mostrar vista previa del video en canal 3 durante 5 fotogramas;
    end
    while datos_disponibles==0 do
        Obtener datos de color;
        Obtener datos de profundidad;
        Obtener datos de nube de puntos;
        Obtener datos del rostro;
        Obtener datos del esqueleto;
        vidColor = Guardar fotograma de color;
        vidDepth = Guardar fotograma de profundidad;
        EstColor = Guardar fotograma de color como matriz;
        EstDepth= Guardar fotograma de profundidad como matriz;
        EstFace = Guardar información de rostro del fotograma;
        EstSkeleton = Guardar información del esqueleto del fotograma;
        EstPC = Guardar nube de puntos del fotograma;
    end
end
Guadar_archivos[vidColor, vidDepth, EstColor, EstDepth, EstFace, EstSkeleton, EstPC];
```

orientación juega un papel muy importante para la definición de cada seña ya que de esta depende gran parte de su significado; se debe tomar como referencia la palma de la mano completamente extendida y comenzar a describir a través de la posición de la palma (respecto al receptor) la dirección a donde apuntan los dedos, algunas de las orientaciones comunes son las siguientes: palmas al frente, dedos apuntando arriba, palmas atrás y dedos apuntando arriba, palmas al frente y dedos apuntando hacia afuera, palmas hacia atrás y dedos apuntando adentro, palmas hacia abajo y dedos al frente, palmas arriba y dedos al frente, palmas arriba y dedos apuntando adentro, palmas abajo y dedos apuntando adentro, entre otras.

Sin embargo, puede existir un grupo más grande de orientaciones, con más variaciones; por ejemplo, colocar las manos en diagonal y usar grados de inclinación.

Otra parte importante del análisis querológico del lenguaje de señas es la ubicación de la seña, esto es esencial, ya que dependiendo de la ubicación podría cambiar completamente el sentido de la frase. Esto es más notorio en las palabras inicializadas,

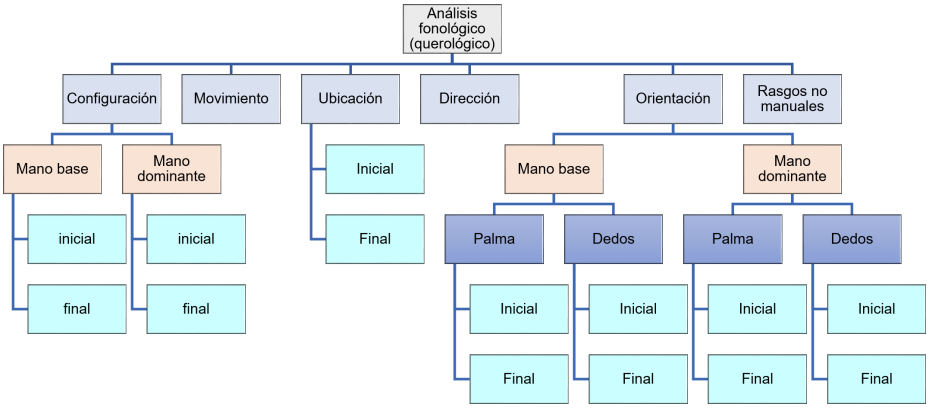


Fig. 8. Estructura del archivo CSV, contiene la descripción fonológica o querológica para cada seña almacenada.

es decir, en palabras que adoptan como configuración inicial su palabra homóloga del idioma español, la ubicación de la seña está determinada por el cuerpo de la persona, algunas ubicaciones que se pueden mencionar son: a la altura del pecho, hombros, cabeza, boca, barbilla, mejilla, estómago, costado, entre otras. Asimismo, para la descripción de las señas dinámicas se utiliza la dirección y el movimiento que adquieren las manos; la dirección se toma en el sentido de la posición inicial de la seña hacia la posición final en relación al cuerpo del señante, como por ejemplo, arriba, abajo, adentro y afuera.

El movimiento no depende del cambio de configuraciones de la mano, es dependiente del movimiento de los brazos, muñecas y codos. Algunos ejemplos de movimientos son: circular, semicircular, diagonal, zigzag y lineal.

El movimiento y dirección pueden cambiar el contexto de una palabra aún cuando posean la misma configuración y ubicación, por ejemplo la(s) palabra(s): año/años, la configuración de la mano dominante y base es empuñada, las manos son ubicadas a la altura del pecho, la mano dominante por encima de la base, el movimiento ejecutado por la mano dominante es el de un semi-círculo, si está en dirección al frente (de arriba hacia abajo y de adelante hacia atrás) la palabra querrá decir años en futuro o presente, en cambio atrás (de arriba hacia abajo, de atrás hacia adelante) significa años en pasado.

Finalmente, también puede apoyarse de rasgos no manuales, los cuales tratan de expresar sentimientos y sensaciones con el resto del cuerpo, principalmente con el rostro, con el propósito de mostrar felicidad, tristeza, repugnancia, intriga, emoción, etc.

Estos rasgos también se utilizan para formular interrogantes, ya que como se menciona en [21] se requiere fruncir las cejas e inclinar hacia adelante la cabeza para preguntar algo mientras se hace el señado de la frase, con el objetivo de denotar que la frase es una interrogante.

Fig. 9. Imagen de color y de profundidad captada por el Kinect v2. En la imagen se representa la seña de la letra "A".

Numero	Nombre	Categoria	Estatica / Dinamica	Unimanuam /Bimanual	Ubicación inicial	Ubicación Final	Movimiento	Dirección	Rasgos no manuales	Configuracion inicial mano base	Configuracion final mano base	Configuracion inicial mano dominante	Configuracion final mano dominante	Orientacion inicial palma mano base	Orientacion final palma mano base	Orientacion inicial palma mano	Orientacion final palma mano dominante	Orientacion inicial de los dedos mano base	Orientacion final de los dedos mano base	Orientacion inicial de los dedos mano	Orientacion final de los dedos mano dominante
1	A	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	A	A	N/A	N/A	Frente	Frente	N/A	N/A	Arriba	Arriba
2	B	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	B	B	N/A	N/A	Frente	Frente	N/A	N/A	Arriba	Arriba
3	C	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	C	C	N/A	N/A	Diagonal Adentro Frente	Diagonal Adentro Frente	N/A	N/A	Arriba	Arriba
4	D	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	D	D	N/A	N/A	Frente	Frente	N/A	N/A	Arriba	Arriba
5	E	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	E	E	N/A	N/A	Frente	Frente	N/A	N/A	Arriba	Arriba
6	F	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	F	F	N/A	N/A	Frente	Frente	N/A	N/A	Arriba	Arriba
7	G	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	G	G	N/A	N/A	Atrás	Atrás	N/A	N/A	Adentro	Adentro
8	H	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	H	H	N/A	N/A	Atrás	Atrás	N/A	N/A	Adentro	Adentro
9	I	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	I	I	N/A	N/A	Frente	Frente	N/A	N/A	Arriba	Arriba
10	J	Abecedario	Dinamica	Unimanual	Pecho	Pecho	Medio circulo	Arriba - Abajo	N/A	N/A	N/A	I	I	N/A	N/A	Frente	Adentro	N/A	N/A	Arriba	Arriba
11	K	Abecedario	Dinamica	Unimanual	Pecho	Pecho	Cuarto de circulo	Frente - Abajo	N/A	N/A	N/A	P	P	N/A	N/A	Frente	Abajo	N/A	N/A	Arriba	Frente
12	L	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	L	L	N/A	N/A	Frente	Frente	N/A	N/A	Arriba	Arriba
13	M	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	M	M	N/A	N/A	Atrás	Atrás	N/A	N/A	Abajo	Abajo
14	N	Abecedario	Estatica	Unimanual	Pecho	Pecho	N/A	N/A	N/A	N/A	N/A	N	N	N/A	N/A	Atrás	Atrás	N/A	N/A	Abajo	Abajo
15	Ñ	Abecedario	Dinamica	Unimanual	Pecho	Pecho	Medio circulo	Adentro-Afuera	N/A	N/A	N/A	N	N	N/A	N/A	Atrás	Atrás	N/A	N/A	Abajo	Abajo

Fig. 10. Estructura del archivo CSV, contiene 22 columnas y 27 filas (se muestran las primeras 15).

