Avances en la Ingeniería del Lenguaje y del Conocimiento

Research in Computing Science

Series Editorial Board

Research in Computing Science es una publicación trimestral, de circulación internacional, editada por el Centro de Investigación en Computación del IPN, para dar a conocer los avances de investigación científica y desarrollo tecnológico de la comunidad científica internacional. Volumen 99, octubre de 2015. Tiraje: 500 ejemplares. *Certificado de Reserva de Derechos al Uso Exclusivo del Título* No. : 04-2005-121611550100-102, expedido por el Instituto Nacional de Derecho de Autor. *Certificado de Licitud de Título* No. 12897, *Certificado de licitud de Contenido* No. 10470, expedidos por la Comisión Calificadora de Publicaciones y Revistas Ilustradas. El contenido de los artículos es responsabilidad exclusiva de sus respectivos autores. Queda prohibida la reproducción total o parcial, por cualquier medio, sin el permiso expreso del editor, excepto para uso personal o de estudio haciendo cita explícita en la primera página de cada documento. Impreso en la Ciudad de México, en los Talleres Gráficos del IPN – Dirección de Publicaciones, Tres Guerras 27, Centro Histórico, México, D.F. Distribuida por el 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, México, D.F. Tel. 57 29 60 00, ext. 56571.

Editor responsable: *Grigori Sidorov, RFC SIGR651028L69*

Research in Computing Science is published by the Center for Computing Research of IPN. Volume 99, October 2015. Printing 500. The authors are responsible for the contents of their articles.

Printed in Mexico City, in the IPN Graphic Workshop – Publication Office.

Avances en la Ingeniería del Lenguaje y del Conocimiento

David Pinto
Darnes Vilariño (eds.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2015

ISSN: 1870-4069

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 and Periodica / Indexada en LATINDEX y Periódica

Printing: 500 / Tiraje: 500

Printed in Mexico / Impreso en México

Editorial

En el presente volumen se publican una serie de artículos seleccionados, los cuales fueron previamente presentados en el simposio en Ingeniería del Lenguaje y del Conocimiento (LKE‘2015), en particular en la tercera edición de esta serie de eventos. Esta conferencia ha sido organizada en el seno de la Facultad de Ciencias de la Computación de la Benemérita Universidad Autónoma de Puebla (BUAP) por tres años consecutivos. Nace como una iniciativa del laboratorio de Ingeniería del Lenguaje y del Conocimiento con la finalidad de ofrecer un espacio académico y de investigación, en el cual sea posible reportar trabajos relacionados con el área. Este evento promueve la cooperación entre diferentes grupos de investigación, pues permite el intercambio de resultados científicos, prácticos y la generación de nuevo conocimiento.

Esta edición especial de la Revista Research in Computing Science contiene entonces una serie de contribuciones originales que han sido seleccionadas a partir de un proceso de evaluación ciega doble (double blind), lo cual significa que los nombres de los autores de los artículos y los nombres de los revisores son ambos desconocidos. Este procedimiento es ejecutado en aras de proveer una evaluación anónima, que derive en artículos de mayor calidad para este volumen; particularmente, en esta ocasión la tasa de rechazo fue del 22%, cuidando que en todos los casos, al menos dos especialistas del comité revisor hicieran una evaluación de la pertinencia, originalidad y calidad de cada artículo sometido.

Esperamos que este volumen sea de utilidad para el lector y los autores de los artículos seleccionados encuentren en esta edición especial un espacio de intercambio científico productivo que enriquezca la colaboración entre estudiantes y académicos en el ámbito de la ingeniería del lenguaje y del conocimiento.

El proceso de revisión y selección de artículos se llevó a cabo usando el sistema libremente disponible llamado EasyChair, http://www.easychair.org.

David Eduardo Pinto Avendaño
Darnes Vilariño Ayala

Octubre 2015

Table of Contents

Page

Towards a Reasoning Model for Context-aware Systems: Modal Logic and the Tree Model Property

Yensen Limón[1], Everardo Bárcenas[1,2], Edgard Benítez-Guerrero[1], Carmen Mezura-Godoy[1]

[1] Universidad Veracruzana, Facultad de Estadística e Informática, Mexico

[2] Consejo Nacional de Ciencia y Tecnología, Mexico

Abstract. Modal logics forms a family of formalisms widely used as reasoning frameworks in diverse areas of computer science. Description logics and their application to the web semantic is a notable example. Also, description logics have been recently used as a reasoning model for context-aware systems. Most reasoning algorithms for modal (description) logics are based on tableau constructions. In this work, we propose a reasoning (satisfiability) algorithm for the multi-modal K_m with converse. The algorithm is based on the finite tree model property and a Fischer-Ladner construction. We show the algorithm is sound and complete, and we provide the corresponding complexity analysis. We also present some exploratory results of a preliminary implementation of the algorithm.

Keywords: Modal logic, context-aware systems, reasoning.

1 Introduction

Modal logics forms a family of formalism including temporal, dynamic, epistemic and description logics. These formalisms have been widely used as reasoning frameworks in diverse areas of computer science, such as artificial intelligence, databases, program analysis, distributed computing, etc. [17, 15, 10]. The application of description logics as knowledge representation framework (language and reasoning for ontologies) in the semantic web is a notable example [3]. In recent years, due to the well-known excellent balance between the expressive power of description logic and the computational complexity of the associated algorithms, context aware computing inference systems have been studied in the modal (descriptive) setting [8, 6]. However, current context aware systems, which are supposed to efficiently interact with a legion of context variables, still demands more expressive power without performance detriment [8]. Motivated by the development of expressive context aware inference systems, as an starting point, we propose in the current work a reasoning algorithm for the multimodal logic

K_m with converse, which can be seen as a syntactic variant of the description logic $\mathcal{ALCI}$.

The basic modal logic K can be syntactically introduced as classical propositional logic extended with constructors for expressing modalities, such as possibility and necessity. From the seminal work of van Benthem [7], we know modal logic forms an important fragment of first order logic. The importance of this fragment comes from the associated well known and behaved computational properties, such as model checking and satisfiability. A possible explanation of this nice and robust computational behavior, provided by van Benthem [7], is that modal logic cannot distinguish bisimilar models. Another explanation comes from Vardi [20], and it concerns the tree model property of the logic. In the current work, we follow the second approach, more precisely, we propose a satisfiability algorithm for modal logic based on its finite tree model property. The algorithms actually builds candidate trees in the style of Fischer-Ladner [4, 5]. Moreover, the algorithm works for the modal logic extended with multimodalities K_m, including converse modalities, known in description and dynamic logics as inverse roles and programs, respectively.

We distinguish two types of techniques in the development of modal reasoners: purpose-built and translational. Most of purpose-built translational tools are based on tableau constructions [13, 12, 19, 11], however, there are other few methods, such as sequent calculus [21] and coalgebras [9]. In the translational methods, we may find translations to SAT solvers [18], to first order logic [14, 2], including SMT and QBF solvers [1, 16], respectively. In [16], it is also presented a translation of modal logic formulas to types, which can be seen as an on-the-fly construction of the corresponding automaton. For the current work, we focus on the purpose-built approach. In particular, the proposed algorithm is based in the finite tree model property of modal logic. More precisely, the algorithm performs a Fischer-Ladner construction of candidate trees. Although there are already highly optimized modal solvers successfully working in practice [13, 12], to the best of our knowledge, this is the first reasoning algorithm based on the finite tree model property.

The paper is organized as follows: in Section 2, we describe the multimodal logic K_m with converse; the finite tree model property of the logic is studied in Section 3; in Section 4, a satisfiability algorithm is described in terms of Fischer-Ladner constructions of trees, it is also shown the algorithm is correct, that is, sound and complete, and a complexity analysis is also provided; we conclude in Section 5 with a summary of the current work and a discussion on further research perspectives.

2 Preliminaries

In the current section, we define the propositional multimodal logic K_m with converse. Whereas syntax of the logic is defined as boolean combination of propositions and modal constructors, formula semantics is described in terms of Kripke (relational) structures.

2.1 Multimodal logic K_m with converse

We assume a fixed alphabet (P, M), where P is a countable set of proposition variables and M is a finite set of modalities. We also assume there is a bijection partitioning the set of modalities. We write $\overline{m}$ to denote the inverse of a modality m.

Definition 1 (Syntax)**.** The set of modal formulas is defined by the following grammar:

$$\phi := p \mid \neg\phi \mid \phi \vee \phi \mid \langle m \rangle \phi$$

where p ranges over propositions and m over modalities

Formulas are interpreted as subset nodes on Kripke structures, which can be intuitively seen as labeled directed graphs. Propositions serve as node labels, whereas negation and disjunction are interpreted as the complement and union of sets, respectively. Existential modal formulas $\langle m \rangle \phi$ holds in nodes able to access through m to a node belonging to the interpretation of ϕ.

Other classical syntactic sugar is also considered: tautologies $\top := \phi \vee \neg\phi$; contradictions $\bot := \neg\top$; conjunctions $\phi \wedge \psi := \neg(\neg\phi \vee \neg\psi)$; and universal modal formulas $[m]\,\phi := \neg\langle m \rangle \neg\phi$. These shorthands are interpreted as expected, in particular, universal modal formulas $[\phi]$ denote the nodes where *all* their accessible nodes through m support ϕ.

Before giving a formal semantics of modal formulae, we give a precise notion of Kripke structures.

Definition 2 (Kripke structure)**.** A Kripke structure is a tuple $K = (N, R, L)$, where: N is a non-empty and countable set of nodes; R is a family of binary relations $R^m : N \times N$, written $n \in R(n, m)$ for each modality m; and L is a left-total labeling function $L : N \mapsto 2^P$.

We now give a precise notion of formula semantics.

Definition 3 (Semantics)**.** Given a Kripke structure $K = (N, R, L)$, modal formulas are interpreted as follows:

$$\begin{aligned}
[\![p]\!]^K &= \{n \mid p \in L(n)\} \\
[\![\neg\phi]\!]^K &= N \setminus [\![\phi]\!]^K \\
[\![\phi \vee \psi]\!]^K &= [\![\phi]\!]^K \cup [\![\psi]\!]^K \\
[\![\langle m \rangle \phi]\!]^K &= \left\{ n \mid R(n, m) \cap [\![\phi]\!]^K \neq \emptyset \right\}
\end{aligned}$$

We say a formula ϕ is satisfiable, if and only if, there is a Kripke structure K such that the interpretation of ϕ with respect to K is not empty, that is, $[\![\phi]\!]^K \neq \emptyset$. In such a case, we also say K is a model of ϕ. If any Kripke structure is a model of ϕ, we say ϕ is valid. We say two formulas ϕ and ψ are equivalent, if and only if, their interpretations coincide for any Kripke structure K, that is, $[\![\phi]\!]^K = [\![\psi]\!]^K$.

The negation normal form of a formula is an equivalent formula where negation only occurs in front of propositions. In order to give a precise notion of the negation normal form, we then consider the following definition.

Definition 4. Given a modal formula ϕ, its negation in normal form $\mathrm{nnf}(\phi)$ is inductively defined as follows:

$$\begin{aligned} \mathrm{nnf}(p) =& \neg p \\ \mathrm{nnf}(\phi \vee \psi) =& \mathrm{nnf}(\phi) \wedge \mathrm{nnf}(\psi) \\ \mathrm{nnf}\left(\langle m \rangle \phi\right) =& [m]\, \mathrm{nnf}(\phi) \end{aligned}$$

The negation normal form of a formula ϕ is defined by $\phi\left[{}^{\mathrm{nnf}(\psi)}/_{\neg\psi}\right]$. Then, in order to consider formulas in negation normal form only, we need to consider an extension of the logic with conjunctions and universal modal formulas with the expected semantics.

Proposition 1. *A formula and its negation normal form are equivalent.*

Hence, with out loss of generality, we consider now formulas in negation normal form.

3 The Tree Model Property

In this Section, we provide a description of the finite tree model property: if a formula is satisfiable, then it is also satisfiable in a finite tree shaped Kripke structure.

Definition 5. Given a formula ϕ and a Kripke structure $K = (N, R, L)$ satisfying ϕ in a node $n \in N$, we inductively define the following finite tree shaped Kripke structure $\sigma\left(\phi, K, n\right) = \left(\sigma\left(N\right), \sigma\left(R\right), \sigma L\right)$ as follows:

- the root is $\sigma(n) \in \sigma\left(N\right)$, in case $\sigma(n)$ has already been added in a previous step, then a fresh copy of $\sigma(n)$ is considered;
- in case ϕ is a proposition p, then $p \in \sigma\left(L(\sigma(n))\right)$;
- if ϕ is a negation $\neg p$, $p \notin \sigma\left(L(\sigma(n))\right)$;
- when ϕ is a disjunction $\psi \vee \varphi$, then we know that n satisfies ψ or φ, in case n satisfies ψ, then $\sigma\left(\phi, K, n\right)$ is defined by $\sigma\left(\psi, K, n\right)$, otherwise, it is defined by $\sigma\left(\varphi, K, n\right)$;
- if it is the case that ϕ is a conjunction $\psi \wedge \varphi$, then $\sigma\left(\phi, K, n\right)$ is defined by the two branched tree formed by $\sigma\left(\psi, K, n\right)$ and $\sigma\left(\varphi, K, n\right)$ with $\sigma(n)$ as the common root.
- if ϕ is an existential modal formula $\langle m \rangle \phi$, then $\sigma\left(\phi, K, n\right)$ is formed by $\sigma\left(\psi, K, n\right)$, such that for a node $n' \in R(n, m) \cap [\![\psi]\!]^K$, $\sigma\left(n'\right) \in \sigma(R(\sigma, m))$ and $\sigma\left(n'\right)$ is the root of $\sigma\left(\psi, K, n\right)$;
- the case of universal modal formulas $[m]\, \psi$ is similar as the previous one, but the construction is done with respect to each node $n' \in R(n, m) \cap [\![\psi]\!]^K$.

