Sistema de moderación automática de contenido basado en PLN para entornos que implementan chats en tiempo real

Jonathan Fernández Córdova

TecNM/Instituto Tecnológico de Veracruz,
Departamento de Sistemas y Computación, Veracruz, Ver.,
México

L25020480@veracruz.tecnm.mx

Resumen. Este trabajo presenta el diseño e implementación de un sistema de moderación automática de contenido basado en técnicas de Procesamiento de Lenguaje Natural (PLN) para entornos de chat en tiempo real. El sistema propuesto tiene como objetivo detectar y clasificar mensajes generados por los usuarios en dos categorías: discurso de odio (hate) y contenido no dañino (no hate), con el fin de mejorar la calidad y seguridad de las interacciones digitales. El modelo fue entrenado utilizando múltiples datasets públicos en español e inglés, lo que permite un enfoque multilingüe y una mayor capacidad de generalización. Se emplean técnicas de aprendizaje automático basadas en modelos de lenguaje preentrenados para la clasificación de texto. Además, el sistema se integra con un bot de moderación capaz de analizar mensajes en tiempo real y aplicar acciones automáticas según la clasificación obtenida. Los resultados experimentales muestran un desempeño sólido en la identificación de contenido dañino, reduciendo la necesidad de moderación manual y permitiendo soluciones escalables para plataformas digitales.

Palabras clave: Procesamiento del lenguaje natural, clasificación de texto, redes neuronales, detección de discurso de odio, moderación de chats.

NLP-based Automated Content Moderation System for Environments Implementing Real-time Chats

Abstract. This work presents the design and implementation of an automatic content moderation system based on Natural Language Processing (NLP) techniques for real-time chat environments. The proposed system aims to detect and classify messages generated by users into two categories: hate speech (hate) and non-harmful content (no hate), in order to improve the quality and security of digital interactions. The

model was trained using multiple public datasets in Spanish and English, allowing for a multilingual approach and greater generalizability. Machine learning techniques based on pre-trained language models are used for text classification. In addition, the system is integrated with a moderation bot capable of analyzing messages in real time and applying automatic actions according to the classification obtained. Experimental results show strong performance in identifying harmful content, reducing the need for manual moderation and enabling scalable solutions for digital platforms.

Keywords: Natural language processing, text classification, neural networks, hate speech detection, chat moderation.

1. Introducción

En los últimos años, especialmente tras la pandemia de COVID-19, las plataformas de transmisión en vivo han experimentado un crecimiento exponencial, consolidándose como uno de los principales medios de interacción digital en tiempo real [1]. Servicios de streaming y redes sociales han permitido a millones de usuarios participar activamente mediante sistemas de chat en vivo, donde se intercambian opiniones, comentarios e ideas de manera instantánea. Este fenómeno ha transformado la forma en que las comunidades digitales se comunican, generando espacios altamente dinámicos y participativos.

Sin embargo, este incremento en la interacción también ha traído consigo importantes desafíos relacionados con la moderación del contenido. La aparición frecuente de mensajes ofensivos o de odio representa un problema significativo, ya que puede afectar la experiencia de los usuarios y deteriorar la calidad de las comunidades en línea. La moderación manual resulta insuficiente ante el volumen y la velocidad de los mensajes generados en chats de alta concurrencia. Se estima que transmisiones con miles de espectadores requieren múltiples moderadores para mantener el control del flujo de mensajes, lo que incrementa los costos operativos y limita la escalabilidad de este enfoque.

En este contexto, el uso de técnicas de Procesamiento de Lenguaje Natural (PLN) ha emergido como una solución prometedora para la detección automática de contenido dañino. Los avances en aprendizaje automático y redes neuronales han permitido desarrollar modelos capaces de comprender y clasificar texto con un alto grado de precisión [2], facilitando la automatización de tareas que anteriormente dependían exclusivamente de la intervención humana.

A pesar de estos avances, la integración de sistemas de moderación automática en entornos de chat en tiempo real continúa representando un desafío, debido a la necesidad de procesar grandes volúmenes de datos de forma eficiente y con baja latencia. Asimismo, es fundamental que estos sistemas no solo detecten contenido inapropiado, sino que también sean capaces de ejecutar acciones correctivas de manera inmediata dentro de la plataforma.