3. Método y materiales

Esta sección describe la metodología propuesta para la generación de una base de datos para el reconocimiento automático de la LSM. El dispositivo utilizado para la adquisición de los datos es un Kinect versión 2 (Kinect v2).

Sin embargo, el procedimiento puede ser replicado para otro tipo de cámaras RGB-D. Los datos capturados son videos, imágenes de color y de profundidad para cada seña, además de la información de cada fotograma como la detección del esqueleto y del rostro, así como una nube de puntos del escenario completo, también se incluye el análisis fonológico de cada una de las señas adquiridas.

Otro punto importante sobre el uso de este dispositivo es la compatibilidad que tiene con diferentes lenguajes de programación. Para el desarrollo de la base de datos se utilizó el lenguaje de programación MATLAB en conjunto con la librería Kin2 [30], debido a que MATLAB es una plataforma altamente optimizada para la resolución de este tipo de problemas computacionales. No obstante, también podría utilizarse algún otro tipo de lenguaje de programación de alto nivel como C# o python.

La captura de los datos de la propuesta descrita en este documento se realizó con un enfoque similar al utilizado en otros trabajos [32, 7, 13, 25, 11, 12, 19] es decir, el participante se encuentra de forma frontal a la cámara.

Los pasos que se llevan a cabo para para la adquisición y almacenamiento de los datos son los siguientes (ver Figura 6): Se establece un escenario controlado, se inicia el programa de captura y se realiza una espera hasta obtener datos válidos de la cámara.

Enseguida, se adquieren las 7 estructuras de información descritas en la Sección 3.1. Estas estructuras se almacenan en la computadora, posteriormente se realiza el análisis Querológico (Fonológico) de cada seña (categoría) obtenida y finalmente se almacena esta información en un archivo separado por comas (CSV).

3.1. Adquisición de los datos

Para el desarrollo de la presente base de datos se adquirieron un total de 27 señas o clases distintas correspondientes al alfabeto de la lengua de señas mexicana, para cada una de estas señas o clases se capturaron 20 repeticiones, alternando la mano dominante entre la izquierda y la derecha, dando un total de 540 entradas de datos.

Cabe mencionar que la base de datos se encuentra en su etapa inicial y se ampliará esta cantidad de datos debido a los requerimientos del proyecto para el reconocimiento automático de señas. Cada una de las 540 entradas incluye lo siguiente:

1. Un video o imagen de color (de acuerdo al tipo de seña: dinámica o estática) con una resolución de 1920×1080 pixeles.
2. Un video o imagen de profundidad (de acuerdo al tipo de seña: dinámica o estática) con una resolución de 512×424 pixeles.
3. Una estructura que contiene matrices tridimensionales correspondientes a las capas de color para cada fotograma de dimensiones $1920 \times 1080 \times 3$.
4. Una estructura que contiene la matriz de información de profundidad de dimensiones 512×424.
5. Una estructura con información sobre el rostro, esta estructura está formada por cuatro clases:
   - FaceBox: Una caja rectangular que enmarca la cara del usuario.
   - FacePoints: Cinco puntos alineados localizados en el rostro del usuario.
   - FaceRotation: La rotación del rostro expresada en ángulos de Euler (pitch, yaw, roll).
   - FaceProperties: Se incluyen los siguientes ocho datos: Happy, Engaged, WearingGlasses, LeftEyeClosed, RightEyeClosed, MouthOpen, MouthMoved y LookingAway.
6. Una estructura que contiene información sobre el esqueleto del señante:
   - Position,
   - Orientation,

- TrackingState,
- LeftHandState,
- RightHandState.

7. Una estructura que contiene para cada fotograma una nube de puntos 3D de tamaño N $\times$ 3, tal que N corresponde a la resolución (512 $\times$ 424) de la imagen de profundidad. Esta matriz de 217088 $\times$ 3 contiene la información espacial de cada pixel.

En la Figura 7 se muestra un ejemplo de la colección de datos obtenida para una seña. Como se mencionó anteriormente, los datos fueron adquiridos por medio del sensor Kinect v2; para realizar esta adquisición se colocó el dispositivo en un tripié a una altura de 1.20 m y a una distancia de 1.5 metros del señante, como características adicionales se puede mencionar que el señante utiliza ropa negra, el enfoque del cuadro comienza cerca de la rodilla y termina más arriba de la cabeza, esto con la finalidad de capturar la información del esqueleto y el rostro, así como la ubicación (respecto al cuerpo) en donde se realizan las señas.

Para la adquisición de los datos se desarrolló un programa (ver Algoritmo 1) para adquirir 60 fotogramas en caso de señas dinámicas o 1 en caso de señas estáticas, los fotogramas proporcionan suficiente información para el señado de palabras individuales.

Como dato de entrada al programa, se indica la cantidad de fotogramas que se desean adquirir además, el nombre de la seña que se va a realizar y como dato de salida entrega la colección de datos con las 7 estructuras anteriormente mencionadas.

Los datos se guardan en un archivo de MATLAB, con la siguiente nomenclatura: "nombre de la seña_Numero de repeticion_img.m" o "vid_tipo de dato.m"; en caso de ser una seña estática utiliza "Img" o en caso de ser dinámica "vid". El tipo de dato puede ser: color, depth, face, PC (point cloud) o skeleton, por ejemplo K_13_VidPC es la información de video para la nube de puntos de la repetición 13 de la letra K.

Después de recibir los datos de entrada se crean automáticamente las variables locales para almacenar los archivos de salida. Antes de continuar con la captura de los datos inicia un ciclo de control para esperar a que el dispositivo esté disponible, este paso es importante debido a que si no se realiza podría haber pérdida de información.

Cuando el dispositivo esté disponible, mostrará una vista previa del video de color obtenido por el Kinect para indicar al usuario cuando comenzar a realizar una seña. Después de tomar los 60 cuadros se almacenarán las variables y guardarán los archivos generados dentro de sus respectivas carpetas contenedoras.

Después de la obtención de la base de datos el último paso es el etiquetado de las señas en un archivo delimitado por comas (CSV), este formato fue elegido debido a que es fácil de importar y editar desde cualquier editor de hojas de cálculo o de archivos de texto; este paso se llena de forma manual con la ayuda de un procesador de hojas de cálculo y se realiza un análisis querológico de cada una de las señas capturadas, en este caso las 27 señas del abecedario.

La tabla de datos obtenida contiene 22 columnas, organizadas de la siguiente manera: Número, nombre, categoría, estática/dinámica, unimanual/bimanual, ubicación

inicial, ubicación final, movimiento, dirección, rasgos no manuales, configuración inicial de la mano base, configuración final de la mano base, configuración inicial de la mano dominante, configuración final de la mano dominante, orientación inicial de la palma de la mano base, orientación final de la palma de la mano base, orientación inicial de la palma de la mano dominante, orientación final de la palma de la mano dominante, orientación inicial de los dedos de la mano base, orientación final de los dedos de la mano base, orientación inicial de los dedos de la mano dominante y orientación final de los dedos de la mano dominante; en cada una de estas columnas se agrega la información correspondiente, si alguna columna no aplica para la seña en cuestión se agrega la etiqueta N/A, en la Figura 8 se muestra un diagrama sobre la estructura del archivo CSV.

4. Resultados

Como resultados se obtuvieron un total de 540 entradas de datos, las cuales corresponden a 27 señas con 20 repeticiones cada una. Cada entrada de datos contiene 7 archivos: 2 de ellos son imágenes (ver Figura 9) o videos demostrativos, mientras que las otras 5 entradas de datos corresponden a información que puede procesarse en MATLAB u otros lenguajes de programación similares, estos archivos tienen información de color, profundidad, puntos del rostro, puntos del esqueleto y nubes de puntos 3D para 1 o los 60 fotogramas de cada una de las señas.