Theorem 1 (Finite tree model property). *For any formula ϕ and Kripke structure K, we have that*

$$n \in [\![\phi]\!]^K \text{ if and only if } \sigma(n) \in [\![\phi]\!]^{\sigma(\phi,K,n)}$$

By Definition 5, the proof goes smoothly by structural induction on ϕ. Also, it is clear that $\sigma(\phi, K, n)$ is finite and tree shaped.

4 Satisfiability

In this Section, we describe a satisfiability algorithm for the multi-modal K_m with converse, that is, given a formula, the algorithm decides whether or not the input formula is satisfiable. Recall that if a formula is satisfiable, then the formula is satisfiable by a tree shaped Kripke structure. Then, the algorithm builds candidate trees in a bottom-up manner: starting from the leaves, the algorithm adds parents iteratively and consistently until a satisfying tree is found. The representation of trees is in the style of Fischer-Ladner [4, 5]. Before defining the algorithm, we first need some notation.

Definition 6. . We define the following binary relation R^{FL} on formulas with $i = 1, 2$ as follows:

$$R^{FL}(\phi_1 \circ \phi_2, \phi) \qquad R^{FL}(\langle m \rangle \phi, \phi) \qquad R^{FL}([m]\phi, \phi)$$

where $\circ \in \{\wedge, \vee\}$ and $i = 1, 2$.

We now define the Fischer-Ladner closure.

Definition 7 (Fischer Ladner Closure)**.** Given a formula ϕ, the Fischer-Ladner closure of ϕ, written $FL(\varphi)$, is defined as $F_k(\phi)$ for the smallest k, such that $F_k(\phi) = F_{k+1}(\phi)$, where

$$\begin{aligned} FL_0(\varphi) =& \{\varphi\} \\ FL_{i+1}(\varphi) =& FL_i(\varphi) \cup \{\psi' \mid R^{FL}(\psi, \psi'), \psi \in FL_i(\varphi)\} \end{aligned}$$

for $i > 0$.

The Fischer-Ladner representation of leaves, and hence trees, is based in the lean set, which is now defined.

Definition 8 (Lean)**.** Given a formula ϕ, its lean is defined as the set composed by proposition and modal subformulas of ϕ, together with formulas $\langle m \rangle \top$, for every m occurring in ϕ, and p' which is a proposition not occurring in ϕ. More precisely,

$$lean(\varphi) = \{p, \langle m \rangle \psi, [m]\psi \in FL(\varphi)\} \cup \{\langle m \rangle \top, p'\}$$

We are now ready to define the set of nodes in Fischer-Ladner trees.

Algorithm 1 Satisfiability algorithm.

$Y \leftarrow N^{\phi}$
$X \leftarrow Leaves(Y)$
$X_0 \leftarrow \emptyset$
while $X \neq X_0$ **do**
  if $X \vdash \phi$ **then**
    return true
  end if
  $X_0 \leftarrow X$
  $(X, Y) \leftarrow Update(X, Y)$
end while
return false

Definition 9 (Nodes)**.** Given a formula ϕ, a ϕ-node, or simply a node, is defined as a lean subset with the following constraints:

- at least one proposition occurs in it; and
- if $\langle m \rangle \psi$ occurs, then $\langle m \rangle \top$ also does.

The set of nodes corresponding to a given formula ϕ is written N^{ϕ}.

Definition 10 (Fischer-Ladner tree)**.** Given a formula ϕ, a ϕ-tree T, or simply a tree, is inductively defined as follows:

- the empty set is a tree;
- the tuple $(n, T_1, \ldots, T_n)$ is a tree, provided that n is a node, called the root, and T_i $(i = 1, \ldots, n)$ are trees.

The algorithm corresponding to the satisfiability of formulas is defined in Algorithm 1.

We now give a precise description of each notion involved in the algorithm.

We first define the set of leaves as the trees $(n, \emptyset)$, where n does not contain formulas of the form $\langle m \rangle \phi$. Then, $Leaves(N)$ for a set of nodes N, contains all the leaves in N.

We now define the entailment relation between nodes and formulas.

Definition 11. Given a formula ϕ and a node n, we inductively define the entailment relation $n \vdash$ as follows:

$$\frac{\phi \in n}{n \vdash \phi} \qquad \frac{p \notin n}{n \vdash \neg p} \qquad \frac{}{n \vdash \top}$$

$$\frac{n \vdash \phi \text{ and } n \vdash \psi}{n \vdash \phi \wedge \psi} \qquad \frac{n \vdash \phi \text{ or } n \vdash \psi}{n \vdash \phi \vee \psi}.$$

Abusing of notation, we extend the notion of entailment between trees $T = (n, T_1, \ldots, T_n)$ and formulas ϕ, written $T \vdash \phi$, when $n \vdash \phi$. This notion is also extended to set of trees X, written $X \vdash \phi$, when there is a tree $T \in X$, such that $T \vdash \phi$. The complement of this relation $\nvdash$ is defined as expected.

The $Update(X, Y)$ function, for a set of trees X and a set of nodes Y, adds parents in Y to trees in X in a consistent manner. This results in a pair (X', Y'), where X' is the new set of trees and Y' is the new set of nodes (the nodes used as parents are removed).

Definition 12. Given a formula ϕ, a set of trees X and a set of nodes Y, we define the $Update$ function as follows

$$Update(X, Y) = (X', Y')$$

where

$$X' = \{(n, T_1, \ldots, T_n) \mid n \in Y, \Delta_m(n, T_i)\}$$

for each $i = 1, \ldots, n$ and m occurring in ϕ, and where

$$\Delta_m(n, T) = \begin{cases} \textbf{true} & \text{if } \forall \langle m \rangle \psi \in lean(\phi) : \langle m \rangle \psi \in n \text{ iff } n' \vdash \psi \\ & \forall [m] \psi \in lean : [m] \phi, \langle m \rangle \top \in n \text{ iff } n' \vdash \psi \\ & [m] \psi \in n, \langle m \rangle \top \notin n \text{ iff } n' \nvdash \psi \\ \textbf{false} & \text{otw.} \end{cases}$$

where n' is the root of T, and

$$Y' = Y \setminus \{n \mid \text{ n is the root of tree in } X'\}$$

Notice that $Update$ is a monotone function, hence, it has a fixed-point. We now show the algorithm is correct. This is shown through soundness and completeness.

Theorem 2 (Soundness). *Given a formula ϕ, if the algorithm returns true, then ϕ is satisfiable.*

Proof. If the algorithm returns true, we know there is a tree $T = (n, T_1, \ldots, T_n)$ entailing ϕ, that is, $T \vdash \phi$. We now construct a tree shaped Kripke structure $K = (N, R, L)$ from T, satisfying ϕ.

- N is composed by each node in T;
- for each subtree $T' = (n', T'_1, \ldots, T'_k)$ in T and for each $i = 1, \ldots, k$, $n'' \in R(n', m)$, such that n'' is the root of T'_i and $\Delta_m(n', n'')$; and
- for each node n' in N and each proposition $p \in lean(\phi)$, if $p \in n'$, then $p \in L(n')$.

That K satisfies ϕ is proven by a straightforward structural induction, due to the soundness of relations Δ and $\vdash$. □

Theorem 3 (Completeness). *If a satisfiable formula ϕ is given to the algorithm, then the algorithm returns true.*

Proof. By Theorem 1, we know there is a tree shaped Kripke structure $K = (N, R, L)$ satisfying ϕ. Moreover, K is defined as described in Definition 5.

We first show a tree T isomorphic to K entails ϕ. For this, we first define T as follows:

- for each each node n in N, there is a corresponding node $\tau(n)$ in T;
- for each $n_1, n_2, \ldots, n_k \in R(n, m)$ for any m occurring in ϕ, $(\tau(n), T_1, T_2, \ldots, T_k)$ is a subtree in T, such that $\tau(n_i)$ is the root of T_i for $i = 1, \ldots, k$;
- for each $p \in L(n)$, $p \in \tau(n)$;
- for each $\langle m \rangle \psi, \in lean(\phi)$, if $n \in [\![\langle m \rangle \psi]\!]^K$, then $\langle m \rangle \psi \in \tau(n)$, the same applies for formulas $[m]\psi$ in the lean; and finally,
- for each $\tau(n)$ in T, if $\langle m \rangle \psi \in \tau(n)$, then $\langle m \rangle \top \in \tau(n)$.

By a straightforward structural induction on ϕ, it is shown that $T \vdash \phi$ (recall ϕ is satisfied in the root of K).

We now show T is constructed by the algorithm. This is shown by induction on the height of K. It is clear T has the same height of K. The base case is immediate. We now assume K has has height n, then the algorithm has constructed trees $T_1, T_2, \ldots, T_k$ corresponding to the subtrees of K. Now notice that if $n > 1$, then ϕ is of the form $\langle m \rangle \psi$ or $[m]\psi$. Hence, considering n is the root of K and that it is the one satisfying ϕ, then node $\tau(n)$ is still in Y. By completeness of relation *Delta*, we know then for each m occurring in ϕ and each $i = 1, \ldots, k$, $\Delta_m(n, n_i)$, provided that n_i is the root of T_i. We then conclude $\tau(n) \vdash \phi$. □

Theorem 4 (Complexity). *The satisfiability algorithm is in EXPTIME.*

Proof. First notice that the lean set has linear size with respect to the input formula ϕ. Hence, there is an exponential number of nodes. Each node is at most of the same size than ϕ. Hence, for each node n, $n \vdash \phi$ takes linear time. When it comes to trees, $\vdash$ is clearly at most exponential. Testing the set of trees also takes at most exponential: it takes the sum of exponentials searches. Finally, the exponential bound on the *Update* function comes from the exponentially bounded size of its search space (nodes and trees), and from the fact that Δ has linear cost. □

We are currently implementing a preliminary version of the algorithm in Java language. Some exploratory results of this preliminary version are depicted in Figure 1. We tested the implementation in a computer with the following features: Windows 8 operating system, AMD processor A6 2.7GHz., 8Gb of RAM. In this preliminary version, we have implemented the set of nodes in an explicit way, which results very expensive in practice. In Figure 1, it is easy to notice that incrementing a single modal level in the input formula, which implies a higher tree, drastically impact in the performance of the algorithm. In order to alleviate this issue, we plan to soon incorporate to the algorithm a non-explicit representation of nodes, such as Binary Decision Diagrams [16].

5 Conclusions

In this paper, we introduced a satisfiability algorithm of the multi-modal logic K_m with converse. The algorithm is based on the finite tree model property

Formula	Time (milliseconds)
$\langle 1\rangle\, a$	2153
$\langle 1\rangle\, a \wedge b$	5180
$\langle 1\rangle\, a \wedge \langle 2\rangle\, b$	11419
$\langle 1\rangle\, a \wedge \langle \overline{1}\rangle\, b$	9188
$\langle 1\rangle\, a \wedge \langle \overline{2}\rangle\, b$	9110
$\langle 1\rangle\, a \wedge \langle 2\rangle\, b \wedge c$	42604
$\langle 1\rangle\, a \wedge \langle 2\rangle\, b \wedge \neg c$	41403
$\langle 1\rangle\, a \wedge \langle 2\rangle\, b \wedge \neg(c \vee d)$	38937
$\langle 1\rangle\, \langle 1\rangle\, a$	157359
$\langle 1\rangle\, \langle 1\rangle\, a \wedge b$	318714
$\langle 1\rangle\, \langle 1\rangle\, a \wedge \langle 1\rangle\, b$	863249
$\langle 1\rangle\, \langle 1\rangle\, a \wedge \langle \overline{1}\rangle\, b$	595539
$\langle 1\rangle\, \langle 1\rangle\, a \wedge \langle 2\rangle\, b$	897349
$\langle 1\rangle\, \langle 1\rangle\, a \wedge \langle \overline{2}\rangle\, b$	904157
$\langle 1\rangle\, \langle 1\rangle\, a \wedge \langle 2\rangle\, \langle 2\rangle\, b$	2043845

Fig. 1. Results of a preliminary implementation of the satisfiability algorithm.

of the logic. We also showed the algorithm is sound and complete, and that it takes exponential time. Some exploratory results of a naive and non-optimized implementation of the algorithm were also described.

We are currently implementing non-explicit representations of the set of nodes. In particular, we are implementing a BDD-based version the algorithm [16]. We plan to extend the current algorithm, as described in [4, 5], to more expressive logics, such as the μ-calculus with arithmetic constraints. We are also studying the Description Logics counterpart of these expressive logics. This is with the final aim to provide an efficient and expressive reasoning framework for context-aware systems [6, 8].

Acknowledgment. This work was partially developed under the support of the Mexican National Science Council (CONACYT) in the scope of the Cátedras CONACYT project Infraestructura para Agilizar el Desarrollo de Sistemas Centrados en el Usuario (Ref 3053).