En este trabajo se propone un sistema de moderación automática basado en PLN, diseñado para analizar y clasificar mensajes en tiempo real en una

tarea binaria: discurso de odio (*hate*) y contenido no dañino (*no hate*). Esta formulación reduce la ambigüedad semántica presente en enfoques multiclase y mejora la consistencia del modelo en la detección de contenido dañino. El sistema considera un enfoque bilingüe, permitiendo el análisis de mensajes en español e inglés, lo cual amplía su aplicabilidad en entornos digitales globales. El modelo de clasificación se integra con un bot que permite aplicar acciones automatizadas dentro del chat, como advertencias o sanciones.

Como contribución principal, esta investigación presenta una solución escalable que combina modelos de PLN con mecanismos de acción en tiempo real, permitiendo mejorar la calidad de la interacción en plataformas digitales. El objetivo es reducir la dependencia de la moderación manual y contribuir a la creación de entornos digitales más seguros, eficientes y sostenibles.

El resto de este documento se organiza de la siguiente manera: la Sección 2 presenta el trabajo relacionado con la temática abordada; la Sección 3 describe los materiales y métodos utilizados en el desarrollo del sistema, incluyendo el dataset, el preprocesamiento y el modelo empleado; la Sección 4 expone los resultados obtenidos y su respectivo análisis; finalmente, la Sección 5 presenta las conclusiones del trabajo y posibles líneas de investigación futura.

2. Trabajo relacionado

En los últimos años, la detección automática de contenido tóxico y discurso de odio ha sido ampliamente estudiada dentro del campo del Procesamiento de Lenguaje Natural (PLN), impulsada por el crecimiento de las plataformas digitales y la necesidad de moderar grandes volúmenes de texto en tiempo real. Diversos enfoques han sido propuestos, desde métodos tradicionales de aprendizaje automático hasta modelos basados en arquitecturas profundas.

Uno de los avances más relevantes en esta área es el uso de modelos de lenguaje preentrenados como BERT [2], los cuales han demostrado un alto desempeño en tareas de clasificación de texto al capturar el contexto semántico de las palabras. Posteriormente, modelos más robustos como RoBERTa [10] han optimizado el proceso de preentrenamiento, logrando mejoras significativas en múltiples tareas de PLN, incluyendo la detección de contenido ofensivo.

Asimismo, variantes multilingües como Multilingual BERT han permitido extender estas capacidades a múltiples idiomas, facilitando la detección de contenido dañino en contextos globales.

En cuanto a los datos utilizados para el entrenamiento, iniciativas como el Toxic Comment Classification Challenge de Jigsaw [3] han impulsado el desarrollo de modelos robustos mediante el uso de datasets a gran escala.

Por otro lado, estudios como el de Davidson et al. [11] evidencian la dificultad de distinguir entre lenguaje ofensivo y discurso de odio, señalando que ambos conceptos no son equivalentes y pueden generar ambigüedades en los modelos de clasificación. Esta problemática resalta la importancia de una adecuada definición de las etiquetas en sistemas de moderación automática.

En esta misma línea, Kolhatkar et al. [12] proponen la clasificación de comentarios constructivos, destacando la relevancia de diferenciar entre críticas útiles y contenido dañino. Este enfoque permite enriquecer los sistemas de moderación al considerar no solo la toxicidad, sino también la intención comunicativa del mensaje.

Más recientemente, Mathew et al. [13] introducen HateXplain, un dataset que incorpora anotaciones humanas sobre las razones detrás de una clasificación, promoviendo el desarrollo de modelos explicables. Este tipo de enfoques resulta especialmente relevante en aplicaciones reales, donde la transparencia y la interpretabilidad son fundamentales.

Adicionalmente, modelos basados en redes neuronales recurrentes, como Long Short-Term Memory (LSTM) [4], han sido utilizados para la clasificación de texto, logrando resultados competitivos al capturar dependencias secuenciales en el lenguaje. Sin embargo, estos enfoques presentan limitaciones en términos de eficiencia y escalabilidad frente a modelos basados en transformers.

Finalmente, diversos estudios de revisión [6] han analizado el estado del arte en la detección de discurso de odio, destacando que este problema no depende únicamente de palabras explícitas, sino también del contexto, la intención y factores culturales, lo cual representa un desafío significativo para los sistemas automáticos.