La base de datos se actualizará periodicamente cuando se obtengan nuevos datos, para su consulta se encuentra disponible en el siguiente enlace: https://github.com/ICKMejia/Kenneth_DatasetLSM. Se obtuvo un archivo CSV con 27 categorías para cada una de las señas realizadas, a lo largo de cada una de sus columnas se realiza el análisis fonológico o querológico de las señas almacenadas (ver Figura 10), en esta descripción se tomaron en cuenta los 6 aspectos principales presentados en la Sección 2: configuración, movimiento, ubicación, dirección, orientación y rasgos no manuales para cada una de las manos.

Estos datos son importantes porque permiten clasificar las señas con más precisión y no únicamente por la categoría a la que pertenecen según su significado. En comparación a las bases de datos descritas en la Tabla 1 la base de datos propuesta es la única que incluye un análisis fonológico para cada una de las 27 clases capturadas, además se incluyen 7 tipos de datos para cada uno de los 540 ejemplos, los cuales son: imagen/video de color y de profundidad, matriz de color y de profundidad, información sobre el rostro, el esqueleto y una nube de puntos 3D, mientras que la mayoría de las bases de datos citadas en la Tabla 1, proporcionan uno o dos tipos de datos como lo son: imágenes o videos de color y/o profundidad.

5. Conclusiones

La creación de la base de datos descrita en este documento surge de la necesidad de contar con información para el desarrollo de un proyecto que consiste en la detección y reconocimiento automático de palabras de la lengua de señas mexicana.

La base de datos permite capturar el cuerpo del participante, con la finalidad de no perder información relevante de acuerdo al análisis querológico desarrollado.

De esta manera, la generación del archivo de etiquetado permite establecer las relaciones entre las señas y su fonología; se obtuvieron un total de 540 entradas de datos, las cuales corresponden a 27 señas con 20 repeticiones cada una.

La generación de los datos para cada una de las señas tomó un tiempo considerable debido al tiempo de procesamiento que requiere el programa para crear los videos de alta resolución y generar los archivos con los datos estructurados, además, el espacio de almacenamiento es considerablemente alto ya que el espacio consumido para 540 señas (sólo 6 de ellas estáticas) es cercano a los 21 GB.

Como trabajo futuro se plantea la idea de delimitar un área de interés sin perder información relevante y así reducir considerablemente los datos adquiridos; además se podrían utilizar técnicas de escalamiento para reducir el tamaño de las imágenes de color y profundidad, así como el de las nubes de puntos capturadas.

Referencias

1. Aguilar, I., Reina, A. J., Mandow, A.: Herramienta para el aprendizaje del lenguaje dactilológico mediante vision artificial (2015)
2. Aldrete, M. C.: Gramatica de la lengua de senas mexicana. Estudios de lingüística del español, , no. 28, pp. 1 (2009)
3. Araiza, A. B., Díaz, D. A., Segundo, L. M.: Herramienta en realidad virtual para el aprendizaje del lenguaje de senas mexicano. Research in Computing Science, vol. 148, pp. 55–61 (2019)
4. Burquest, D. A.: Análisis fonológico: Un enfoque funcional. Summer Institute of Linguistics International, no. 17, pp. 336 (2009)
5. Cervantes, J., García-Lamont, F., Rodríguez-Mazahua, L., Rendon, A. Y., Chau, A. L.: Recognition of mexican sign language from frames in video sequences. In: Intelligent Computing Theories and Application, pp. 353–362 (2016)
6. Duarte, A., Palaskar, S., Ghadiyaram, D., DeHaan, K., Metze, F., Torres, J., Giro-i Nieto, X.: How2Sign: A large-scale multimodal dataset for continuous american sign language (2020)
7. Forster, J., Schmidt, C., Hoyoux, T., Koller, O., Zelle, U., Piater, J. H., Ney, H.: Rwth-phoenix-weather: A large vocabulary sign language recognition and translation corpus. In: LREC. vol. 9, pp. 3785–3789 (2012)
8. Galicia, R., Carranza, O., Jiménez, E., Rivera, G.: Mexican sign language recognition using movement sensor. In: 2015 IEEE 24th International Symposium on Industrial Electronics (ISIE). pp. 573–578 (2015) doi: 10.1109/ISIE.2015.7281531
9. Garcia-Bautista, G., Trujillo-Romero, F., Caballero-Morales, S. O.: Mexican sign language recognition using kinect and data time warping algorithm. In: International Conference on Electronics, Communications and Computers (2017) doi: 10.1109/conielecomp.2017.7891832
10. Ghazanfar, L., Jaafar, A., Nazeeruddin, M., Roaa, A., Rawan, A.: ArASL: Arabic alphabets sign language dataset, vol. 23 (2019)
11. Joze, H. R. V., Koller, O.: MS-ASL: A large-scale data set and benchmark for understanding american sign language (2018) doi: 10.48550/arXiv.1812.01053
12. Kapuscinski, T., Oszust, M., Wysocki, M., Warchol, D.: Recognition of hand gestures observed by depth cameras. International Journal of Advanced Robotic Systems, vol. 12, no. 4, pp. 36 (2015) doi: 10.5772/60091

13. Li, D., Rodriguez, C., Yu, X., Li, H.: Word-level deep sign language recognition from video: A new large-scale dataset and methods comparison. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 1459–1469 (2020)
14. Londono, C., Castro, R., Bedoya, H., Elias, O.: Application to support the process of training sign language through microsoft kinect®. In: Proceedings of the 10th Computing Colombian Conference (10CCC) (2015) doi: 10.1109/columbiancc.2015.7333462
15. Mancilla-Morales, E., Vázquez-Aparicio, O., Arguijo, P., Meléndez-Armenta, R. Á., Vázquez-López, A. H.: Traducción del lenguaje de señas usando visión por computadora. Research in Computing Science, vol. 148, pp. 79–89 (2019)
16. Martinez-Seis, B., Pichardo-Lagunas, O., Rodriguez-Aguilar, E., Saucedo-Diaz, E. R.: Identification of static and dynamic signs of the mexican sign language alphabet for smartphones using deep learning and image processing. Research in Computing Science, vol. 148, pp. 199–211 (2019)
17. Martínez-Gutiérrez, M., Rojano-Cáceres, J. R., Bárcenas-Patiño, I. E., Juárez-Pérez, F.: Identificación de lengua de señas mediante técnicas de procesamiento de imágenes. Research in Computing Science, vol. 128, pp. 121–129 (2016)
18. Neidle, C., Thangali, A., Sclaroff, S.: Challenges in development of the american sign language lexicon video dataset ASLLVD corpus. In: 5th Workshop on the Representation and Processing of Sign Languages: Interactions between Corpus and Lexicon, LREC (2012)
19. Özdemir, O., Kındıroğlu, A. A., Camgöz, N. C., Akarun, L.: Bosphorussign22k sign language recognition dataset, vol. 1 (2020) doi: 10.48550/arXiv.2004.01283
20. Pérez, L. M., Rosales, A. J., Gallegos, F. J., Barba, A. V.: LSM static signs recognition using image processing. In: 2017 14th International Conference on Electrical Engineering, Computing Science and Automatic Control (CCE), pp. 1–5 (2017) doi: 10.1109/ICEEE.2017.8108885\textbf
21. Rivas-Perea, P. E.: Desarrollo de un intérprete básico del lenguaje de señas para dactilología empleando inteligencia artificial. Ph.D. thesis, Tecnológico Nacional de México, Instituto Tecnológico de Nogales (2019)
22. Rodríguez-Fuentes, A., Lineth, A., García-García, F.: EnSenias: Technological tool to learn, teach, improve and use panamanian sign language. Íkala, vol. 25, no. 3, pp. 663–678 (2020)
23. Rodríguez-González, M. A.: Lengua de signos (2015)
24. Ronchetti, F., Quiroga, F., Estrebou, C. A., Lanzarini, L. C., Rosete, A.: LSA64: An Argentinian sign language dataset. In: XXII Congreso Argentino de Ciencias de la Computación (2016)