References

1. Areces, C., Fontaine, P., Merz, S.: Modal satisfiability via SMT solving. In: Nicola, R.D., Hennicker, R. (eds.) Software, Services, and Systems - Essays Dedicated to Martin Wirsing on the Occasion of His Retirement from the Chair of Programming and Software Engineering. Lecture Notes in Computer Science, vol. 8950, pp. 30–45. Springer (2015)
2. Areces, C., Gennari, R., Heguiabehere, J., de Rijke, M.: Tree-based heuristics in modal theorem proving. In: ECAI 2000, Proceedings of the 14th European Conference on Artificial Intelligence. pp. 199–203 (2000)
3. Baader, F., Hollunder, B.: KRIS: knowledge representation and inference system. SIGART Bulletin 2(3), 8–14 (1991)

4. Bárcenas, E., Genevès, P., Layaïda, N., Schmitt, A.: Query reasoning on trees with types, interleaving, and counting. In: Walsh, T. (ed.) IJCAI 2011, Proceedings of the 22nd International Joint Conference on Artificial Intelligence. pp. 718–723. IJCAI/AAAI (2011)
5. Bárcenas, E., Lavalle, J.: Global numerical constraints on trees. Logical Methods in Computer Science 10(2) (2014)
6. Benítez-Guerrero, E.: Context-aware mobile information systems: Data management issues and opportunities. In: Arabnia, H.R., Hashemi, R.R., Vert, G., Chennamaneni, A., Solo, A.M.G. (eds.) Proceedings of the 2010 International Conference on Information & Knowledge Engineering. pp. 127–133. CSREA Press (2010)
7. van Benthem, J.: Modal Logic and Classical Logic. Bibliopolis (1983)
8. Bettini, C., Brdiczka, O., Henricksen, K., Indulska, J., Nicklas, D., Ranganathan, A., Riboni, D.: A survey of context modelling and reasoning techniques. Pervasive and Mobile Computing 6(2), 161–180 (2010)
9. Calin, G., Myers, R.S.R., Pattinson, D., Schröder, L.: Coloss: The coalgebraic logic satisfiability solver. Electr. Notes Theor. Comput. Sci. 231, 41–54 (2009)
10. Fitting, M.: Modality and databases. In: Dyckhoff, R. (ed.) Automated Reasoning with Analytic Tableaux and Related Methods, International Conference. Lecture Notes in Computer Science, vol. 1847, pp. 19–39. Springer (2000)
11. Genevès, P., Layaïda, N., Schmitt, A., Gesbert, N.: Efficiently deciding μ-calculus with converse over finite trees. ACM Trans. Comput. Log. 16(2), 16 (2015)
12. Haarslev, V., Möller, R.: RACER system description. In: Goré, R., Leitsch, A., Nipkow, T. (eds.) Automated Reasoning, First International Joint Conference. Lecture Notes in Computer Science, vol. 2083, pp. 701–706. Springer (2001)
13. Horrocks, I., Patel-Schneider, P.F.: Optimizing description logic subsumption. J. Log. Comput. 9(3), 267–293 (1999)
14. Hustadt, U., Schmidt, R.A., Weidenbach, C.: MSPASS: subsumption testing with SPASS. In: Lambrix, P., Borgida, A., Lenzerini, M., Möller, R., Patel-Schneider, P.F. (eds.) Proceedings of the 1999 International Workshop on Description Logics. CEUR Workshop Proceedings, vol. 22. CEUR-WS.org (1999)
15. Kraus, S., Lehmann, D.J.: Knowledge, belief and time. Theor. Comput. Sci. 58, 155–174 (1988)
16. Pan, G., Sattler, U., Vardi, M.Y.: Bdd-based decision procedures for the modal logic K. Journal of Applied Non-Classical Logics 16(1-2), 169–208 (2006)
17. Pnueli, A.: The temporal logic of programs. In: 18th Annual Symposium on Foundations of Computer Science. pp. 46–57. IEEE Computer Society (1977)
18. Sebastiani, R., Vescovi, M.: Automated reasoning in modal and description logics via SAT encoding: the case study of k(m)/alc-satisfiability. J. Artif. Intell. Res. (JAIR) 35, 343–389 (2009)
19. Tanabe, Y., Takahashi, K., Hagiya, M.: A decision procedure for alternation-free modal μ-calculi. In: Areces, C., Goldblatt, R. (eds.) Advances in Modal Logic 7. pp. 341–362. College Publications (2008)
20. Vardi, M.Y.: Why is modal logic so robustly decidable? In: Immerman, N., Kolaitis, P.G. (eds.) Descriptive Complexity and Finite Models, Proceedings of a DIMACS Workshop. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 31, pp. 149–184. American Mathematical Society (1996)
21. Voronkov, A.: How to optimize proof-search in modal logics: new methods of proving redundancy criteria for sequent calculi. ACM Trans. Comput. Log. 2(2), 182–215 (2001)

Discovering Semantic Relationships between NCD and Lifestyle Patterns using Ontologies

María J. Somodevilla, Ismael Mena, Ivo H. Pineda, Concepcion Perez de Celis

Computer Science Department, Puebla,
Mexico

{mariajsomodevilla, imenav85, ivopinedatorres, mcpcelish,
dvilarinoayala}@gmail.com

Abstract. The volume of biomedical spatial information available on line increases day by day, which need to be exploited and shared by users in different knowledge areas. NCD[1] kill 38 million people each year and considering the seriousness of the problem globally, there are already underway government strategies to reduce risk factors and early detection and timely treatment. In this article, it's discussed the problem of the NCD using Semantic Web tools. The proposed system uses six ontologies which formalize concepts related to people, physical activity, the NCD, nutrition, geographic regions and symptoms to give information about lifestyle patterns. SWRL[2] rules are used to define accurate axioms which allow improving classification of the individuals.

Keywords: Ontologies system, SWRL rules, lifestyle patterns, NCD.

1 Introduction

NCDs kill 38 million people each year and are classified as cardiovascular, cancer, respiratory and diabetes. Affect all age groups and all geographic regions. The risk factors associated with NCDs are unhealthy stuff consumption, physical inactivity and non-balanced diets. These habits lead to metabolically and/or physiologically change that increase the risk of suffering NCD: hypertension, overweight/obesity, hyperglycemia and hyperlipidemia. The higher costs in the long term treatment of NCDs determine a region with high rates of extreme poverty such as in a developing country. WHO[3] has been developing a various types of strategies in order to reduce risks factors that include early detection and on-time treatment of diseases.

One of the characteristics of OWL-DL is its OWA[4] and how it affects to axioms defined in any ontology. As a result, those axioms must be clearly defined in order to avoid inconsistencies on reasoning time. These axioms involve several classes, then

[1] Non-Communicable Diseases
[2] Semantic Web Rule Language
[3] World Health Organization
[4] Open World Assumption

the reasoner will properly deduct their child classes and thus produce more accurate results.

In Section 2, the related work about spatial and biomedical ontologies is presented. The ontology construction methodology is discussed in Section3. In Section 4, the SWRL Rules are properly presented considering their application in the lifestyle patterns deduction. Finally, in Section 5 the conclusions and future work are presented.

2 Previous Work

Ontology integration can be achieved in three main ways: by merging ontologies, by mapping local ontologies to a global ontology, and by integrating local ontologies by means of semantic bridges that define mappings between the ontologies. Ontology merging is suitable for use in traditional systems which are small or moderate in size and are fairly static, and where scalability is not a core requirement. In ontology mapping, specific ontologies can be derived from global or 'reference' ontology. Ontology mapping in this case becomes much easier since concepts in different ontologies that need to be mapped are derived from the same ontology.

Our currently work on ontology integration is based on a new approach of interaction between ontologies [1, 2] which is called ontology system, where a set of ontological modules with semantic relationships among them are defined. This process allows specifying domain ontologies separately and then integrating them into a new ontology, where rules are defined to generate new knowledge. It is necessary to note that based from another work introduced by Rodriguez, J., Romero, M., & Bravo M., the methodology design is carried out successfully incorporating the elicitation term i.e., the competence questions and the division in phases of the methodology.

The last methodology allows for a seamless integration among distinct individual ontologies as is shown in [3] and [4]. This has been a first approach to achieve health & spatial ontology integration.

There exist works in the mapping health ontologies area, where the systems like ICD-10[5] or RxNorm[6] are discussed. Both ICD-10 and RxNorm are vocabularies of diseases, symptoms and findings. The first is an WHO's effort and the second one is a work from the USA´s Health organization. Another relevant approach, Ontology Integration Systems (OIS), involves descriptive logic, as a deduction mechanism. Therefore, in health ontology mapping approach exists 3 different categories: global view to local view, semantic mapping between targets and entities and mapping to enable ontology re-use [5].

In spatial ontologies field, different works have been reported using various approaches, involving the development of new spatial relationships, like spatial relationships in 3D [6]. However, it is difficult to properly define the geographic ontologies in terms of semantic, geometric and topological relationships.

A process to extend an existent geographic ontology with data mining is reviewed in [7]. Some extensions to be highlighted is concern of the semantic bridge axioms

[5] International Statistical Classification of Diseases and Related Problems 10th Revision
[6] Bodenreider, O.: Unified Medical Language system (UMLS)

layer which interacts between two ontologies, and also perform spatial reasoning by *PelletSpatial*.

Additionally, in [8], is presented another approach to join up spatial and biomedical ontologies through a bridge layer axiom which provides the necessary axioms to work between a spatial and a biomedical ontology. This work intends to reunite all the reviewed work by adding another useful approach by reusing ontologies and, implementing SWRL Rules.

3 Ontology System Development

As presented, in Section 2, the general methodology is in Bravo, M. (2014), and it is consists in the system design in 3 stages which are drawn in Fig. 1.

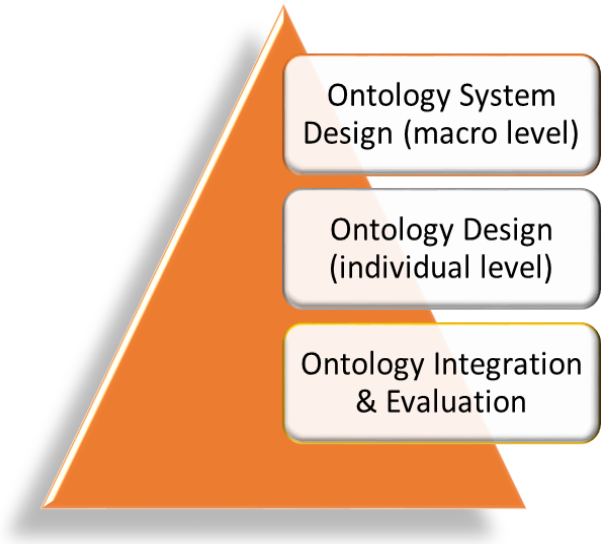


Fig. 1. Construction Methodology of a General System Ontologies.

3.1 Ontology System Design

The system integrates 6 different ontologies: *Person, NCD, Physical Activity, Nutrition, OntoMex* and *Symptom*. The integration is carried out by establishing relationships among instances of classes in different ontologies, in Fig. 2, the system's conceptual and relationship design is presented. In the case of the Region, Entity and Locality concepts, only a part of all *OntoMex* was extracted, in order to use reduced resources. To achieve the semantic relationship' conceptual design, the following competence questions were identified:

- What are the principal risk factors to suffer diabetes?
- Is the risk of suffer diabetes related with the nutrition high in fat?
- How is a mid-level lifestyle pattern defined?
- What are the characteristics of a person with nutrition high in fat?

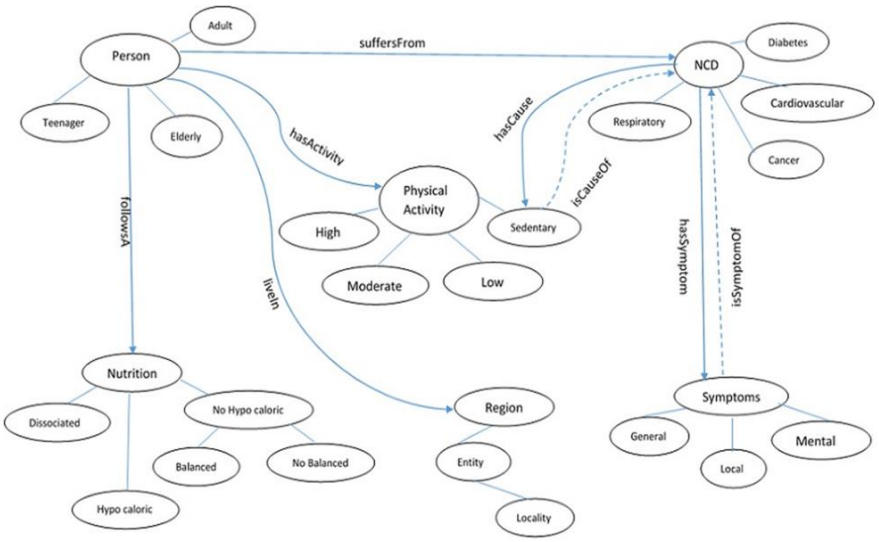


Fig. 2. Ontologies system relationships.

3.2 Individual Level Ontologies Design

Once the design stage is achieved, the ontologies which have been described in stage 1 and will form part of the system will be designed.

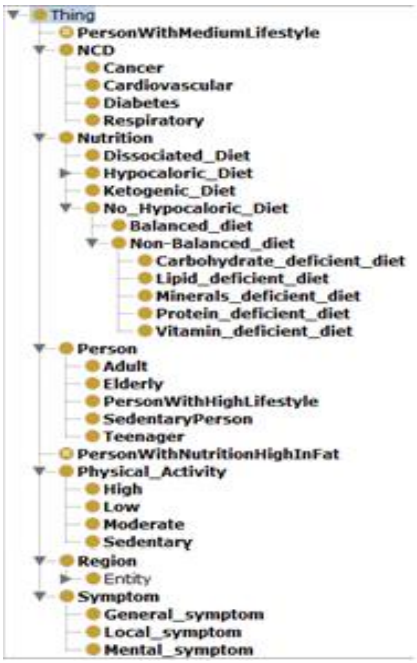


Fig. 3. Lifestyle Pattern Ontology.

In Fig. 3, these ontologies are presented as classes, once imported by the Protége's Import Ontologies tool. As an test of the system, in order to check the consistency of SWRL Rules, the OntoMex ontology have been divided, only a part of it is actually in use. The aforementioned extracting process was made to avoid overflow memory.

In Fig. 4, the ontologies system's graph is shown. Note that this diagram is not complete yet, and in subsequent works the specialization in some ontologies will be make.

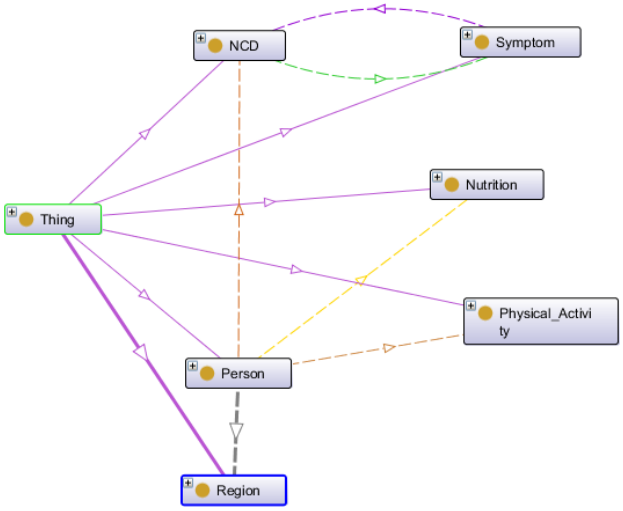


Fig. 4. Graph of the system's classes in Protégé.