A pesar de estos avances, la mayoría de los trabajos existentes se centran únicamente en la detección de contenido dañino, sin considerar la integración de sistemas capaces de ejecutar acciones automáticas en tiempo real dentro de entornos de chat. En este sentido, el presente trabajo propone un enfoque integral que combina un modelo de clasificación bilingüe basado en PLN con un sistema de moderación automatizado, permitiendo no solo identificar contenido inapropiado, sino también actuar de manera inmediata dentro de la plataforma.

3. Materiales y métodos

En esta sección se describen los materiales y métodos utilizados para el desarrollo del sistema de moderación automática basado en técnicas de Procesamiento de Lenguaje Natural (PLN).

3.1. Dataset

El conjunto de datos fue construido mediante la integración de múltiples datasets públicos ampliamente utilizados en tareas de detección de discurso de odio y análisis de sentimiento, con el objetivo de garantizar diversidad lingüística, robustez y representatividad del problema.

Para el idioma español, se emplearon los datasets Spanish Hate Speech Superset y TASS 2019 [15,16], los cuales contienen ejemplos de lenguaje ofensivo y análisis de sentimiento. En el caso de TASS, las etiquetas originales fueron adaptadas a una formulación binaria (hate / no hate) para mantener consistencia con el enfoque del modelo.

Asimismo, se incorporó el dataset HatEval [17], enfocado en la detección de discurso de odio en redes sociales, particularmente contra grupos vulnerables.

Para el idioma inglés, se utilizó el dataset Hate Speech and Offensive Language [11], así como el dataset Stanford Sentiment Treebank (SST-2) [18], el cual contribuye a mejorar la comprensión contextual del modelo en textos no ofensivos.

Tabla 1. Datasets utilizados

Dataset	Idioma	Tipo
Spanish Hate Speech Superset	Español	Hate Speech
TASS 2019	Español	Sentimiento adaptado
HatEval	Español/Inglés	Hate Speech
Hate Speech Offensive Language	Inglés	Hate / Offensive
SST-2	Inglés	Sentimiento

El dataset final está compuesto por aproximadamente 56,000 instancias textuales, integrando datos en español e inglés. Esta combinación permite entrenar un modelo bilingüe capaz de generalizar en distintos contextos lingüísticos.

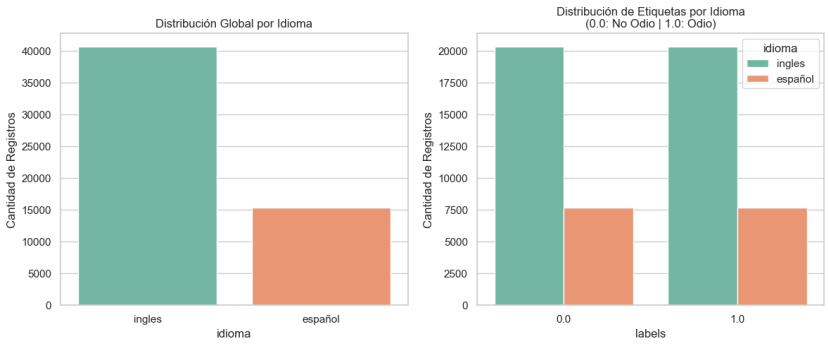

Fig. 1. Distribución del dataset por idioma y etiquetas (0: no hate, 1: hate).

Como se muestra en la Figura 1, se observa una mayor proporción de datos en inglés en comparación con el español. No obstante, la distribución de etiquetas se mantiene balanceada entre las clases de discurso de odio (hate) y contenido no dañino (no hate), lo cual favorece el entrenamiento del modelo.

Asimismo, el análisis de la longitud de los textos revela diferencias entre idiomas. En general, los textos en español presentan una mayor cantidad de palabras y caracteres en comparación con los textos en inglés, lo cual introduce un desafío adicional en términos de generalización del modelo.

El conjunto de datos fue dividido en tres subconjuntos: entrenamiento (70 %), validación (15 %) y prueba (15 %). Esta partición permite entrenar el modelo,

ajustar hiperparámetros y evaluar su desempeño en datos no vistos. La división se realizó de manera estratificada para preservar la distribución de clases en cada subconjunto.

Tabla 2. Distribución por idioma

Idioma	Porcentaje
Inglés	72 %
Español	28 %

3.2. Preprocesamiento

Los datos fueron sometidos a un proceso de limpieza que incluyó la eliminación de caracteres especiales, normalización del texto y tokenización. Posteriormente, los textos fueron convertidos a identificadores numéricos (input IDs) y se generaron máscaras de atención (attention masks).