25. Sehyr, Z. S., Caselli, N., Cohen-Goldberg, A. M., Emmorey, K.: The ASL-LEX 2.0 project: A database of lexical and phonological properties for 2,723 signs in american sign language. The Journal of Deaf Studies and Deaf Education, vol. 26, no. 2, pp. 263–277 (2021) doi: 10.1093/deafed/enaa038
26. Serafín, M., González, R.: Manos con voz, diccionario de lenguaje de senas mexicana. Consejo Nacional para Prevenir la Discriminación, pp. 15–19 (2011)
27. Smith-Stark, T. C., Cruz-Aldrete, M.: La morfología en la lengua de señas mexicana. In: Conferencia magistral preparada para el II Congreso Internacional de Logogenia México (2006)
28. Solís, F., Martínez, D., Espinoza, O.: Automatic mexican sign language recognition using normalized moments and artificial neural networks. Engineering, vol. 8, no. 10, pp. 733–740 (2016) doi: 10.4236/eng.2016.810066
29. Sosa-Jimenez, C. O., Rios-Figueroa, H. V., Rechy-Ramirez, E. J., Marin-Hernandez, A., Gonzalez-Cosio, A. L. S.: Real-time mexican sign language recognition. In: IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC) (2017) doi: 10.1109/ropec.2017.8261606

30. Terven, J. R., Córdova-Esparza, D. M.: Kin2. a kinect 2 toolbox for matlab. Science of Computer Programming, vol. 130, pp. 97–106 (2016) doi: 10.1016/j.scico.2016.05.009
31. Torres, S., Sánchez, J., Carratalá, P.: Curso de bimodal (2008)
32. von-Agris, U., Friedrich-Kraiss, K.: Towards a video corpus for signer-independent continuous sign language recognition (2007)
33. Zamora Cedeño, N. A., Córdova Rivadeneira, L. S., Bohorquez Escobar, C. B., Villacres Cornejo, L. M.: Diseño de un sistema interfaz para el reconocimiento y traducción de gestos corporales al lenguaje natural (escrito, hablado) mediante el sensor kinect de microsoft, para personas con capacidades diferentes. Trabajos de Grado - Maestría en Telecomunicaciones (2018)

ISSN 1870-4069

Segmentación de caminos en entornos virtuales

José Héctor León-Chávez, Juan-Manuel Ramos-Arreguin,
Sebastián Salazar-Colores, Saúl Tovar-Arriaga,
Jesús Carlos Pedraza-Ortega

Universidad Autónoma de Querétaro,
Facultad de Ingeniería,
México

jleon08@alumnos.uaq.mx, jsistdig@yahoo.com.mx

Resumen. Los vehículos autónomos tienen una gran importancia en el campo de la inteligencia artificial. Se aplican redes neuronales entrenadas para adaptarse a entornos no conocidos. Algunas redes neuronales convolucionales (CNN) tuvieron gran impacto en diferentes campos de la ciencia, como la biología. Sin embargo, su eficiencia en otros campos es poco explorada. En la actualidad podemos emplear modelos computacionales para simular el funcionamiento de diferentes componentes de la vida diaria como vehículos, tráfico o el clima con la finalidad de llevar a cabo experimentos sin ningún riesgo inherente. Este trabajo presenta el comportamiento de los modelos U-NET, Mobile-Net y Pix2Pix para la segmentación de caminos en un entorno virtual. Se aprovechan las facultades del entorno virtual para generar la base de datos que compartiremos. Finalmente se muestran los resultados estadísticos de cada modelo empleando como base de comparación la métrica intersección sobre unión (IoU) así como la cantidad de imágenes procesadas por segundo.

Palabras clave: CNN, UNET, Mobile-Net, Pix2Pix Unity, entornos virtuales.

Road Segmentation in Virtual Environments

Abstract. Autonomous vehicles are of great importance in the field of artificial intelligence. Trained neural networks are applied to adapt to unknown environments. Some convolutional neural networks (CNN) had a great impact in different fields of science, such as biology. However, its efficiency in other fields is little explored. At present we can use computational models to simulate the operation of different components of daily life such as vehicles, traffic or the weather in order to carry out experiments without any inherent risk. This work presents the behavior of the U-NET, Mobile-Net and Pix2Pix models for road segmentation in a virtual environment. The faculties of the virtual environment are used to generate the database that we will share. Finally, the statistical results of each model are shown using the intersection over union (IoU) metric as a basis for comparison, as well as the number of images processed per second.

Keywords: CNN, UNET, Mobile-Net, Pix2Pix Unity, virtual environments.

1. Introducción

La segmentación de imágenes es la separación de elementos en la imagen mediante algún proceso de agrupación [18]. Es una de las tareas más importantes en el área de visión por computadora y tiene una gran relevancia en campos de estudio que van desde la medicina hasta la industria bélica [13, 7]. La información contenida en una imagen puede ser reagrupada, dependiendo de los componentes con una misma clasificación o cada componente por separado y esto corresponde al tipo de segmentación empleada que puede ser semántica o por instancias.

La segmentación semántica busca asignar una etiqueta a cada elemento en la imagen agrupando a aquellos con características similares. La segmentación por instancias busca dos cosas la detección de objetos y la asignación de etiquetas a dicho objeto separando incluso a aquellos con características similares. En los últimos años con el uso de las redes neuronales convolucionales (CNN) se ha logrado un gran avance en la tarea de segmentar tanto de manera semántica como por instancias tanto en sistemas con capacidad de computo alto como en sistemas embebidos para su incorporación en vehículos autónomos [15].

El uso de vehículos autónomos de diferente índole, como son aéreos, marítimos o terrestres [16, 19], requieren de la comprensión del medio para determinar las diferentes acciones de control en la detección de caminos u obstáculos. La segmentación de caminos permite separar los diferentes elementos de la imagen, como pueden ser vegetación y cielo del objeto de interés. Para ello se emplean técnicas de agrupamiento o de inteligencia artificial.

Algunos modelos de inteligencia artificial basados en redes convolucionales, han mostrado buenos resultados en tareas de segmentación [8], por ello tienen gran relevancia en la actualidad. Sin embargo, algunos de los modelos fueron diseñados para segmentación de componentes específicos y no se han probado en diferentes tareas.

Uno de los trabajos más relacionados con este artículo es el mostrado en [14], donde se emplea dos redes convolucionales profundas (CNN), dedicadas a la detección de caminos y detección de objetos. La red que se encarga de detectar el camino para determinar la acción necesaria para alinear el centro del camino con el centro de la imagen. Emplea una arquitectura s-ResNet-18 para determinar si el vehículo se encuentra dentro o fuera del camino, así como realizar las acciones necesarias.

El mapeo del área silvestre tiene gran importancia en la comunidad científica, ya que se puede registrar información del deterioro o progreso del medio ambiente. Debido a esto, trabajos como el de [9] empleando un algoritmo de localización y mapeo simultáneo (SLAM), se generan mapas de bosque. Los sensores empleados son un LIDAR (Velodyne VLP-16), una cámara estéreo, una IMU (Unidad de medición inercial) y un GPS (Sistema de posicionamiento global).

El mapa resultante, en nube de puntos 3D, fue evaluado en términos de precisión y exactitud teniendo un error de estimación promedio de 2cm. El algoritmo de mapeo empleado fue graph-SLAM, debido a que reduce la complejidad computacional respecto a un $\mathrm{O}(\mathrm{n}^2)$ del filtro Kalman, que además requiere del mapa m todos los n puntos de referencia. Para el enfoque graph-SLAM cada nodo del grafo representa una posición particular del robot, mientras que los bordes codifican restricciones de odometría o cierres de bucle.