3.3 Integration & Evaluation

Finally the Integration & Evaluation Stage is defined. This stage will be performed by using the SWRL Rules in order to deduct the pattern lifestyles. First of all, the axioms which are operating in the system will be designed and implemented. Those axioms are the following:

- Datatype properties,
- Object properties,
- Classes needing to be inferred,
- Necessary & sufficient conditions.

The SWRL Rules which are being used are shown in Fig. 5.

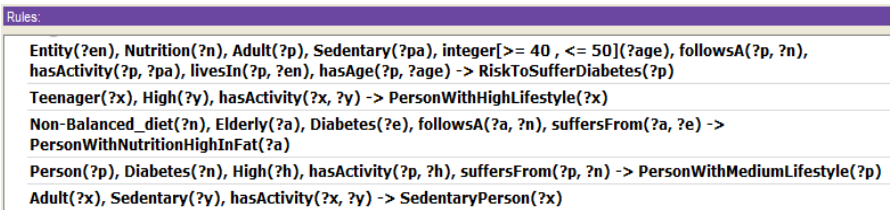


Fig. 5. Ontologies system's SWRL Rules.

In actual state of the system, SWRL rules have been checked with 85 individuals and an automatic insertion from public datasets have been in research.

As was being described, the *datatype properties axioms* or restrictions were used, then, a rule like (1), classifies a person if is an adolescent by means of their age range:

Person(?p), integer[>=12, <=17](?age), hasAge(?p, ?age) -> Teenager(?p). (1)

Datatype properties are not enough to express all the capabilities of systems relationships', i.e., accessing to the other ontologies (classes) using or design and implement another rule like (1). As a result, *Object Properties* were the next phase in the system axiomatization.

Object Properties and its proper definition of classes range and domains, guarantees the integration in a more specific level. Rule 2 was defined in order to verify the statement, and it expresses that "*If a person has a little or none physical activity, then is a sedentary person*".

Person(?p), Physical_Activity(?act), hasActivity(?p, ?act) -> SedentaryPerson(?p). (2)

SedentaryPerson have been defined and declared as a Person's subclass. Then, *SedentaryPerson* will be an Inferred Class when running the reasoner and it will contain the individuals fulfilling these restrictions.

The other Inferred Class at this stage will be *PersonWithNutritionHighInFat*, and its correspondent rule is shown in (3), which classifies an Elderly Person with high fat consumption and suffering Diabetes.

Non-Balanced_Diet(?n), Elderly(?a), Diabetes(?h),followsA(?a, ?n), suffersFrom(?a, ?e) -> PersonWithNutritionHighInFat (?a). (3)

4 Deduction by applying SWRL Rules

The mid-level lifestyle is founded by the rule in (4) which states "if a person suffers from diabetes, but have a high physical activity, then their lifestyle can be mid-level". In Fig. 6 an example of the mid-level pattern is shown.

Person(?p), Diabetes(?n), High(?h),hasActivity(?p, ?h), suffersFrom(?p, ?n) -> PersonWithMediumLifestyle(?p). (4)

The following lifestyle pattern is called the *RiskToSufferDiabetes*, as it's shown in Fig.7. It is founded by the application of rule 5 which classifies a person with a risk to suffer Diabetes. Knowledge required in rule 5 is about its physical activity which turns out to be a sedentary one, age range plus 40, living in a particular Country's State (Entity), a northern State as Nuevo Leon as an example.

Entity(?en), Nutrition(?n), Adult(?p), Sedentary(?pa), integer[>=40, <=50](age), followsA(?p, ?n), hasActivity(?p, ?pa), livesIn(?p, ?en), has Age(?p,?age) -> RiskToSufferDiabetes(?p). (5)

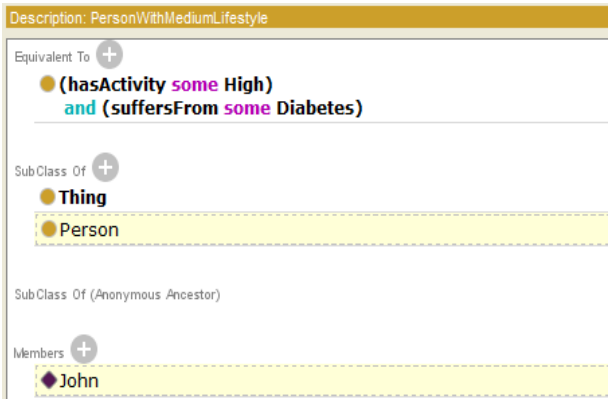


Fig. 6. Medium Lifestyle Pattern´s example.

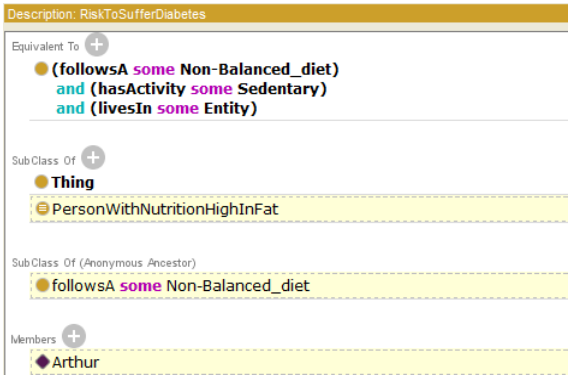


Fig. 7. RiskToSufferDiabetes Lifestyle Pattern.

As a consequence of *RiskToSufferDiabetes* pattern, *RiskToSufferDisease* can be deducted by interchanging certain parameters to concur the value partitions with, and the result is shown in Fig. 8. The pattern it's founded by rule (6), which in natural language can be expressed as: classify a person with a high level of risk to suffer a disease, if this person has a sedentary physical activity and age about plus 40.

$$\text{High\_Level}(?hl), \text{Person}(?n), \text{Sedentary}(?pa), \text{hasActivity}(?p, ?pa), \text{hasLevelOfRisk}(?s, ?lr) \rightarrow \text{PersonWithHighRiskToSufferDiabetes}(?p). \quad (6)$$

It has High Level of values of LevelOfRisk value Partition.

Finally in Fig. 9, a later reasoner' run it is shown, denoting *RiskToSufferDiabetes* class inferred as subclass of *PersonWithNutritionHighInFat*. This statement is obvious in a normal human knowledge and reasoning. However, OWL-DL[7] 2 is based in OWA, which states that a statement is not false until it is completely proven to be false, i.e., it must have been clearly expressed whatever class A is a subclass of another class B, by means of some restriction that clearly implied the fact. If such restrictions were not correctly defined or did not exist, the reasoner will infer the

[7] Web Ontology Language based in Description Logic

information not complete; therefore, class A could be subclass of another class C, instead of class B.

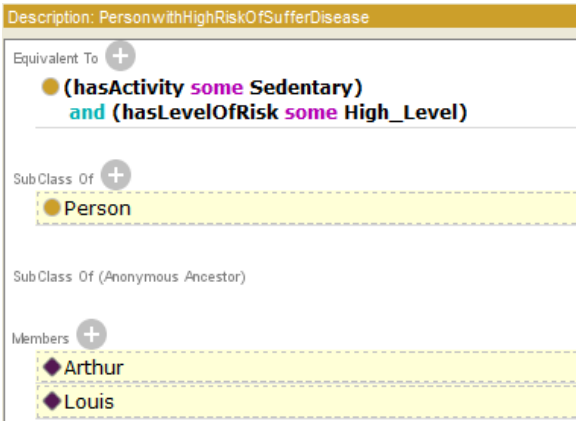


Fig. 8. RiskToSufferDisease Lifestyle Pattern.

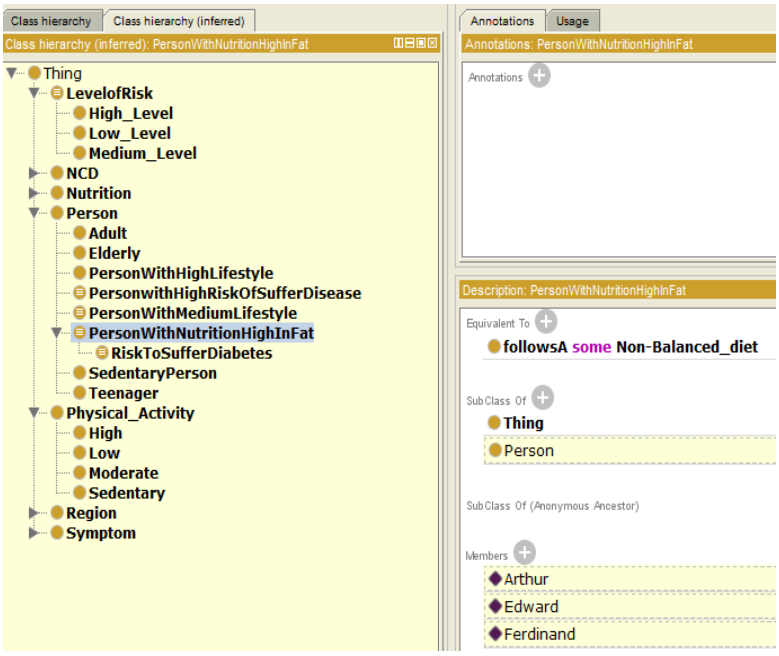


Fig. 9. Reasoner running with RiskToSufferDiabetes as a subclass.

5 Conclusions

The lifestyle patterns found through the SWRL rules application have been useful in discovering relationships among multiple dimensions of information such as: people, symptoms, NCDs, physical activity and geographical regions.

The ontology system is useful to make decisions about multiple data, all related and contained in different ontologies. Also, it shows the descriptive logic rules can be able to response the competence questions resulting from the analysis of requirements. Defining necessary and sufficient conditions' axioms allow inferred results showing more accuracy. As an example, it means to infer correctly classes as child classes or subclasses of other classes with similar restrictions. This particularly obeys to characteristics of individuals which are member of the aforementioned classes.

Ongoing work is about to migrate the current system to a server approach. Under this platform OntoMex could be loaded entirely and SWRL Rules could also be checked with all the individuals. This will allow inclusion of certain localities which could not be added for storage costs reasons.

Finally, other specializations are being planned related to the nutrition and symptom ontologies. Such that specializations permit more precise scenarios when dealing with specific diseases in order to have inferred diagnostics closer to actual cases.

Acknowledgments. We thank the Vice-Rector for *Docencia* at the Autonomous University of Puebla and the National Council for Science and Technology for the support offered to perform this work.

References

1. Bravo, M., Rodriguez, J., Pascual, J.: SDWS - Semantic Description of WebServices. International Journal of Web Services Research 11(2) (2014)
2. Bravo, M.: Similarity Measures for Web Service Composition Models. International Journal on Web Services Computing 5(1) (2014)
3. Somodevilla, M., Mena, I., Pineda, I.H., Perez de Celis, M.C.: Deducting Lifestyle Patterns by Ontologies' SWRL Rules. In: Spies, M., Wagner, R.W., Tjoa, A.M. (eds), 26th International Workshop on Database and Expert Systems: DEXA 2015, IEEE CPS (2015)
4. Somodevilla, M., Perez de Celis, M.C., Pineda, I.H., Colmenares, L.E., Mena, I.: System Development Ontology to Discover Lifestyle Patterns Associated with NCD. In: Ortuño, F. and Rojas, I. (eds), IWBBIO 2015, LNCS, Part II, Vol. 9044, pp. 47–56, Springer International Publishing Switzerland (2015)
5. Puri, C., Gomadan, K., Jain, P., Yeh, P.Z., Verma, K.: Multiple Ontologies inhealthcare InformationTechnology: Motivations and Recommendation for Ontology Mapping. In: International Conference on Biomedical Ontologies, pp. 367–369 (2011)
6. Laurini, R.: Geographic and spatial relationships. In: Input 2012 (2012)
7. Somodevilla, M., Perez de Celis, M.C., Hernandez, J., Pineda, I.H., Colmenares, L.E.: GeoHealthOntoMex: Extending a Geographic Ontology using Data Mining. In: ENC 2014, (2014)
8. Somodevilla, M., Perez de Celis, M.C., Pineda, I.H., Hernandez, J., Carrillo, M., Mena, I.: Development of an Ontologies System for Spatial Biomedical Applications. In: MICAI 2014, pp. 24–28 (2014)
9. Rodriguez, J., Romero M., Bravo, M.: Dynamic Context for Document Search and Recovery. In: Murgante, B., Misra, S., Carlini, M., Torre, C., Nguyen, H.-Q., Taniar, D., Apduhan, B.O., Gervasi, O. (eds.) Computational Science and its Applications, pp. 452–663. Springer Berlin Heidelberg (2013)

10. Bravo, M.: Ontology to Represent Similarity Relations between Public Web Services. In: Meersman, R., Dillon, T., Herrero, P., (eds.). On the Move to Meaningful Internet Systems: OTM 2011 Workshops, LNCS, Vol. 7046, pp.434–443, Springer Berlin Heidelberg (2011)
11. Donnelly, M., Bittner, T., Rosse, C.: A formal Theory for spatial representation and reasoning in biomedical ontologies. Art. Int. in Med. 36(1), pp. 1–27 (2006)
12. Smith, B., Jose, L.V., Mejino, Jr., Schulz, S., Kumar, A., Rosse, C.: Anatomical Information Science. In: Cohn, A.G. and Mark, D.M. (eds), COSIT 2005. LNCS, Vol. 3693, pp. 149--164, Springer-Verlag Berlin Heidelberg (2005)
13. Donnelly, M., Bittner, T.: Spatial Relations between Classes of Individuals. In: Cohn, A.G. and Mark, D.M. (eds), COSIT 2005. LNCS, Vol. 3693, pp. 182–199, Springer-Verlag Berlin Heidelberg (2005)
14. Fielding, J., Marwede, D.: The Image as Spatial Region: Location and Adjacency within the Radiological Image. In: Bennett, B. and Fellbaum, C. (eds), Fourth International Conference FOIS 2006, IOS Press, pp. 89–100 (2006)

Modelos para detectar la polaridad de los mensajes en redes sociales

Yuvila M. Sanzón, Darnes Vilariño, María J. Somodevilla, Claudia Zepeda, Mireya Tovar

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Puebla,
México

{yuvisrosas, dvilarino, mariajsomodevilla, czepdac}@gmail.com, mtovar@cs.buap.mx

Resumen. En el presente artículo se presentan dos modelos para descubrir la polaridad de mensajes en redes sociales, en particular extraídos del Twitter. El primer modelo extrae las características léxico-sintácticas de cada tweet. El segundo modelo obtiene las características de cada tweet basándose en la centralidad de grafos.