Como se muestra en la Tabla 3, el proceso de tokenización transforma el texto original en una representación estructurada que puede ser interpretada por el modelo.

Tabla 3. Ejemplo del proceso de tokenización

Etapa	Resultado
Texto original	Ï hate this game"
Tokenización	[Ï", "hate", "this", "game"]
Input IDs	[101, 5223, 2023, 2208, 102]
Attention Mask	[1, 1, 1, 1, 1]

3.3. Modelo

Se utilizó el modelo *bert-base-multilingual*, basado en la arquitectura de transformers [2]. Para su implementación se empleó la biblioteca PyTorch [7], junto con la librería Transformers de HuggingFace [8], la cual facilita la carga y ajuste fino de modelos preentrenados.

Este modelo cuenta con un vocabulario de aproximadamente 119,547 tokens y más de 90 millones de parámetros, lo que le permite procesar múltiples idiomas y capturar relaciones contextuales complejas en el texto.

Durante el entrenamiento, se incorporó una capa de regularización mediante *Dropout*, con el objetivo de reducir el sobreajuste y mejorar la capacidad de generalización del modelo. El modelo fue entrenado utilizando el optimizador AdamW durante 10 épocas, determinando este valor de manera empírica mediante el monitoreo del conjunto de validación.

Desde el punto de vista matemático, BERT recibe como entrada una secuencia de tokens:

$$X = (x_1, x_2, ..., x_n).$$

Cada token es representado mediante la suma de embeddings de token, posición y segmento:

$$E_i = E_i^{token} + E_i^{position} + E_i^{segment}.$$

Como se muestra en la Figura 2, la representación de entrada integra estas tres componentes, permitiendo al modelo capturar información contextual tanto a nivel léxico como estructural [2].

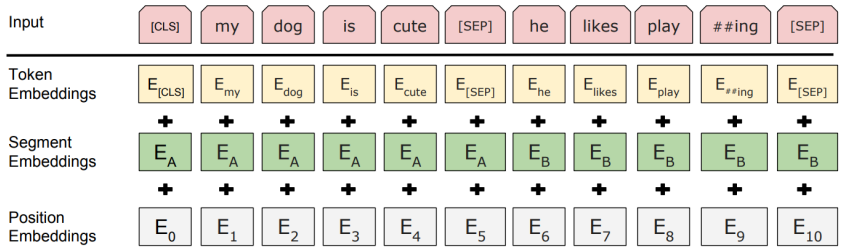

Fig. 2. Representación de entrada en BERT. Los embeddings de entrada se obtienen como la suma de los embeddings de token, posición y segmento. Adaptado de [2].

Estas representaciones se procesan a través de capas de autoatención, definidas como [14]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V.$$

donde Q, K y V representan las matrices de consultas, claves y valores, respectivamente.

El modelo utiliza atención multi-cabeza para capturar diferentes relaciones semánticas [14]:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(head_1, \ldots, head_h)W^O.$$

donde cada cabeza se define como:

$$head_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V).$$

Para la tarea de clasificación binaria (*hate / no hate*), se utiliza la representación del token especial $[CLS]$, siguiendo el esquema de clasificación propuesto en BERT [2]:

$$\hat{y} = \text{softmax}(Wh_{[CLS]} + b),$$

donde $h_{[CLS]}$ corresponde a la representación final del token $[CLS]$, y W, b son los parámetros entrenables de la capa de salida.

3.4. Configuración de entrenamiento

El modelo utilizado fue *bert-base-multilingual-cased*, basado en la arquitectura Transformer encoder y compuesto por 12 capas, 12 cabezas de atención y una dimensión oculta de 768 características. Se realizó un proceso de *fine-tuning* completo, sin congelamiento de capas, permitiendo la actualización de todos los parámetros del modelo durante el entrenamiento.

La representación contextual obtenida del token especial $[CLS]$ fue utilizada como entrada de una capa lineal de clasificación binaria. Antes de la capa de salida, se aplicó regularización mediante *Dropout* con probabilidad $p = 0{,}4$, con el objetivo de reducir el sobreajuste durante el ajuste fino.

La tokenización se realizó mediante *BertTokenizer*, utilizando truncamiento y *padding* hasta una longitud máxima de secuencia de 150 tokens. Cada entrada fue transformada en *input IDs* y *attention masks*, preservando el formato requerido por el modelo BERT.