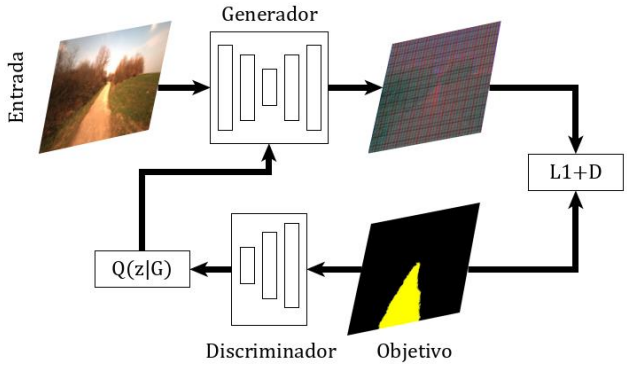


Fig. 1. Arquitectura Pix2Pix basado en [6].

En el trabajo de [1] se busca conocer la especie y el diámetro de vegetación en un bosque, mostrando que métodos no se desempeñan de manera satisfactoria en este tipo de entornos, por ejemplo LIDAR capta mucho ruido. El algoritmo propuesto en este trabajo es SLOAM (Semantic Lidar Odometry and Mapping). El propósito del algoritmo de odometría lidar es estimar el movimiento rígido de 6-DOF (grados de libertad) del lidar dentro de un barrido de 360 grados, mientras que el propósito del algoritmo de mapeo lidar es estimar la posición de 6-DOF del lidar en el mundo, y registrar la nube de puntos en el marco mundial.

El trabajo de [17] también tiene gran relevancia para este proyecto, tanto por la orientación que tomó (ADAS, Sistema Avanzado de Asistencia al Conductor), como por los resultados obtenidos. En este trabajo proponen una combinación de estructuras CNN (Redes Neuronales Convolucionales) y LSTM (Memoria de largo plazo) mostrando previamente el desempeño de ambas por separado. Se probó con la base de datos KITTI [3] obteniendo un precisión promedio de 91.6 %.

Una de las desventajas del modelo propuesto es la reducción de su desempeño en condiciones de poca luz y cuando hay presencia de sombras. En este artículo se presenta tres diferentes arquitecturas de segmentación basadas en redes neuronales convolucionales, entrenadas con una base de datos creada a partir de un entorno virtual realista. También se muestra el desempeño del modelo en tiempo real tomando en cuenta la cantidad de fotogramas procesados por segundo.

2. Modelos de segmentación

En esta sección se describirán los modelos empleados. Estos modelos comparten algunas características que nos llevaron a determinar que la comparación es adecuada. Se tiene contemplado el modelo U-Net como una de las arquitecturas referentes en la tarea de segmentación siendo una de las primeras redes en emplear el concepto de bloques residuales, Mobile-Net es una red basada en bloques residuales optimizada para agilizar operaciones y consumir menos memoria y Pix2Pix emplea como generador una red U-Net.

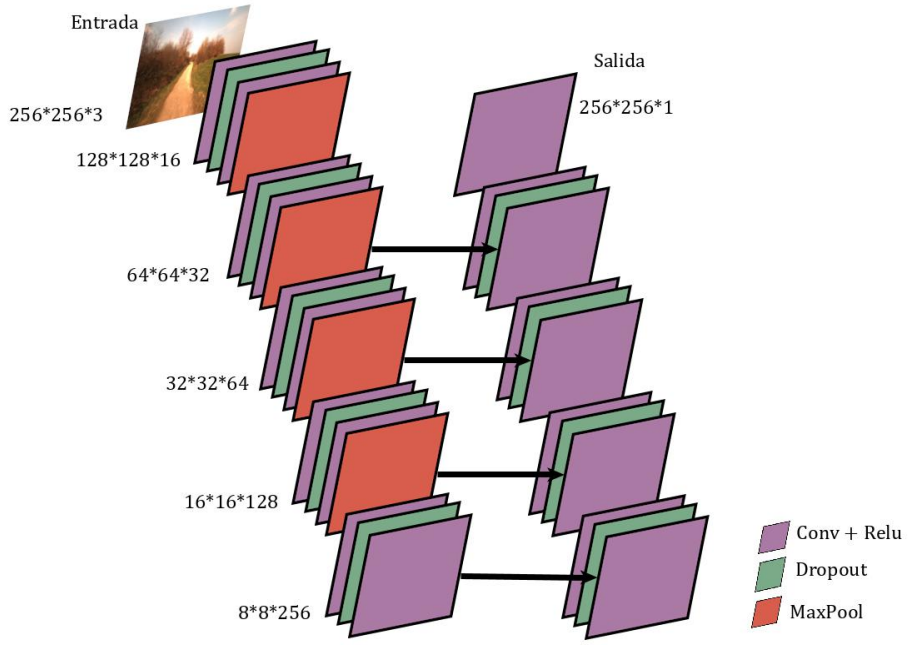


Fig. 2. Arquitectura U-Net basado en [10].

2.1. U-Net

La arquitectura U-Net propuesta en [10] es un modelo encoder-decoder que busca la extracción de características mediante un proceso de contracción y un proceso de expansión. En el proceso de contracción, la imagen reduce su tamaño (ancho y largo) a la mitad, mientras que su dimensionalidad (profundidad) aumenta, en el proceso de expansión el tamaño de la imagen aumenta al doble mientras que la dimensionalidad se reduce.

Este proceso tiende a ser muy agresivo, dado que el tamaño de la imagen se ve afectado hasta reducirlo a un valor muy pequeño. Por lo anterior, en [10] proponen conexiones entre la parte de contracción y la parte de expansión de modo que se pueda rescatar una mayor cantidad de información. En la Fig. 2 se muestra gráficamente la arquitectura U-Net.

La Fig. 2 muestra a la red U-Net como un proceso de dos pasos. En el primero (Contracción) se tiene en la entrada una imagen de $(256, 256, 3)$ a la cual se le aplican una serie de operaciones (Convolución, Dropout, Maxpool), dependiendo del punto de la extracción de características en la que se encuentre con la finalidad de reducir las dimensiones de la imagen y ampliar su profundidad.

En el segundo paso (Expansión), se aplican del mismo modo una serie de operaciones pero en este punto la finalidad es ampliar las dimensiones de la imagen y reducir la profundidad hasta uno. De manera simultánea a cada bloque de la expansión se le concatena un bloque de mismo tamaño de la contracción.

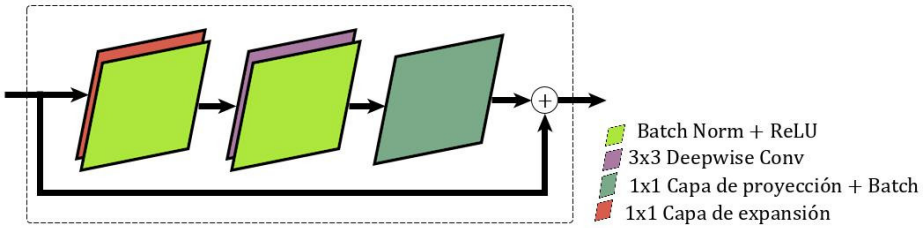


Fig. 3. Arquitectura del bloque de cuello de botella residual basado en [11].

2.2. Pix2Pix

Pix2Pix [6] es una arquitectura dentro del paradigma de redes adversarias generativas (GAN's), es decir, se entrenan simultáneamente dos redes, donde una intentará generar una imagen a partir de una imagen de entrada (red generadora) y la otra intentará diferenciar una imagen generada de una imagen objetivo (red discriminadora). En Fig. 1 se muestra un diagrama de cómo interactúan ambas arquitecturas.

En Fig. 1 se entrenan dos redes de manera simultánea, la red discriminadora se entrena de modo que pueda diferenciar una imagen real (De la base de datos) de una falsa (Generada). La red generadora se entrena de modo que se obtenga una imagen igual a la la imagen Objetivo. El modelo generador está basado en la arquitectura U-Net. El valor de $\mathrm{L1} + \mathrm{D}$ corresponde al valor de pérdida, es decir, la comparación entre la imagen generada y la imagen objetivo. El valor de pérdida ayuda a entrenar al modelo discriminador que a su vez ayuda a entrenar el modelo generador.