Palabras clave: Análisis de sentimientos, redes sociales, grafos de co-ocurrencia.

1. Introducción

Gracias a la expansión de la Web 2.0 y a la participación activa de los usuarios en redes sociales, blogs, foros y páginas dedicadas a críticas *(reviews)* en los últimos años, se ha visto un crecimiento exponencial de la información subjetiva[1] disponible en Internet. Este fenómeno ha originado interés por detectar sentimientos, emociones y opiniones expresadas sobre tópicos u objetos diferentes. De esta manera, surge la necesidad de contar con herramientas que detecten, extraigan y estructuren dicha información subjetiva.

El análisis de sentimientos se ha abierto camino en años recientes y han aparecido multitud de escenarios de uso y aplicaciones. La relevancia de herramientas de este tipo radica en la posibilidad de evaluar las opiniones expresadas por los usuarios acerca de un tópico u objeto de interés y encontrar problemas, debilidades y fortalezas en diferentes aspectos de los productos y servicios que consumen los mismos. Es posible también medir el grado de satisfacción de los usuarios acerca de un fenómeno

[1] Es la información que se presenta desde un solo punto de vista. Generalmente, expresa la interpretación o perspectiva de una persona o de un grupo de personas.

y predecir su evolución (el "sentimiento del mercado") o incluso medir la tendencia de la preferencia política. Esa información es clave para identificar áreas de oportunidad de desarrollo o cambio en la imagen de un producto, una campaña o simplemente observar la opinión popular acerca de un tópico de interés general.

El análisis de sentimientos impone retos para el Procesamiento del Lenguaje Natural (PLN), principalmente debido a la diversidad de dominios sobre los cuales se expresan las opiniones, la informalidad en la escritura de los textos y la falta de conjuntos de datos muestra.

En el presente trabajo se presentan dos modelos que permiten detectar la polaridad de un mensaje, el primero utiliza características léxico- sintácticas y el segundo extrae las características a utilizar a través de un grafo de co-ocurrencia. La polaridad se refiere a la presencia o ausencia de partículas gramaticales que definen si la oración es positiva o negativa.

En la sección 2 se presentan los diversos trabajos acerca del análisis de sentimientos. En la sección 3 se describe la metodología que se empleó para realizar los modelos propuestos. Finalmente en la sección 4 se muestran los resultados obtenidos.

2. Estado del arte

Se han desarrollado distintos trabajos en el área de análisis de sentimientos, la mayoría de estos se han centrado en el idioma inglés, dado que existe una gran cantidad de herramientas de procesamiento del lenguaje disponibles y se dispone de conjuntos de datos que pueden ser usados para el entrenamiento y creación de modelos de clasificación.

En [1] se propone el desarrollo de un motor de detección (disponible en www.umigon.com) que está diseñado específicamente para detectar el sentimiento positivo, negativo o neutral en *tweets*, el cual consta de cuatro partes principales: detección de rasgos semánticos del tweet como emoticones y onomatopeyas; evaluación de los hashtags; descomposición del *tweet* en una lista de n-gramas, comparando cada n-grama con los términos léxicos, en caso de coincidencia se aplica una heurística; finalmente, aprovechando los rasgos semánticos detectados anteriormente, se aplica una serie de heurísticas a través de todo el *tweet*. Dado que los emoticones y onomatopeyas tienen fuertes indicios de sentimiento, pero también tienen una gran variedad ortográfica, se tiene una lista con las exclamaciones más comunes y se utilizan expresiones regulares para capturar la variedad de formas que pueden asumir. Para la evaluación de los *hashtags* se aplica una serie de heurísticas de forma que el *hashtag* (en caso de que sea un *hashtag* compuesto, éste se descompone) coincida con los términos léxicos. El *tweet* es descompuesto en una lista de unigramas, bigramas, trigramas y tetragramas, se recorren todos los *n-gramas* del *tweet*, y se realizan comprobaciones de su presencia en diccionarios de términos léxicos. Si un *n-grama* es encontrado en alguno de los diccionarios, se aplica la heurística adjunta a este término presente en aquel diccionario, regresando una clasificación (positivo, negativo o neutro) para ese n-grama. Para el análisis de sentimientos se utilizaron cuatro diccionarios: tono positivo, tono negativo, fuerza del

sentimiento y negaciones; los cuales fueron creados manualmente. Con este sistema se obtuvo una precisión promedio (positivo y negativo) de 69.02%.

Otro sistema desarrollado que se debe destacar es el presentado en [2], para su implementación se hizo uso del kit de herramientas MALLET (paquete basado en Java para el procesamiento de lenguaje natural). Para la normalización de los *tweets* se realizaron las siguientes tareas: todas las palabras se convierten a su forma minúscula (utilizando el algoritmo de *Porter Stemming*), se sustituyen @ y # por las notaciones `[usuario]`, `[tag]` respectivamente, los emoticones se clasifican en positivos y negativos, se remueven caracteres innecesarios, en el caso de palabras que contienen repeticiones de caracteres se reduce la longitud, sólo teniendo en cuenta una secuencia de tres caracteres, con el fin de unificar estas repeticiones, finalmente se realizó un filtrado de palabras (palabras no significativas). Después de la normalización de los mensajes se determinó la polaridad de cada palabra utilizando el diccionario de sentimientos *SentiWordNet*, se consideró una palabra como positiva si el valor positivo relacionado es mayor a 0.3; como negativa, si el valor negativo relacionado es mayor a 0.2 y como neutral si el valor relacionado es mayor a 0.8. Una vez calculada la polaridad se contemplan tres características para cada *tweet* que son: el número de palabras positivas, negativas y objetivas respectivamente; se verifica si a una palabra positiva le precede una negación, si es así la polaridad se invierte; se utilizó un diccionario de siglas y para cada sigla se utilizó una polaridad. El mejor resultado que se reporta es de una precisión del 54%, esto usando un modelo basado en la obtención de características y la normalización de los tweets, además de usar como clasificador de aprendizaje de máxima entropía.

Encontrar las características o elementos relevantes presentes en un texto, es una parte fundamental en el proceso de clasificación supervisada. En la literatura existe una amplia variedad de trabajos relacionados a la extracción de características, pero de los trabajos que hacen énfasis en la extracción de características para el análisis de sentimientos se pueden encontrar como relevantes los descritos en [3] y [4]. En [3] y [4] se muestra que el uso de las categorías gramaticales[2] (PoS tag, Part of Speech tag) de las palabras como características sintácticas puede ayudar de forma simple a desambiguar la polaridad de las palabras. También se muestra que el uso de frecuencia de aparición de los sustantivos y los adjetivos es importante para identificar la subjetividad de las oraciones para todas las categorías gramaticales. Además, en [5] se muestra que el uso de ciertas reglas asociadas a las categorías gramaticales puede ser de utilidad para detectar patrones de sentimiento dentro de los textos. Entre los patrones más relevantes se muestra que el uso de adverbios puede ayudar a detectar la negación de las oraciones (asociado normalmente a un sentimiento positivo).

La representación de la información permite conjuntar todas las características de un texto en un esquema específico, el cual facilita la construcción de modelos de clasificación para descubrir el sentimiento de un texto dado. En el resto de esta sección se mencionan algunos de los artículos relacionados con las representaciones usadas para el análisis de sentimientos.

[2] Proceso de asignar una etiqueta gramatical a cada una de las palabras de un texto según su categoría léxica.

En [6] se presenta una comparación sobre el uso de distintas formas de representación del conocimiento para la minería de opinión. El artículo destaca el uso de una representación vectorial basada en la frecuencia de aparición de los elementos, como una forma de encapsular información numérica relevante de una forma fácil y optimizada, pero que ignora la información estructural y semántica presente en los textos, es decir, toda la información acerca de cómo están relacionadas las palabras entre sí, ya sea dentro de un párrafo o una oración, así como la relación que guardan las palabras por medio del contexto que las rodea.

En [7] se describe el uso de una representación vectorial basada en la presencia de las características como una forma reducida de la frecuencia de aparición en el contexto de la clasificación supervisada. En el artículo se muestra que una representación basada en la presencia de las características puede ser de utilidad en problemas donde no exista una gran cantidad de documentos asociados al entrenamiento ya que sólo es necesario cuantificar una sola ves un elemento para que tenga una fuerte presencia dentro del vector.

En [8] se introduce el uso de grafos para representar la información de los textos en la fase de entrenamiento. En específico se propone el uso de una representación basada en la ocurrencia de términos en una ventana de tamaño predeterminado llamada co-ocurrencia, usada para determinar relaciones de proximidad semántica relevante. Como aportación adicional se propone el uso de técnicas de ranking sobre los nodos del grafo asociado al uso de medidas de similaridad como la distancia euclidiana, entre otros.

En [9] se presenta el uso de técnicas de análisis de redes sociales para determinar la importancia de los nodos a través de las relaciones que estos forman y los caminos posibles entre ellas. En específico se propone el uso de varias métricas novedosas entre las que se encuentran las centralidades de grado, de cercanía, entre otras; con el propósito de analizar el comportamiento de una red de nodos y cuáles son los nodos más importantes y centrales por los que debe de propagarse la información.

A pesar de que se han realizado diversas investigaciones que permiten detectar la polaridad de un cierto mensaje, los resultados que se han obtenido dependen mucho de las características del corpus de entrenamiento, en este sentido es conveniente buscar la forma de extraer características que no estén asociadas directamente al corpus, sino a la forma en que se expresan los usuarios en las redes sociales. Esto justifica perfectamente la presente investigación.

3. Metodología

Para descubrir la polaridad de mensajes en redes sociales y en específico los mensajes provenientes del tweeter, se han desarrollado dos modelos implementados en el lenguaje Python, con ayuda de las herramientas Network X y CLIPS Pattern. El primero se basa en las características léxico – sintácticas de cada *tweet* y consta de tres fases (normalización, entrenamiento y prueba). El segundo modelo obtiene las características de cada *tweet* basándose en la centralidad de grafos aplicada a todo el corpus de entrenamiento, este modelo se compone de cinco fases (normalización, representación del grafo, selección de características y representación vectorial, entrenamiento y prueba). En esta sección se detallan ambos modelos.

3.1. Modelo léxico-sintáctico

Este modelo esta implementado en tres fases (Fig. 1). Para el desarrollo del mismo se utilizó el corpus de la competencia SemEval 2014, el cual contiene un conjunto de *tweets* etiquetados con cinco sentimientos diferentes (positivo, negativo, neutral, objetivo y objetivo o neutral). Sólo se trabajó con las clases: positivo, negativo y neutral. Las tres fases mencionadas anteriormente se describen a continuación:

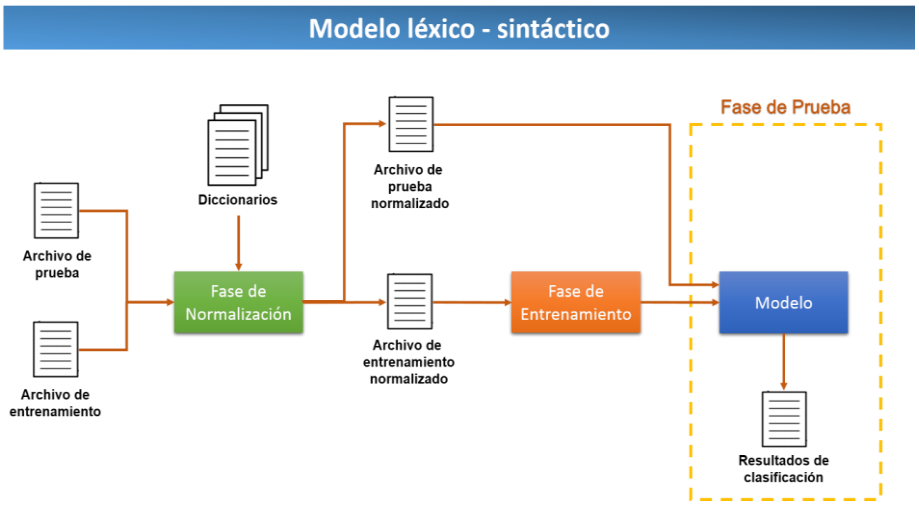


Fig. 1. Arquitectura del modelo léxico-sintáctico.

Fase de normalización. En esta fase se realiza el pre-procesamiento de los datos de entrenamiento y de los datos de prueba, para esto se desarrollaron dos diccionarios de forma manual, el primero contiene emoticones y una palabra representativa de su significado; el otro diccionario contiene algunas de las siglas empleadas en redes sociales y su significado. Esta fase comienza convirtiendo el contenido de los archivos a su representación en minúsculas, posteriormente se reemplazan los emoticones y abreviaturas, por la palabra o palabras correspondientes. Se eliminaron elementos como las *URLs*, los *hashtags* y los nombres de usuario, debido a que se consideró que no eran candidatos a características que permitiera detectar algún sentimiento. Al finalizar esta fase se obtienen los archivos de entrenamiento y prueba que son utilizados en la fase de entrenamiento.

Fase de entrenamiento. En esta fase se utilizan los clasificadores Naive Bayes y Máquina de Soporte Vectorial, proporcionados por la herramienta CLIPS Pattern. El modelo de clasificación es desarrollado con cada uno de los *tweets* contenidos en el archivo de entrenamiento normalizado, este modelo se utiliza para clasificar los datos en la fase de prueba.