El conjunto de datos fue dividido de manera estratificada en entrenamiento (70 %), validación (15 %) y prueba (15 %), manteniendo la distribución original de las clases en cada subconjunto.

El entrenamiento se ejecutó utilizando un tamaño de lote (*batch size*) de 32 muestras durante un máximo de 7 épocas. La función de pérdida utilizada fue *CrossEntropyLoss*, adecuada para tareas de clasificación multiclase y binaria basadas en logits.

Para la optimización de parámetros se utilizó el algoritmo AdamW [2], configurado con una tasa de aprendizaje inicial de 2×10^{-5} y un término de regularización L_2 (*weight decay*) de 0.01. El scheduler empleado fue *linear learning rate decay* sin fase de warmup, calculado en función del número total de pasos de entrenamiento:

$$\text{total\_steps} = |\mathcal{D}_{train}| \times \text{epochs},$$

donde $|\mathcal{D}_{train}|$ representa el número de lotes del conjunto de entrenamiento.

Durante la retropropagación se aplicó *gradient clipping* con norma máxima igual a 1.0:

$$\|\nabla\theta\|_2 \leq 1{,}0$$

con el objetivo de estabilizar el entrenamiento y prevenir explosión de gradientes.

Asimismo, se implementó entrenamiento en precisión mixta (*Automatic Mixed Precision, AMP*) mediante `torch.amp.autocast` y `GradScaler`, reduciendo el consumo de memoria GPU y acelerando el tiempo de entrenamiento.

El proceso de entrenamiento incorporó una estrategia de *Early Stopping* basada en la pérdida de validación. El entrenamiento finalizaba automáticamente cuando no se observaban mejoras durante 2 épocas consecutivas (*patience* = 2). Además, se almacenó el estado correspondiente al menor valor de pérdida de validación para restaurar posteriormente el mejor modelo obtenido.

Finalmente, el desempeño del modelo fue evaluado mediante las métricas *accuracy* y *macro F1-score*, calculadas sobre el conjunto de prueba, complementadas con matriz de confusión y reporte de clasificación para analizar el comportamiento del modelo en ambas clases.

3.5. Sistema de Implementación

El sistema fue desarrollado en Python 3.10.20, utilizando un enfoque modular orientado a la integración de modelos de Procesamiento de Lenguaje Natural (PLN) en entornos de moderación automática en tiempo real.

La inferencia del modelo se implementó mediante FastAPI, proporcionando una arquitectura basada en servicios REST para el procesamiento de mensajes en tiempo real. La API recibe mensajes de texto, ejecuta el proceso de tokenización utilizando *BertTokenizer* y posteriormente realiza la inferencia utilizando el modelo *bert-base-multilingual-cased* ajustado mediante fine-tuning.

La integración con plataformas de streaming se realizó mediante TwitchIO, una biblioteca asincrónica para la interacción con el protocolo IRC de Twitch. El bot desarrollado monitorea continuamente los mensajes publicados en el chat y envía cada instancia textual al sistema de inferencia para su clasificación automática.

El flujo operacional del sistema se compone de las etapas presentadas en la Fig. 3.

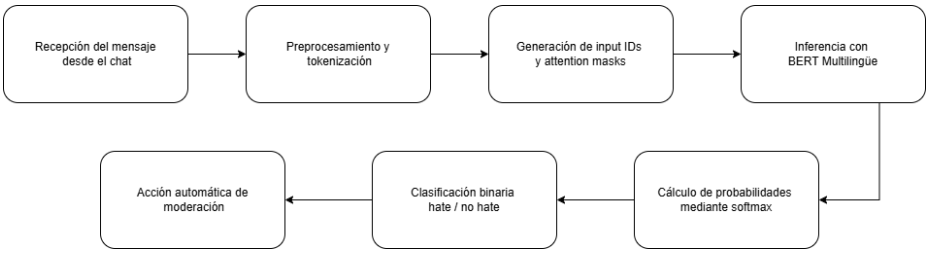

Fig. 3. Flujo operacional del sistema de moderación automática basado en BERT multilingüe.

Cuando un mensaje es clasificado como contenido dañino, el sistema ejecuta automáticamente acciones de moderación sobre el chat, incluyendo la eliminación del mensaje y la generación de advertencias dirigidas al usuario involucrado. En caso contrario, el mensaje permanece visible sin intervención adicional.

La arquitectura implementada permite desacoplar el módulo de inferencia del cliente de moderación, facilitando la escalabilidad del sistema y la posible integración con otras plataformas de comunicación en tiempo real.