2.3. Mobile-Net V2

Mobile-Net V2 [11] es una mejora al trabajo realizado por [5], la cual cuenta con tres capas convolucionales en el bloque. Los dos últimos son los mismos que los mostrados en Mobile-Net V1: una convolución en profundidad que filtra las entradas, seguida de una capa de convolución puntual 1x1. Sin embargo, esta capa 1x1 ahora tiene una función diferente. En Mobile-Net V1, la convolución puntual mantuvo el mismo número de canales o los duplicó. En V2 hace lo contrario, reducir el número de canales.

Esta es la razón por la que esta capa ahora se conoce como la capa de proyección: proyecta datos con un gran número de dimensiones (canales) en un tensor con un número mucho menor de dimensiones. La primer capa del bloque también es una convolución 1×1. Su propósito es expandir el número de canales en los datos antes de que entren en la convolución en profundidad.

Por lo tanto, esta capa de expansión siempre tiene más canales de salida que canales de entrada, y hace prácticamente lo contrario de la capa de proyección. La segunda novedad del componente básico de Mobile-Net V2 es la conexión residual. Esto funciona como en ResNet [4] y existe para ayudar con el flujo de gradientes a través de la red. La conexión residual sólo se utiliza cuando el número de canales que entran en el bloque es el mismo que el número de canales que salen de él.

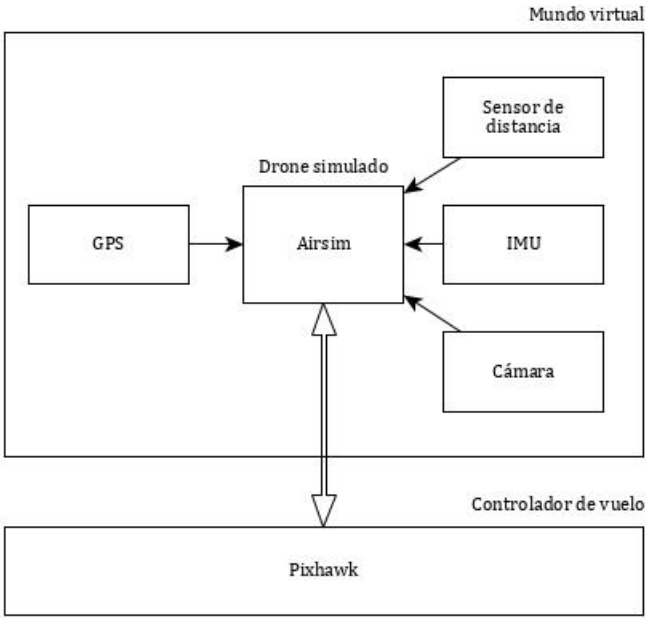


Fig. 4. Interacción entre los componentes del sistema.

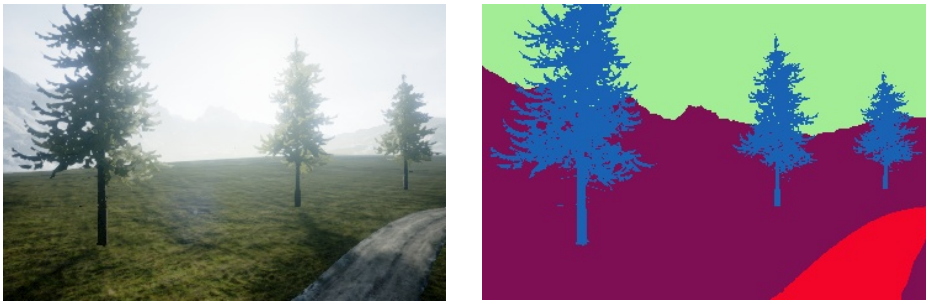
En Fig. 3 muestra gráficamente el bloque de cuello de botella residual de Mobile-Net V2. En Fig. 3 se muestra el bloque de cuello de botella en el cual a una imagen de entrada se le aplican una serie de operaciones con la finalidad de reducir la dimensionalidad de la imagen y ampliar la profundidad para después regresar a sus dimensiones originales para finalmente concatenar la entrada del bloque con la salida del mismo, esto con la finalidad de resaltar las características deseadas sin pérdida de información.

3. Materiales y métodos

En esta sección se mostrarán los diferentes pasos para la obtención de un modelo basado en redes neuronales convolucionales funcional, sobre una simulación en Unity. Los pasos mostrados abarcaran desde la realización del entorno de pruebas, pasando por la generación de la base de datos, hasta el entrenamiento y las pruebas.

3.1. Descripción del sistema

Para asegurar una plataforma estable para realizar diferentes pruebas se opta por realizarlas sobre un entorno virtual realista. El sistema se compone por un mundo virtual empleando como base los componentes de Landscape Mountain proporcionado por [2] y un simulador de drones [12] que funciona sobre el mundo virtual. El simulador de drones tiene compatibilidad con diferentes controladores de vuelo externos.



(a) Escenario (b) Segmentación

Fig. 5. Imágenes que componen la base de datos.

Se creó un componente externo al entorno suministrado por Unreal Engine (UE4). Dicho componente está basado en splines y ayuda a generar un camino de manera simple. Únicamente tiene opciones como crear una spline cerrada o abierta, mantener visible la spline o el camino. Este componente fue necesario ya que los caminos generados en el Landscape son parte del mismo y no pueden ser segmentados.

El drone simulado es un cuadricóptero que emplea diferentes sensores como lo son Lidar, IMU, GPS, sensores, cámara de distancia, entre otros. Para las pruebas se empleó un sensor de distancia orientado a -90 grados en el eje de vuelo de elevación (cabeceo), para determinar la altura del drone, respecto al piso en la simulación, dado que la posición del drone es relativa al centro del mapa.

También se incorporó un controlador de vuelo PixHawk 2.4.8, empleando HIL (hardware-in-the-loop), con la finalidad de incorporar mayor estabilidad en el vuelo y para incorporar un transmisor de radio. El controlador de vuelo va conectado via USB a un puerto COM, la comunicación entre la simulación y el controlador de vuelo requiere de un programa extra que interprete los comandos en el protocolo MAVLink.

El controlador de vuelo se encarga de realizar la estimación en la posición del drone a partir de los diferentes sensores incorporados, y ejecuta un filtro Kalman extendido (EKF) para la realización de dicha estimación. Otra función del controlador de vuelo es determinar los valores de PWM suministrados a los motores en la simulación, ya que para las diferentes acciones se requiere un comportamiento especifico de los motores. En Fig. 4 se muestra las conexiones entre el simulador de drones, el mundo virtual y el controlador de vuelo.

En Fig. 4 se muestra como los sensores que emplea el simulador de drones adquiere las mediciones del mundo virtual, es decir que las lecturas del GPS estarán reverenciadas a un punto del mundo virtual que en este caso es el punto $(0, 0, 0)$. Estas mediciones se envían al controlador de vuelo el cual regresa la estimación de la posición de la aeronave así como las señales de los motores.

La cámara incorporada en el drone simulado tiene como características un ángulo de visión de -90 grados, orientado a $(0, 0, 0)$ grados respecto al cuerpo de la aeronave, velocidad de exposición de 100 y gama objetivo de $1{,}5$. Las imágenes que arroja son de segmentación de las mallas en la imagen y la imagen del panorama en un formato RGB y un tamaño de 360×420 pixeles.