Fase de prueba. En esta fase se emplea el archivo de prueba normalizado, de éste se extraen los *tweets* que serán enviados al modelo quien se encargará de asignar un sentimiento a cada *tweet*.

3.2. Modelo sobre grafos

Este modelo está formado por cinco fases como se aprecia en la Fig. 2, al igual que en el modelo anterior se emplearon los corpus proporcionados en la tarea 9: análisis de sentimientos en Twitter del SemEval 2014. A continuación se describe de forma detallada el proceso empleado en este modelo.

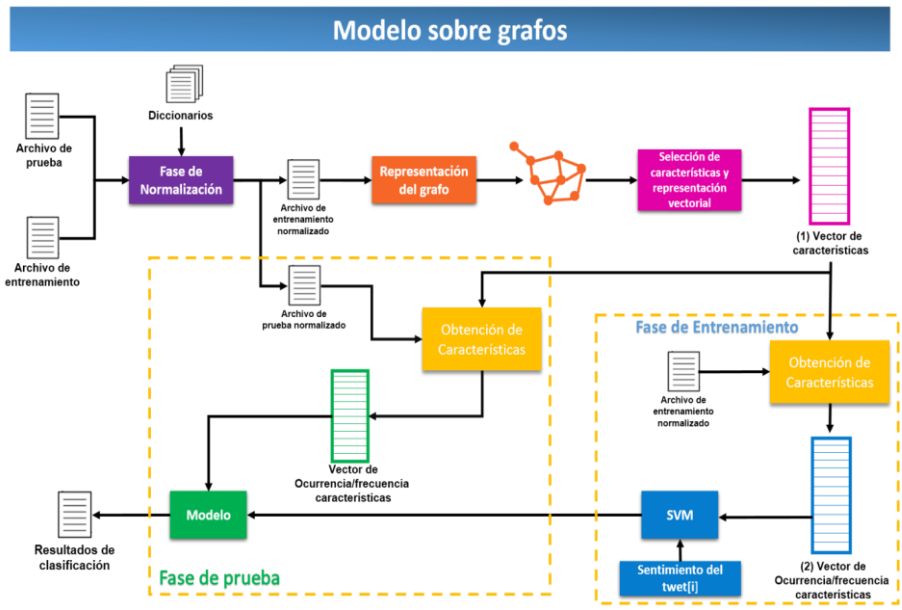


Fig. 2. Arquitectura del modelo sobre grafos.

Fase de normalización: al igual que en el modelo anterior se realizó un pre-procesamiento de los datos de entrenamiento y de prueba. Todos los tweets se llevan a minúsculas, posteriormente se reemplazan los emoticones y abreviaturas por su significado, utilizando los diccionarios antes mencionados, por último se eliminan *URLs*, *hashtags* y los nombres de usuario.

Representación del grafo:
Un grafo se define como un par (V, E), donde V es un conjunto no vacío cuyos elementos son denominados vértices o nodos y E es un subconjunto de pares no ordenados de vértices y que reciben el nombre de aristas o arcos [10].
Entre las distintas propuestas para la representación de grafos en el análisis de textos, la co-ocurrencia de palabras se ha convertido en una forma simple, pero eficaz de representar la relación de un término con respecto a otros en un grafo. Formalmente dos términos co-ocurren si están presentes en una ventana[3] de texto N [8]. Tomando en cuenta lo anterior, un grafo de co-ocurrencia no dirigido, es representado por:

[3] Se refiere a la cantidad de n sucesivas palabras con las que tendrá una conexión cada palabra en el texto.

G = (V, E), donde:

- V, es un conjunto de vértices que está formado por los términos contenidos en uno o varios textos.

- E, es un subconjunto de pares de vértices, que representa la relación entre los términos que forman dichos vértices.

En esta fase, se construyó la representación del corpus de entrenamiento por medio de un grafo, empleando el siguiente procedimiento.

1. Se crea un grafo vacío no dirigido.
2. Se obtienen las palabras que componen cada *tweet*.
3. Para cada palabra que se obtuvo se agrega una arista dentro del grafo, que una a esta palabra y a las siguientes *n* palabras, donde *n* es el valor de la ventana.
4. Se repite el procedimiento a partir del paso 2 para cada *tweet* dentro del corpus.

De manera que cada palabra distinta en el corpus se convierte en un nodo dentro del grafo, así bien los nodos se conectarán con otros nodos, si las palabras que representan dichos nodos co–ocurren dentro del valor de la ventana. En las Fig. 3 y 4 se ejemplifican la representación del grafo de co–ocurrencia con un ancho de ventana igual a 2 y un ancho de ventana igual a 3 respectivamente, utilizando los siguientes tres *tweets*, a los cuales ya les fueron removidas las *stopwords.*

1. count day tomorrow shouldnt winans crazy,
2. finish watching vow tomorrow cute movie,
3. excited nuggets game tomorrow.

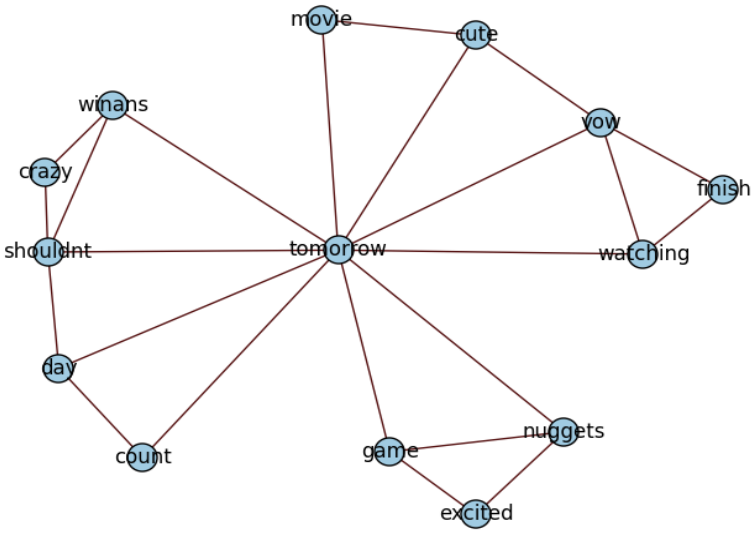


Fig. 3. Grafo con ventana igual a 2.

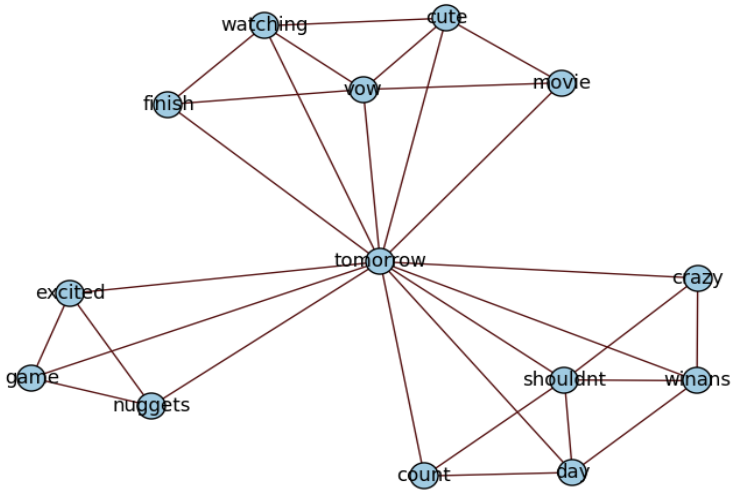


Fig. 4. Grafo con ventana igual a 3.

Se realizaron dos variaciones de este modelo, la primera utilizando todo el corpus de entrenamiento y la segunda separando el corpus de entrenamiento en tres archivos (positivo, neutral y negativo) obteniendo así tres grafos. Estas representaciones se discuten a continuación.

a) Para la representación utilizando todo el corpus de entrenamiento, en primer lugar se eliminaron las *stopwords*[4] del archivo de datos. Con el archivo resultante de este pre-procesamiento se construyó el grafo de co–ocurrencia con un ancho de ventana igual a 2 y ancho de ventana igual a 3, usando la herramienta Network X.
b) De igual manera se eliminaron las *stopwords* del corpus de entrenamiento. El archivo de entrenamiento sin *stopwords*, se separó en tres archivos: *tweets* positivos, *tweets* neutros, *tweets* negativos. Para cada uno de estos archivos se construyeron los grafos de co–ocurrencia con un ancho de ventana igual a 2 y un ancho de ventana igual a 3.

Selección de características y representación vectorial: En esta fase se describen dos variaciones una con el grafo conformado por todo el corpus de entrenamiento y otra con los tres grafos obtenidos por la separación del corpus con respecto a su etiqueta de sentimiento. A los grafos les fue aplicado un algoritmo de centralidad. Para las pruebas se emplearon cuatro de estos algoritmos:

1. Centralidad de grado: Corresponde al número de enlaces que posee un nodo con los demás [11].
2. Centralidad de cercanía: La suma o bien el promedio de la distancias más cortas desde un nodo hacia todos los demás en un grafo [11].

[4] Palabras sin significado como artículos, pronombres, preposiciones, etc.

3. Centralidad de intermediación: Es una medida que cuantifica la frecuencia o el número de veces que un nodo actúa como un puente a lo largo del camino más corto entre otros nodos [11].
4. Centralidad de vector propio: Mide la influencia de un nodo en una red. Los nodos que poseen un valor alto de esta medida de centralidad están conectados a muchos nodos que a su vez están bien conectados [11].

Dichas medidas las posee la herramienta Network X, estos algoritmos retornan un diccionario[5] de Python con los nodos (palabras) y su valor de centralidad, ya sea de grado, cercanía, intermediación o vector propio. A continuación se describe el resto del proceso para las dos alternativas de selección de características.

a) A partir de los valores contenidos en el diccionario, éstos se ordenaron de mayor a menor, se eligieron los 300 nodos (palabras) más centrales, tomando en cuenta sólo: adjetivos, adverbios, verbos, adverbios comparativos, adverbios superlativos, adjetivos comparativos, adjetivos superlativos, interjecciones, verbos en tercera persona, en presente, en pasado, en gerundio y sustantivos, estas fueron las palabras seleccionadas para obtener el vector de características de tamaño 300.
b) Para esta variación se tienen tres grafos, uno para cada sentimiento, a cada grafo le fue aplicado un algoritmo de centralidad, obteniendo así un diccionario con los valores de centralidad. Los valores contenidos en cada diccionario se ordenaron de mayor a menor, y se seleccionaron 100 nodos (palabras) más centrales de cada diccionario, tomando en cuenta sólo: adjetivos, adverbios, verbos, adverbios comparativos, adverbios superlativos, adjetivos comparativos, adjetivos superlativos, interjecciones, verbos en tercera persona, en presente, en pasado, en gerundio y sustantivos. Finalmente se unieron las 100 palabras más centrales de cada diccionario para formar el vector de 300 características.

Fase de entrenamiento: Para esta fase se crean dos tipos de vector: uno de ocurrencia y otro de frecuencia. Los tipos de entrenamiento realizados se describen a continuación.

a) Con el vector de características obtenido en la fase anterior, se procede a obtener características. Se calcula el vector de ocurrencia para cada *tweet* que se encuentre en el archivo de entrenamiento sin *stopwords*. Si la palabra *i* en el vector de características se encuentra en el *tweet*, se coloca un *1* en la posición *i* en el vector de ocurrencia, una vez terminado este proceso se entrena una máquina de soporte vectorial usando el vector de ocurrencia junto con la etiqueta de sentimiento de cada *tweet*. Obteniendo asi un modelo de clasificación supervisado.
b) Basados en el vector de características se calcula el vector de frecuencia para cada *tweet* que se encuentre en el archivo de entrenamiento sin *stopwords*. Si la palabra *i* en el vector de características se encuentra en el *tweet*, se incrementa en *1* el contenido en la posición *i* en el vector de

[5] Contenedor de pares clave – valor

frecuencia, de igual forma se entrena una máquina de soporte vectorial usando el vector de frecuencia junto con la etiqueta de sentimiento de cada *tweet*. Se obtinene un modelo de clasificación supervisado.

Fase de prueba: De la misma manera que en la fase de entrenamiento se obtiene el vector de ocurrencia o frecuencia, según sea el caso. Con el modelo obtenido se procede a clasificar los tweets que contiene el archivo de prueba normalizado, obteniendo el archivo de resultados de la clasificación.

4. Resultados

En esta sección se presentan los resultados obtenidos por los modelos para la detección de sentimiento en los *tweets*, se describen el clasificador empleado, los aciertos, y su porcentaje de precisión, finalmente se muestran gráficas para la comparación de los mejores resultados.

Como se planteó con anterioridad, el corpus tanto de entrenamiento como de prueba fue obtenido de la Conferencia Semeval 2014. El corpus de entrenamiento se compone de 6364 *tweets* (Tabla 1); se hizo una variante de este corpus pero balanceado, en el cual se igualaron la cantidad de elementos por clase, se tomaron 905 tweets de cada clase, eliminando aleatoriamente tweets de las clases que tenían un sobrante (Tabla 2). El corpus de prueba tiene un total de 8987 *tweets* (Tabla 3). Para las pruebas de ambos modelos se realizó el entrenamiento, con el corpus de entrenamiento completo y con el corpus de entrenamiento balanceado es decir, éste corpus se formó con 905 *tweets* negativos, positivos y neutros, obteniendo un total de 4525 *tweets*.

Tabla 1. Composición del corpus de entrenamiento.

Corpus de entrenamiento			
Tweets positivos	Tweets neutros	Tweets negativos	Total de tweets
2319	905	3140	6364

Tabla 2. Composición del corpus de entrenamiento balanceado.

Corpus de entrenamiento			
Tweets positivos	Tweets neutros	Tweets negativos	Total de tweets
905	905	905	2715

Tabla 3. Composición del corpus de prueba.

Corpus de prueba			
Tweets positivos	Tweets neutros	Tweets negativos	Total de tweets
3506	1541	3940	8987

4.1. Resultados del modelo léxico-sintáctico

En las pruebas de éste modelo se trabajó con el corpus de entrenamiento completo y con el corpus de entrenamiento balanceado, y los clasificadores que se utilizaron fueron Naïve Bayes y SVM.