Adicionalmente, durante el entrenamiento e inferencia se incorporaron técnicas de optimización computacional como *Automatic Mixed Precision* (AMP), procesamiento por lotes y pre-tokenización de datos, reduciendo el tiempo de ejecución y el consumo de memoria GPU.

4. Resultados y discusión

En esta sección se presentan los resultados experimentales obtenidos durante la evaluación del modelo propuesto para detección automática de discurso de odio en escenarios bilingües. La evaluación se realizó utilizando el conjunto de prueba estratificado, aplicando métricas estándar de clasificación implementadas mediante la biblioteca Scikit-learn [9].

Las métricas utilizadas fueron *accuracy*, *precision*, *recall* y *F1-score*. La métrica *accuracy* mide la proporción total de predicciones correctas, mientras que *precision* evalúa la proporción de predicciones positivas correctas respecto al total de predicciones positivas realizadas. Por otro lado, *recall* cuantifica la capacidad del modelo para identificar correctamente instancias positivas reales, y el *F1-score* representa la media armónica entre precisión y exhaustividad.

La Figura 4 presenta la matriz de confusión obtenida sobre el conjunto de prueba. Los resultados evidencian una adecuada capacidad discriminativa entre las clases *No Odio* y *Odio*, obteniendo altos valores en la diagonal principal y una cantidad reducida de errores de clasificación.

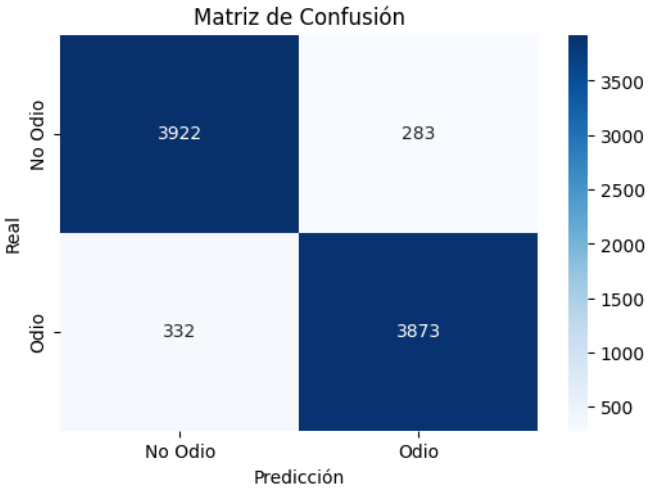

Fig. 4. Matriz de confusión obtenida sobre el conjunto de prueba para la clasificación binaria de discurso de odio.

A partir de la matriz de confusión se obtuvieron 3922 verdaderos negativos y 3873 verdaderos positivos, mientras que los falsos positivos y falsos negativos correspondieron a 283 y 332 instancias, respectivamente. Estos resultados indican que el modelo presenta un equilibrio adecuado entre sensibilidad y especificidad, minimizando errores críticos asociados a la moderación automática de contenido.

Con el objetivo de realizar una evaluación comparativa, el modelo basado en transformers fue contrastado frente a algoritmos tradicionales de clasificación de

texto ampliamente utilizados en tareas de procesamiento de lenguaje natural. La Tabla 4 muestra los resultados obtenidos.

Tabla 4. Comparación de modelos de clasificación

Modelo	**Accuracy**	**F1-score**
Naive Bayes	0.8514	0.8514
Logistic Regression	0.9063	0.9063
Random Forest	0.8923	0.8923
SVM	0.9128	0.9128
BERT	**0.9287**	**0.9287**

Los resultados muestran que el modelo *bert-base-multilingual-cased* obtuvo el mejor desempeño global, superando consistentemente a los modelos clásicos de aprendizaje automático. Este comportamiento se atribuye a la capacidad de los modelos transformer para capturar dependencias semánticas complejas y relaciones contextuales de largo alcance mediante mecanismos de autoatención multi-cabeza.

Asimismo, se realizaron múltiples ejecuciones independientes con diferentes inicializaciones aleatorias para analizar la estabilidad estadística del modelo. Los resultados se presentan en la Tabla 5.

Tabla 5. Resultados del modelo en múltiples ejecuciones

Ejecución	**Accuracy**	**Recall**	**F1-score**
1	0.93	0.93	0.93
2	0.92	0.92	0.92
3	0.92	0.92	0.92
4	0.93	0.93	0.93
5	0.93	0.93	0.93
Promedio	**0.93**	**0.93**	**0.93**

La baja variabilidad observada entre ejecuciones indica estabilidad durante el proceso de optimización y una adecuada capacidad de generalización sobre datos no vistos.