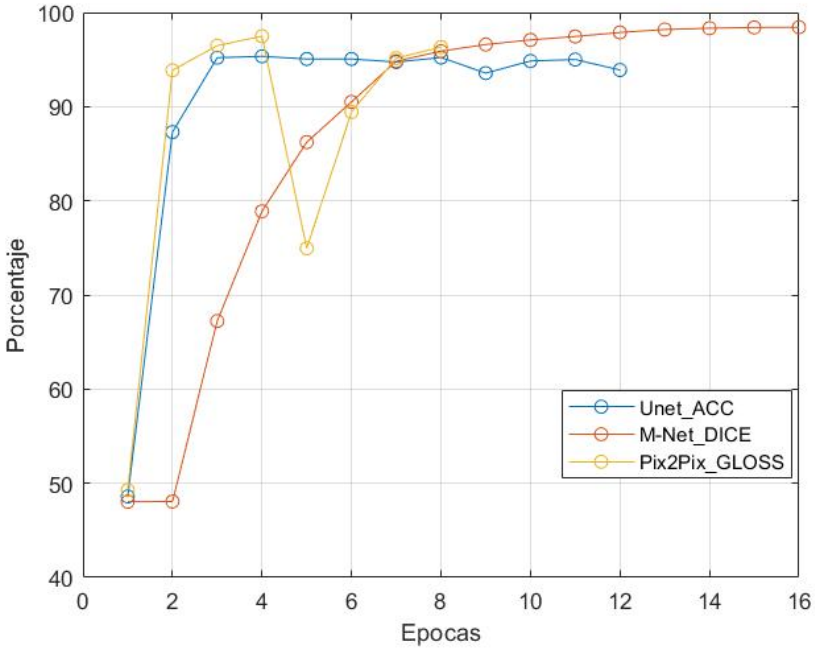


Fig. 6. Entrenamiento de las tres arquitecturas.

3.2. Generación de base de datos

La base de datos se genera a partir de las imágenes de escenario y segmentación. Se ejecuta el mundo virtual y la simulación de drone para posteriormente llevar a cabo la adquisición de dichas imágenes y almacenarlas en formato JPG. Adicional a esto se almacena la posición de la aeronave en los ejes $\mathrm{x}, \mathrm{y}, \mathrm{z}$ respecto al centro del mundo virtual.

Se tomaron cuatro fases para la generación de la base de datos. En una se intentaba que el centro del camino y el centro de la imagen se mantuvieran alineados (funcionamiento óptimo), en otra se procuraba que el centro del camino se mantuviera en el extremo izquierdo o en el extremo derecho y la tercer fase se procuraba que el camino no apareciera en la imagen.

En total se generaron 5000 imágenes con la características antes mencionadas y un mapa de puntos del recorrido en tres ejes. En Fig. 5 se muestra tanto la imagen de escenario como su segmentación.

En Fig. 5a podemos apreciar diferentes componentes que afectan la imagen, como son iluminación del sol en un punto del panorama y sombras de los diferentes componentes. En Fig. 5b se aprecian cuatro componentes segmentados, siendo el landscape el más extenso, ya que abarca todo el mundo virtual y únicamente los componentes compuestos por mallas serán segmentados.

3.3. Entrenamiento

Se entrenaron las tres arquitecturas U-Net, Mobile-Net V2 y Pix2Pix con el conjunto de datos separado en entrenamiento y pruebas. Para el entrenamiento de la red U-Net se empleó como función de pérdida entropía cruzada con pérdida logística (BCEWithLogitsLoss) descrita en 3.3:

$$\ell(x, y) = L = \{l_1, ..., l_N\}, \tag{1}$$
$$l_n = -w_n[y_n * \log \sigma(x_n) + (1 - y_n) * \log(1 - \sigma(x_n))].$$

Para el entrenamiento de la red Mobile-Net se empleó como función de pérdida el coeficiente Dice el cual está denotado por la ecuación 2:

$$D_c = \frac{2 * |A| \cap |B|}{|A| + |B|}. \tag{2}$$

La arquitectura Pix2Pix tiene como función objetivo 3:

$$G^* = \text{argmin}_G \max_D \mathcal{L}_{CGAN}(G, D) + \lambda \mathcal{L}_{L1}(G). \tag{3}$$

Y la función de pérdida esta denotada por 5:

$$\mathcal{L}(G, D)) = \mathbb{E}_{x,y}[\log D(x, y)] + \mathbb{E}_{x,y}[\log(1 - D(x, G(x, z)))], \tag{4}$$
$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|], \tag{5}$$

donde G y D son los resultados de los modelos generador y discriminador respectivamente.

4. Resultados

Los resultados se muestran comparando el desempeño de las diferentes arquitecturas y una transformación de color y saturación efectuada a la misma imagen de entrada, también se muestra el comportamiento de las métricas empleadas como función de pérdida.

4.1. Entrenamiento

El entrenamiento se llevo a cabo con el mismo conjunto de datos separados de la misma forma 4900 imágenes de entrenamiento y 100 imágenes de validación. En Fig. 6 se muestra parte del proceso de entrenamiento de las tres arquitecturas. Las tres arquitecturas mostraron una pérdida cercana a cero pero con valores mayores a los documentados con bases de datos conocidas. El tiempo de entrenamiento de cada uno de los modelos es de aproximadamente 5 horas.

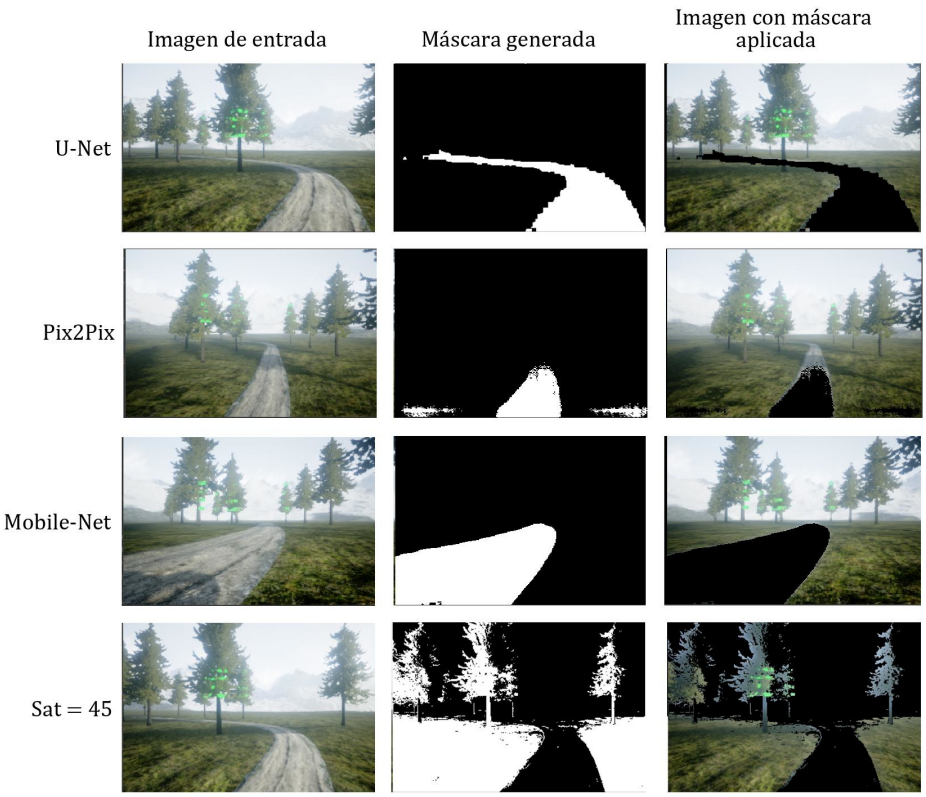


Fig. 7. Comparación de desempeño de las tes arquitecturas y el ajuste a saturación.

4.2. Pruebas

En esta sección se mostrará una comparativa entre las diferentes arquitecturas, además se comparará con la eficiencia de un sistema no basado en redes convolucionales. Para estas pruebas se empleo un trayecto diferente al trayecto empleado para generar la base de datos, este nuevo conjunto consta de 1500 imágenes. En Fig. 7 se muestra la comparación de las diferentes arquitecturas funcionando en tiempo real sobre la simulación en el trayecto de pruebas.

En la columna Imagen de entrada de Fig. 7 se tiene una imagen RGB de $(240 \times 360 \times 3)$, en la columna Máscara generada se tiene una imagen de $(240 \times 360 \times 1)$ resultado de cada modelo y en la columna final se tiene el resultado de aplicar la máscara a la imagen de entrada. El renglón $\mathrm{Sat} = 45$ es el resultado de aplicar una técnica clásica para segmentación de imágenes, el proceso consta en variar los parámetros de saturación, matiz y valor a una imagen en formato .hsv.