En la Tabla 3 se muestran los resultados obtenidos con el clasificador Naïve Bayes para el corpus de entrenamiento completo, aplicando el modelo léxico – sintáctico.

Tabla 3. Resultados de precisión, corpus de entrenamiento completo, con Naïve Bayes.

Naïve Bayes Corpus de entrenamiento completo					
Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de Tweets	% de Precisión
1813	562	1873	4248	8987	47.26

Los resultados obtenidos al balancear el corpus se puede observar en la Tabla 4.

Tabla 4. Resultados de precisión, corpus balanceado, con Naïve Bayes.

Naïve Bayes Corpus de entrenamiento balanceado					
Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de Tweets	% de Precisión
1771	1434	818	4023	8987	44.76

Los resultados obtenidos con el clasificador máquina de soporte vectorial y el corpus de entrenamiento completo se describen en la Tabla 5. El único resultado significativo fue el de la prueba con el kernel lineal que arrojó una precisión del 56.58%, los demás resultados no tienen relevancia, puesto que todos los *tweets* son asignados a la clase neutral.

Tabla 5. Resultados de precisión, corpus de entrenamiento completo, con SVM.

Máquina de Soporte Vectorial Corpus de entrenamiento completo						
Kernel	Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de tweets	% de Precisión
Lineal	1929	2704	452	5085	8987	56.58
Polinomial grado 2, polinomial grado 3 y radial	0	3940	0	3940	8987	43.84

Tabla 6. Resultados de precisión, corpus de entrenamiento balanceado, con SVM.

Máquina de Soporte Vectorial Corpus de entrenamiento balanceado						
Kernel	Positivos	Tweets Neutrales	Negativos	Total de aciertos	Total de tweets	% de Precisión
Lineal	1865	1691	910	4466	8987	49.70
Polinomial de grado 2	210	1	1508	1719	8987	19.12
Polinomial de grado 3	3351	0	239	3590	8987	39.94
Radial	121	3882	67	4070	8987	45.28

En la Tabla 6 se muestran los resultados, empleando como clasificador una máquina de soporte vectorial con los *kernel* lineal, polinomial de grado 2, de grado 3

y radial, el corpus de entrenamiento utilizado fue balanceado. Contrario a las pruebas anteriores y a pesar de que no son resultados altos, la mayoría de estos predicen las tres clases, como lo es la SVM con *kernel* lineal, polinomial de grado 2 y radial.

Como puede apreciarse el resultado más alto obtenido por este modelo es de 56.58% de precisión empleando el corpus de entrenamiento completo y una máquina de soporte vectorial con *kernel* lineal.

4.2. Resultados del modelo sobre grafos

Para este modelo se realizaron 64 experimentos, incluyendo ambas variaciones del modelo, además de utilizar el vector de frecuencia y el de ocurrencia, también se probó con grafos de co – ocurrencia con ancho de ventana igual a dos y tres; se aplicaron los distintos algoritmos de centralidad; finalmente como en el modelo anterior se utilizó el corpus completo y el corpus balanceado para el entrenamiento. Cabe mencionar que en la mayoría de los experimentos con el corpus de entrenamiento completo sólo se predecía la clase neutral, por lo cual no era de utilidad. Para éste modelo, los mejores porcentajes se obtuvieron entrenando con el corpus de entrenamiento balanceado, contrario al modelo léxico – sintáctico.

Como se puede observar en la Fig. 5, los resultados más altos se obtienen utilizando el vector de ocurrencia, aplicando SVM de *kernel* lineal y centralidad de vector propio con un porcentaje de precisión de 47.34%; le sigue con vector de frecuencia, SVM de *kernel* lineal y centralidad de grado con 46.85%, finalmente con el vector de ocurrencia, SVM de *kernel* lineal y centralidad de intermediación con 46.68% de precisión.

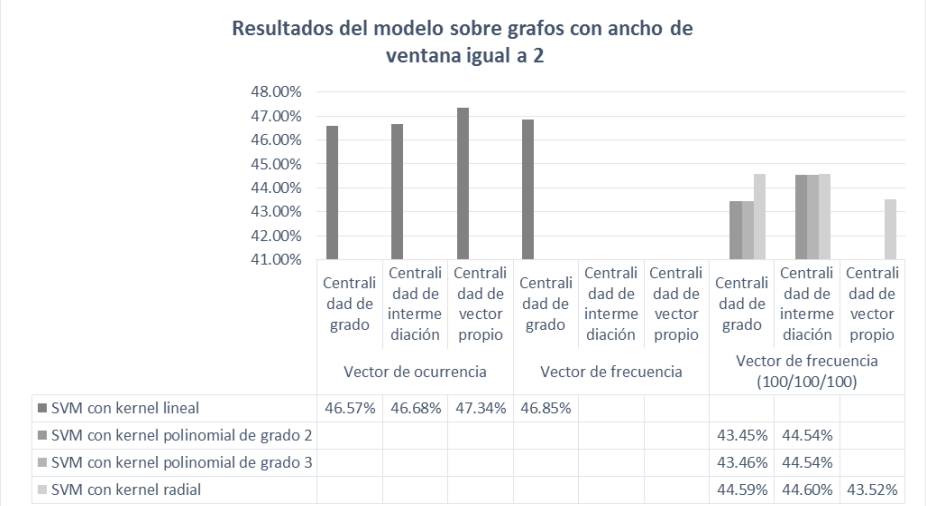


Fig. 5. Resultados del modelo sobre grafos con ancho de ventana igual a 2.

En la Fig. 6 se presentan los resultados con ancho de ventana igual a 3, con los vectores de ocurrencia (100/100/100), de frecuencia y de frecuencia (100/100/100), todos empleando el corpus de entrenamiento balanceado. El experimento con mayores

resultados fue el de vector de frecuencia (100/100/100). A continuación se enuncian los tres resultados que destacan de este conjunto de pruebas. Con vector de frecuencia, centralidad de grado y SVM de *kernel* polinomial de grado 2 y 3 se obtuvo un porcentaje de 45.64%; utilizando el vector de ocurrencia (100/100/100), SVM de *kernel* radial y centralidad de grado marco el porcentaje de 44.86%. Finalmente con el vector de frecuencia (100/100/100), centralidad de grado y SVM de *kernel* lineal fue de 44.78%.

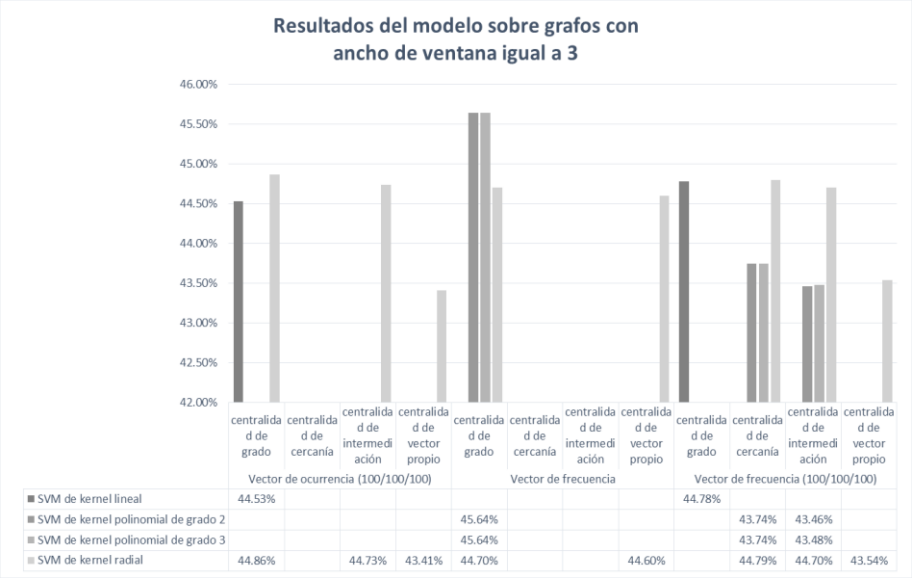


Fig. 6. Resultados del modelo sobre grafos con ancho de ventana igual a 2.

Como se puede apreciar el mejor resultado fue de 56.58% con el modelo léxico-sintáctico, en comparación con los resultados en el estado del arte, el rendimiento del modelo es bajo, pues en [1] se tiene un porcentaje de precisión 69.02%

5. Conclusiones y trabajo futuro

Se desarrollaron dos modelos de aprendizaje, uno utilizando características léxico-sintáctica y el otro extrayendo características a partir de la representación mediante grafos de los datos de entrenamiento. Además, se llevaron a cabo varios experimentos llegando a las siguientes conclusiones:

1. Es necesario balancear el corpus de entrenamiento para que ambos modelos logren descubrir las tres clases (negativo, positivo y neutro).
2. El modelo Léxico-Sintáctico arrojó mejores resultados.
3. El modelo desarrollado a partir de la selección de características utilizando los grafos de co-ocurrencias, no arrojó buenos resultados, se piensa que por

las características de los datos de entrenamiento y prueba, ya que muchos tweets son pequeñas oraciones y sin sentido.

4. El mejor comportamiento fue dado por la máquina de soporte vectorial.

Se planea seguir afinando los modelos ampliando los diccionarios de emoticones y siglas, además de crear un diccionario de *hashtags* en el que se incluya la polaridad de cada uno. También se planea la implementación de modelos que hagan uso de redes neuronales esperando un mejor resultado en la clasificación.

Referencias

1. Levallois, C.: Sentiment Analysis for Tweets based on Lexicons an Heuristics. http://www.cs.york.ac.uk/semeval-2013/accepted/27_Paper.pdf (2013)
2. Hangya, V., Berend, G., Farkas, R.: Sentiment Detection on Twitter Messages. http://www.cs.york.ac.uk/semeval-2013/accepted/102_Paper.pdf (2013)
3. Wilks, Y., Stevenson, M.: The Grammar of Sense: Using part-of-speech tags as a first step in semantic disambiguation. Journal of Natural Language Engineering, Vol. 4, No. 2, pp. 135–143 (1998)
4. Whitelaw, C., Garg, N., Argamon, S.: Using appraisal groups for sentiment analysis. In: Proceedings of the 14th ACM international conference on Information and knowledge management (CIKM'05), pp. 625–631 (2005)
5. Nasukawa, T., Yi, J.: Sentiment Analysis: Capturing Favorability Using Natural Language Processing. In: Proceedings of the 2nd international conference on Knowledge capture (K-CAP'03), pp. 70–77 (2003)
6. Serrano, J., Del Castillo, M.: Text Representation by a Computational Model of Reading. Neural Information Processing 13th International Conference, pp. 237–246 (2006)
7. Wrobel, S., Scheffer, T.: Text Classification beyond the Bag-of-Words Representation. (2002)
8. Sonawane, S., Kulkarni, P.: Graph based Representation and Analysis of Text Document: A Survey of Techniques. International Journal of Computer Applications, Vol. 96, No. 19, pp. 1–8 (2014)
9. Freeman, L.: Centrality in social networks: Conceptual clarification. Journal Social Networks – SOC NETWORKS, Vol. 1, No. 3, pp. 215–239 (1979)
10. Grafos. [Online]. http://www.ual.es/~btorreci/tr-grafos.pdf
11. Jimeng, S., Jie, T.: A survey of models and algorithms for social influence analysis. En Chary C. Aggarwal. Social Network Data Analytics, Springer, pp. 177–214 (2011)

Preprocesamiento de un corpus empleando corrección probabilística para precisar el vocabulario

Viridiana Cruz-Gutiérrez, Mario Alberto Posada-Zamora, Maya Carrillo, Luis Enrique Colmenares-Guillén, Abraham Sánchez-López

Benemérita Universidad Autónoma de Puebla, Puebla, México

{viricruz,mariop}@rockruz.net,{cmaya,lecolme,asanchez}@cs.buap.mx

Resumen. La Organización Internacional del Trabajo estimó que en el 2012 existían 20.9 millones de víctimas de explotación laboral y sexual forzada en el mundo. México ocupa el tercer lugar en trata de personas en América Latina y el Caribe. Particularmente, Puebla se encuentra entre los estados donde hay mayor vinculación de desaparición de mujeres y niñas con la trata y feminicidio. Ante esta situación estamos interesados en desarrollar herramientas que puedan ser utilizadas por padres y autoridades para la prevención de delitos ligados a trata de personas por Internet. El presente trabajo representa uno de los primeros pasos en esta dirección, se explora el preprocesamiento de un corpus de chats con contenido sexual empleando corrección probabilística, mediante teorema de Bayes. Para medir los efectos del procesamiento realizado, se trabajó en el agrupamiento de los documentos mencionados, empleando mapas auto-organizados. Los resultados obtenidos muestran que el procesamiento realizado mejora la efectividad del agrupamiento.

Palabras clave: Preprocesamiento de corpus, mapas auto-organizados, teorema de Bayes.

1. Introducción

La tecnología, sin lugar a dudas, ha sido fundamental en la evolución de la sociedad. Sin la tecnología la vida del hombre no sería como la conocemos. Es así como desde las antorchas, con las que el hombre fue capaz de salir a la obscuridad; la agricultura, que volvió al hombre sedentario; la máquina de vapor del siglo XVII; el primer foco inventado por Thomas Alva Edison, hasta la Internet y los *smartphones*, la tecnología ha impulsado un sinfín de cambios en la sociedad.

La tecnología ha permitido que en los últimos años, nos enteremos casi instantáneamente de lo que ocurre en todos los rincones del mundo, pero también ha provocado que la sociedad sea vulnerable a nuevas formas de victimización,

dando como resultado el incremento de delitos, donde los niños son los más propensos a sufrir ataques por parte de cibercriminales.

Hay 20.9 millones de víctimas de explotación laboral y sexual forzada en el mundo, según estimación del 2012 de la Organización Internacional del Trabajo (OIT) [4]. La asociación civil "Infancia Común" expone que la explotación sexual de niños, niñas y adolescentes (ESNNA) en el 2012 ocupaba el segundo lugar en generación de ganancias en México, ubicada en el orden de los 24 mil millones de dólares anuales por encima de la venta de armas y sólo superada por el narcotráfico [5].