Adicionalmente, se analizó el comportamiento dinámico del entrenamiento mediante el monitoreo de las curvas de pérdida y exactitud. La Figura 5 muestra la evolución de las métricas durante el proceso de ajuste fino del modelo.

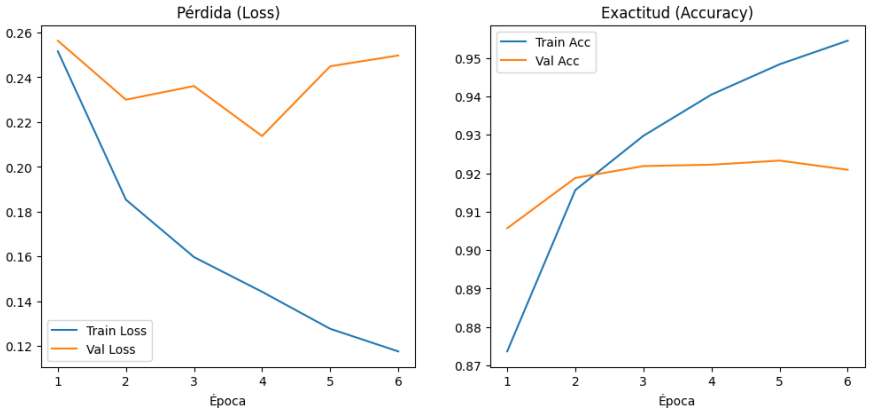

Fig. 5. Evolución de las métricas de entrenamiento y validación durante el proceso de fine-tuning del modelo BERT multilingüe.

La curva de pérdida de entrenamiento presenta una disminución progresiva a lo largo de las épocas, evidenciando convergencia durante la optimización. Por otro lado, la pérdida de validación muestra estabilidad relativa después de las primeras épocas, indicando una reducción controlada del error de generalización.

En términos de exactitud, el modelo alcanzó valores superiores al 95 % sobre entrenamiento y aproximadamente 92 % sobre validación, manteniendo una diferencia moderada entre ambas curvas. Este comportamiento sugiere una adecuada capacidad de aprendizaje sin presencia severa de sobreajuste.

La estabilización de las métricas de validación a partir de las últimas épocas justificó la utilización de mecanismos de regularización como *dropout*, *weight decay*, *gradient clipping* y *early stopping*, los cuales contribuyeron a mantener estabilidad numérica durante el entrenamiento.

4.1. Análisis de casos de falla

A pesar del desempeño obtenido por el modelo, se identificaron casos de error asociados principalmente a sarcasmo, ironía, lenguaje coloquial y expresiones semánticamente ambiguas. En algunos casos, mensajes ofensivos implícitos fueron clasificados como contenido no dañino debido a la ausencia de términos explícitamente tóxicos.

Asimismo, se observaron dificultades en mensajes con mezcla de idiomas (*code-switching*), abreviaciones y modificaciones intencionales de palabras ofensivas mediante símbolos o caracteres especiales, afectando el proceso de tokenización y representación semántica.

Otro factor relevante corresponde a la dependencia contextual de ciertos mensajes. Debido a que el modelo realiza clasificación a nivel de mensaje individual, no siempre es posible capturar referencias implícitas o información proveniente de conversaciones previas.

La Figura 4 muestra que los errores se concentran principalmente cerca del límite de decisión entre las clases *Odio* y *No Odio*, evidenciando la complejidad contextual inherente a la detección automática de discurso de odio.

Como trabajo futuro, se propone incorporar contexto conversacional y modelos especializados por idioma para reducir este tipo de errores.

5. Conclusiones

En este trabajo se presentó el diseño, implementación y evaluación de un sistema de moderación automática de contenido basado en técnicas de Procesamiento de Lenguaje Natural (PLN) y modelos transformer multilingües. El sistema fue desarrollado para la detección automática de discurso de odio en entornos de interacción en tiempo real, integrando un modelo basado en *bert-base-multilingual-cased* junto con mecanismos automáticos de moderación y respuesta.