El desempeño se evaluó a partir de la métrica intersección sobre unión (IoU) mostrada en 6, también se evaluó el desempeño midiendo la cantidad de imágenes que cada arquitectura podía procesar en un segundo. En Tabla 1 se muestra el promedio,

Tabla 1. Imágenes procesadas por segundo e IoU.

Modelo	**Min.**	**Max.**	**Prom.**	**IoU**
U-Net	7.62	8.48	8.33	0.76
Mobile-Net	6.63	7.75	7.02	0.98
Pix2Pix	11.86	13.2	12.37	0.46
Sat=45	13.12	14.52	13.61	0.11

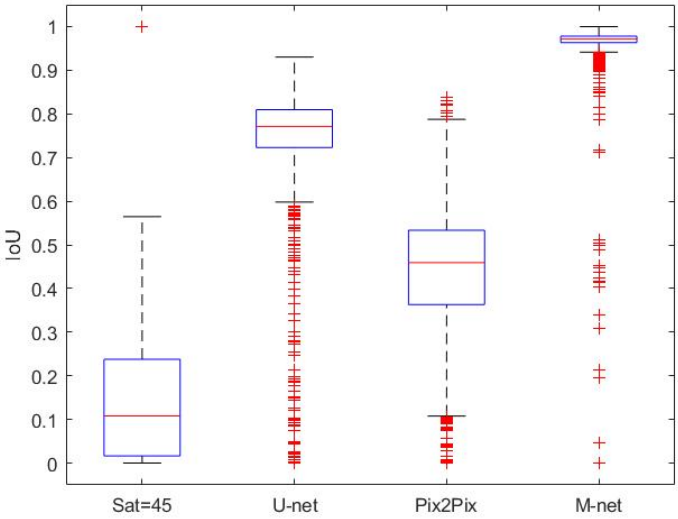


Fig. 8. Gráfica de caja de la métrica IoU sobre la base de datos de prueba.

máximo y mínimo de imágenes procesadas por segundo así como el promedio de IoU sobre las 1500 imágenes de prueba.

Del mismo modo en la Figura 8 se muestra la gráfica de caja de los datos obtenidos con la métrica IoU:

$$\text{IoU} = \frac{A \cap B}{A \cup B}. \tag{6}$$

5. Conclusiones

Se mostró el resultado al evaluar tres arquitecturas sobre una simulación funcionando en tiempo real, las tres arquitecturas tienen características similares U-Net [10] emplea bloques residuales, Mobile-Net [11] también emplea bloques residuales y Pix2Pix [6] emplea una U-Net como generador.

La arquitectura con mayor desempeño empleando como comparación IoU es Mobile-Net sin embargo creemos que la arquitectura U-Net podría alcanzar un desempeño similar si la imagen de entrada tiene mayor resolución de modo que los procesos de contracción y expansión cuenten con una cantidad mayor de bloques.

También se mostró una comparación de las tres arquitecturas contra una de las técnicas clásicas de segmentación desarrollada con un algoritmo simple en OpenCv.

En las diferentes pruebas pudimos notar que este algoritmo es susceptible a los elementos fuera del camino como lo son sombras o arboles por ello creemos que el uso de algoritmos basados en CNN en segmentación esta justificado.

Uno de los factores que más resaltan es la cantidad de imágenes que pueden ser procesadas por segundo ya que este factor es fundamental para el correcto funcionamiento de un vehículo autónomo por ello se propone evaluar las tres arquitecturas al recorrer un trayecto empleando las mismas condiciones.

Este trabajo un paso crucial en un proyecto de vehículos autónomos cuyos avances representan la validación y desarrollo de vehículos autónomos en entornos fuera de riesgo ya que combina el funcionamiento convencional de un drone con una simulación en tiempo real propiciando la integración de técnicas de inteligencia artificial. Finalmente la base de datos se encuentra disponible para el uso de la comunidad en general.

Referencias

1. Chen, S. W., Nardari, G. V., Lee, E. S., Qu, C., Liu, X., Romero, R. A. F., Kumar, V.: SLOAM: Semantic lidar odometry and mapping for forest inventory. IEEE Robotics and Automation Letters, vol. 5, no. 2, pp. 612–619 (2020) doi: 10.48550/ARXIV.1912.12726
2. Epic Games: Unreal engine, https://www.unrealengine.com
3. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for autonomous driving? the KITTI vision benchmark suite. In: Proceedings of the Conference on Computer Vision and Pattern Recognition. pp. 3354–3361 (2012) doi: 10.1109/CVPR.2012.6248074
4. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016) doi: 10.48550/ARXIV.1512.03385
5. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications, (2017) doi: 10.48550/ARXIV.1704.04861
6. Isola, P., Zhu, J. Y., Zhou, T., Efros, A. A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1125–1134 (2017) doi: 10.48550/ARXIV.1611.07004
7. Kim, J. U., Kim, H. G., Ro, Y. M.: Iterative deep convolutional encoder-decoder network for medical image segmentation. In: Proceedings of the 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society. pp. 685–688 (2017) doi: 10.1109/embc.2017.8036917
8. Mortazi, A., Bagci, U.: Automatically designing CNN architectures for medical image segmentation. In: Proceedigns of the International Workshop on Machine Learning in Medical Imaging. pp. 98–106 (2018) doi: 10.48550/ARXIV.1807.07663
9. Pierzchała, M., Giguère, P., Astrup, R.: Mapping forests using an unmanned ground vehicle with 3d LiDAR and graph-SLAM. Computers and Electronics in Agriculture, vol. 145, pp. 217–225 (2018) doi: 10.1016/j.compag.2017.12.034
10. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241 (2015) doi: 10.48550/ARXIV.1505.04597

11. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L. C.: MobileNetV2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4510–4520 (2018) doi: 10.48550/ARXIV.1801.04381
12. Shah, S., Dey, D., Lovett, C., Kapoor, A.: Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In: Field and Service Robotics (2017), https://arxiv.org/abs/1705.05065
13. Shan, P.: Image segmentation method based on k-mean algorithm. EURASIP Journal on Image and Video Processing, vol. 2018, no. 1, pp. 81 (2018) doi: 10.1186/s13640-018-0322-6
14. Smolyanskiy, N., Kamenev, A., Smith, J., Birchfield, S.: Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems. pp. 4241–4247 (2017) doi: 10.48550/ARXIV.1705.02550
15. Sultana, F., Sufian, A., Dutta, P.: Evolution of image segmentation using deep convolutional neural network: A survey. Knowledge-Based Systems, vol. 201 (2020) doi: 10.1016/j.knosys.2020.106062
16. Wang, Y., Peng, M., Di, K., Wan, W., Liu, Z., Yue, Z., Xing, Y., Mao, X., Teng, B.: Vision based obstacle detection using rover stereo images. International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences, vol. XLII-2/W13, pp. 1471–1477 (2019) doi: 10.5194/isprs-archives-xlii-2-w13-1471-2019
17. Yang, X., Li, X., Ye, Y., Lau, R. Y., Zhang, X., Huang, X.: Road detection and centerline extraction via deep recurrent convolutional neural network u-net. IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 9, pp. 7209–7220 (2019) doi: 10.1109/tgrs.2019.2912301
18. Yuheng, S., Hao, Y.: Image segmentation algorithms overview, (2017) doi: 10.48550/ARXIV.1707.02051
19. Zhou, H., Kong, H., Wei, L., Creighton, D., Nahavandi, S.: On detecting road regions in a single UAV image. IEEE Transactions on Intelligent Transportation Systems, vol. 18, no. 7, pp. 1713–1722 (2016) doi: 10.1109/tits.2016.2622280

Electronic edition
Available online: http://www.rcs.cic.ipn.mx

http://rcs.cic.ipn.mx

Centro de Investigación
en Computación