El modelo propuesto fue entrenado mediante un proceso de *fine-tuning* completo utilizando un dataset bilingüe compuesto por textos en español e inglés provenientes de múltiples fuentes públicas. Los resultados experimentales demostraron que el modelo alcanza un desempeño competitivo, obteniendo un *accuracy* y *F1-score* promedio de 0.9287 sobre el conjunto de prueba, superando modelos tradicionales como Naive Bayes, Random Forest, Regresión Logística y SVM.

El análisis de la matriz de confusión mostró una adecuada capacidad discriminativa entre las clases *Odio* y *No Odio*, presentando una baja cantidad de falsos positivos y falsos negativos. Asimismo, las múltiples ejecuciones realizadas evidenciaron estabilidad estadística durante el entrenamiento, indicando una adecuada capacidad de generalización del modelo frente a variaciones en la inicialización aleatoria.

Desde el punto de vista de optimización, la incorporación de técnicas como *dropout*, *weight decay*, *gradient clipping*, *mixed precision* y *early stopping* permitió estabilizar el entrenamiento y reducir el riesgo de sobreajuste. Adicionalmente, el análisis de las curvas de entrenamiento mostró una convergencia progresiva y consistente de la función de pérdida y la exactitud durante el proceso de ajuste fino.

En términos de implementación, el sistema demostró capacidad de operación en tiempo real mediante la integración de FastAPI y TwitchIO, permitiendo procesar mensajes de chat, ejecutar inferencia automática y aplicar acciones de moderación con baja latencia. Esto confirma la viabilidad del sistema como una solución escalable para plataformas digitales y entornos de streaming.

Como principal contribución, este trabajo propone no únicamente un modelo de clasificación de discurso de odio, sino una arquitectura funcional de moderación automática capaz de integrarse en sistemas reales de comunicación en línea.

Como trabajo futuro, se plantea ampliar el tamaño y diversidad del dataset, incorporar evaluación prolongada en escenarios reales, realizar comparativas

frente a APIs comerciales de moderación y explorar arquitecturas transformer más recientes y especializadas para tareas multilingües de detección de contenido dañino.

Finalmente, los resultados obtenidos permiten concluir que los modelos transformer multilingües representan una alternativa efectiva para la moderación automática de contenido, ofreciendo un equilibrio adecuado entre precisión, robustez y capacidad de despliegue en tiempo real.

Referencias

1. Nabity-Grover, T., Cheung, C.M.K., Thatcher, J.B.: Inside out and outside in: How the COVID-19 pandemic affects self-disclosure on social media. International Journal of Information Management, 55 (2020)
2. Devlin, J., Chang, M.-W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In: Proceedings of NAACL-HLT (2019)
3. Jigsaw: Toxic Comment Classification Challenge. Kaggle (2018)
4. Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation, 9(8), 1735–1780 (1997)
5. Devlin, J., et al.: Multilingual BERT Model. Google Research (2019)
6. Schmidt, A., Wiegand, M.: A Survey on Hate Speech Detection using Natural Language Processing. In: Proceedings of SocialNLP (2017)
7. Paszke, A., et al.: PyTorch: An Imperative Style, High-Performance Deep Learning Library. NeurIPS (2019)
8. Wolf, T., et al.: Transformers: State-of-the-Art Natural Language Processing. EMNLP (2020)
9. Pedregosa, F., et al.: Scikit-learn: Machine Learning in Python. JMLR (2011)
10. Liu, Y., Ott, M., Goyal, N., et al.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692 (2019)
11. Davidson, T., Warmsley, D., Macy, M., Weber, I.: Automated Hate Speech Detection and the Problem of Offensive Language. Proceedings of ICWSM (2017)
12. Kolhatkar, V., Thain, N., Sorensen, J., Dixon, L., Taboada, M.: Classifying Constructive Comments. arXiv preprint arXiv:2004.09666 (2020)
13. Mathew, B., Saha, P., Yimam, S.M., et al.: HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection. Proceedings of AAAI (2021)
14. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention Is All You Need. Advances in Neural Information Processing Systems (NeurIPS) (2017)
15. Tonneau, M.: Spanish Hate Speech Superset. HuggingFace (2024)
16. Martínez-Cámara, E., et al.: Overview of TASS 2019: One More Further for the Global Spanish Sentiment Analysis. SEPLN (2019)
17. Basile, V., Bosco, C., et al.: SemEval-2019 Task 5: Multilingual Detection of Hate Speech. SemEval (2019)
18. Socher, R., et al.: Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. EMNLP (2013)