Las modalidades de la ESNNA son: la prostitución, la pornografía infantil, el turismo sexual, el abuso sexual, la trata y la venta de niños, niñas y adolescentes para actividades sexuales. Según datos arrojados por la Coalición Contra el Tráfico de Mujeres y Niñas en América Latina y el Caribe (CATWLAC, por sus siglas en inglés) de cada diez personas que son víctimas de trata en el país, dos son menores de 18 años. Con base en la información recabada por la CATWLAC, México ocupa el tercer lugar en trata de personas en América Latina y el Caribe, y es un país de origen, tránsito y destino.

En el estudio "Violencia en el país, aumenta riesgos para que mujeres y niñas sean víctimas de trata", de la Coalición, los estados donde hay mayor vinculación de desaparición de mujeres y niñas con la trata y, luego, con el feminicidio, son Baja California, Puebla, Chihuahua, Oaxaca, Coahuila, Quintana Roo, Chiapas, San Luis Potosí, Durango, Tamaulipas, Estado de México, Tabasco, Guerrero, Tlaxcala, Hidalgo, Veracruz, Jalisco, Zacatecas y Nuevo León. Si bien existe un marco legal que busca eliminar la trata de personas, se necesita un verdadero trabajo de la sociedad y del gobierno para prevenir este delito [6].

Ante la situación planteada, como sociedad e investigadores estamos interesados en desarrollar herramientas que puedan ser utilizadas por padres y autoridades para la prevención de delitos ligados a trata de personas por Internet. El presente trabajo representa uno de los primeros pasos en esta dirección y si bien no pretende impactar en el estado del arte, si contribuir a probar técnicas de inteligencia artificial y procesamiento de lenguaje natural que permitan el desarrollo de dichas herramientas.

Estamos especialmente interesados en crear corpus de chats en español con conversaciones establecidas entre los posibles predadores sexuales y sus víctimas. Dichas conversaciones en español tendrán que ser ubicadas y recabadas a lo largo del proyecto. Con el objetivo de definir el método adecuado para construir dichos corpus, decidimos explorar el tipo de procesamiento previo que deberá hacerse sobre los chats que se recaben.

Dada la carencia de recursos en español, se utilizaron chats en inglés sobre los cuales se realizó un procesamiento normal: eliminación de palabras vacías, eliminación de palabras de acuerdo a su longitud, pero además se efectuó una corrección probabilística, empleando distancia de edición. Para medir los efectos del procesamiento previo realizado, se trabajó en la tarea de agrupamiento pues los documentos que se obtengan de Internet deberán ser agrupados en documentos útiles para la detección de predadores y los inútiles. El agrupamiento

de documentos se realizó con un mapa auto-organizado (SOM, por sus siglas en inglés). Los resultados obtenidos muestran que el procesamiento previo realizado sobre los documentos mejoró la efectividad del agrupamiento realizado por el SOM.

El resto de este documento está organizado de la siguiente manera, en la Sección 2 se describe brevemente lo que es un mapa auto-organizado, en la Sección 3 se presentan los pasos seguidos para la construcción del corpus de experimentación, así como el procesamiento previo y corrección del mismo. En la Sección 4 se explican los experimentos realizados y resultados obtenidos, finalmente en la Sección 5 se muestran las conclusiones y trabajo futuro.

2. Mapas auto-organizados

Se ha observado que en la corteza cerebral de los animales superiores aparecen zonas donde las neuronas detectoras de rasgos están topológicamente ordenadas; de manera que las información captada del entorno mediante los órganos sensoriales, se representan internamente en forma de mapas bidimensionales.

Aunque en gran medida esta organización neuronal está predeterminada genéticamente, es probable que parte de ella se origine mediante el aprendizaje. Esto sugiere, por tanto, que el cerebro podría poseer la capacidad inherente de formar mapas topológicos a partir de las informaciones recibidas del exterior.

También se ha observado que la influencia de una neurona sobre las demás, está en función de la distancia entre ellas, siendo pequeña cuando están alejadas. A partir de estas ideas, el académico finlandés Teuvo Kohonen presentó en 1982[7] un sistema con un comportamiento semejante. Se trataba de un modelo de red neuronal con capacidad para formar mapas de características de manera similar a como ocurre en el cerebro.

Los mapas auto-organizados de Kohonen constituyen un método de proyección no lineal que mapea un espacio de datos multidimensional en un mapa usualmente bidimensional en forma ordenada. Los nodos del mapa son asociados a los llamados vectores de referencia que actúan como modelos locales de los datos más cercanos y, más ampliamente, de las regiones vecinas del mapa. Gracias a las propiedades del algoritmo de los SOM, posiciones cercanas del mapa contienen datos similares, permitiendo una visualización intuitiva del espacio de datos. Más aún, los vectores de referencia dividen los datos en subconjuntos de datos similares, efectuando una categorización de los datos.

El proceso de aprendizaje del SOM es ilustrado a continuación:

1. Inicialice los pesos con valores al azar:

$$\forall i \in M : w_i(0) = random() \tag{1}$$

   Donde M es el número de neuronas
2. Elegir al azar un patrón $x(t)$ del conjunto de entrenamiento, para la iteración t.

3. Por cada neurona i en el mapa de características Φ, calcular la similitud entre el conjunto de pesos w_i y el patrón $x(t)$. Para esto puede usarse la distancia Euclidiana:

$$\forall i \in M : d^2(w_i, x) = \sum_{k=1}^{N} (w_{ik} - x_k)^2 \tag{2}$$

4. Encontrar un neurona ganadora i^* correspondiente a la que obtuvo la mínima distancia (máxima similitud).
5. Modificar los pesos de la neurona ganadora i^* y los de sus vecinas:

$$\forall j \in \Lambda_{i^*}(t) : w_j(t+1) = w_j(t) + \alpha(t)(x(t) - w_j(t)) \tag{3}$$

Λ_{i^*} corresponde a una función de vecindad centrada en la neurona ganadora i^* y $\alpha(t)$ es una función de proporción de aprendizaje, definida como:

$$\alpha(t) = \frac{1}{t} \tag{4}$$

o también se define de la siguiente manera:

$$\alpha(t) = \alpha_1 \left(1 - \frac{1}{t}\right) \tag{5}$$

6. Regresar al paso dos, hasta que no existan más cambios en el mapa de características Φ o hasta que el número máximo de iteraciones se alcance.

Los SOM han sido utilizados exitosamente en diversas tareas, se pueden destacar aplicaciones relacionadas con el reconocimiento de patrones (voz, texto, imágenes, señales, etc), codificación de datos, comprensión de imágenes, resolución de problemas de optimización, análisis de imágenes y monitoreo de procesos [8,9,10,11,12].

3. Desarrollo

En esta sección se muestran los pasos que se siguieron para la construcción de un corpus con chats de predadores sexuales con sus víctimas, complementado con documentos de tecnología Linux, y la experimentación con dicho corpus.

3.1. Construcción del corpus

Se construyó un corpus balanceado que permitiera comprobar si preprocesar el texto empleando técnicas de corrección probabilistica, podría contibuir a mejorar el rendimiento de la tarea de agrupamiento de textos. Como se ha mencionado previamente, para tal efecto se empleó una red neuronal de tipo mapa auto-organizado (SOM).

El corpus contiene dos grupos de documentos de chats, el primero con contenido sexual y el segundo con contenido referente a tecnologías Linux. Ambos grupos contienen únicamente texto en inglés.

Para la construcción del primer grupo de documentos del corpus, se eligieron los chats y mensajes de texto de 593 predadores sexuales, obtenidos de la página Perverted Justice[1]. Inicialmente se descargaron las páginas de la Web en texto plano. Para conformar el segundo grupo de documentos, se descargó un compendio de chats en texto plano de la página Ubuntu Chats Corpus[2].

Debido a que los chats de los predadores sexuales eran muy extensos, se separaron por fechas y posteriormente, se eliminaron fechas, nombres de usuario e información de sistema de los chats, ya que son datos irrelevantes para el agrupamiento.

Se contabilizó el número de palabras de cada chat de los predadores sexuales y se determinó que en promedio contenían entre 1,000 y 1,300 palabras, por lo que se eligió un chat de cada predador sexual con dicha extensión, obteniendo una muestra de 500 documentos. Para tener corpus balanceados, se eligió el mismo número de chats de Ubuntu que cumplieran con los criterios anteriormente mencionados.

Finalmente se eliminaron de los 1,000 documentos, caracteres especiales y palabras que no cumplieran con la codificación Unicode.

Todo el proceso de construcción del corpus, se realizó con un programa desarrollado en lenguaje Java.

3.2. Preprocesamiento

El preprocesamiento para preparar el corpus de este experimento se realizó empleado el lenguaje de programación Python 3.5 y la librería de procesamiento de lenguaje natural NLTK. Se desarrollaron tres esquemas para hacer más conciso el vocabulario generado:

1. Remoción de palabras 'vacías', signos de puntuación y números.
2. Selección de palabras cuya longitud comprende un rango específico.
3. Uso de un algoritmo de corrección de palabras.

En cada aplicación de las técnicas mencionadas anteriormente, se generó un archivo que contiene el vocabulario perteneciente al corpus resultante, el cual se puede observar como un conjunto sin repeticiones de las palabras utilizadas en todos los documentos.

La técnica de la remoción de palabras vacías se basa en el uso de un conjunto de palabras P_v, las cuales tienen poco o nulo significado y relevancia para la descripción de los documentos.

Para cada documento d del corpus C, al vocabulario generado V_d se le sustrae el conjunto de palabras vacías P_v, con el fin de obtener un nuevo vocabulario V_i' más reducido y con mayor relevancia. La ecuación 6, muestra cómo se eliminan las palabras vacías del documento.

$$\forall d \in C : V'_d = V_d - P_v \tag{6}$$

El conjunto P_v utilizado, está conformado por el conjunto de las palabras que podrían no tener significado para nuestro experimento, tales como pronombres personales, algunos verbos, auxiliares, saludos y números escritos con letra.

Para la técnica de selección de palabras de cierta longitud, por observación se dedujo que las palabras que podrían ser relevantes para representar un documento contenían generalmente de 3 a 13 caracteres, sobre todo en el ámbito de representación de un documento sexual. Algunas muestras de palabras en inglés son: *sex*, *masturbation*, *underage*, por citar algunos ejemplos.

Es entonces que el vocabulario reducido V' para esta técnica se representa de acuerdo con la ecuación 7.

$$\forall d \in C : V'_d = \{x : |x| > 2 \wedge |x| < 14\} \tag{7}$$

Otra justificación para utilizar esta medida es, particularmente para el corpus empleado, que los documentos contienen palabras que, antes de pasar por la eliminación de símbolos y caracteres especiales, eran direcciones http, lo que ocasionaba nuevas palabras en el vocabulario total, cuya longitud generalmente rebasaba los 14 caracteres continuos.

En la técnica de corrección, se empleó un corrector desarrollado en Python basado en el teorema de Bayes[3], para establecer qué tan probable es para una palabra estar mal escrita, y la probabilidad de que la palabra a la que se quiere hacer referencia sea otra. Este programa realizado por Peter Norving, director de investigación de Google, fue desarrollado con la finalidad de crear un corrector ortográfico "de juguete"(comparado con el desarrollado a nivel industrial), que tuviera una considerable precisión y velocidad de procesamiento de al menos 10 palabras por segundo.

Estamos buscando una corrección c para una palabra w de tal manera que se maximice la probabilidad de que sea la palabra buscada, la expresión 8 ilustra esta técnica.

$$argmax_c P(c|w) \tag{8}$$

La ecuación 9 hace referencia a la expresión 8 aplicando el teorema de Bayes.

$$argmax_c P(c|w) = argmax_c P(w|c)P(c)/P(w) \tag{9}$$

Reduciendo el término derecho de 9, se obtiene la expresión 10.

$$argmax_c P(w|c)P(c) \tag{10}$$

Donde $P(c)$ es la probabilidad de que c aparezca en documentos escritos bajo un lenguaje. $P(w|c)$ se refiere a la probabilidad de que la palabra w haya sido escrita queriendo decir realmente c.

Por último, $argmax_c$ es una función que selecciona de la lista todas las posibles formas aceptables de c, la mejor c cuyo puntaje de probabilidad sea el más alto.

Para realizar la corrección de palabras se utilizó la distancia de edición o distancia de Levenshtein, considerando las siguientes operaciones:

- Eliminación de un caracter.
- Trasposición de caracteres.
- Reemplazos de caracteres.
- Inserción de caracteres.

Como resultado obtenemos una palabra corregida que es bastante útil en el contexto de corpus conformados con conversaciones en línea, en las cuales existe una gran cantidad de errores.

En base a los tres esquemas enunciados al inicio de esta sección, se produjeron tres vocabularios, que se muestran en la Tabla 1.

Tabla 1. Descripción de los vocabularios y su tiempo de generación

	Vocabulario 1	**Vocabulario 2**	**Vocabulario 3**
Técnicas utilizadas	Remoción de palabras vacías	Remoción de palabras vacías y longitud acotada	Remoción de palabras vacías, longitud acotada y algoritmo de corrección de palabras
Longitud de vocabulario (número de palabras)	40,043	36,656	19,681
Tiempo de generación de vocabulario	25 seg	40 seg	2.25 horas

3.3. Representación vectorial

Una vez generado el vocabulario con las técnicas descritas en la Tabla 1, se empleó el modelo vectorial para representar los documentos, de acuerdo con el esquema de pesado tf_idf, cuyas formulas se muestran en las ecuaciones 11, 12 y 13.

$$tf = \frac{n_{i,j}}{|d_j|} \tag{11}$$

Siendo $n_{i,j}$ el número de ocurrencias de la palabra i para el documento j y $|d_j|$ la longitud del documento.

$$idf(t, D) = log\left(\frac{|D|}{|\{d \in D : t \in d\}|}\right) \tag{12}$$

La ecuación 12 hace referencia al logaritmo del número de documentos del corpus, entre el número de documentos que contienen el término t.

$$tf\_idf(t, D) = tf(t, d) * idf(t, D) \tag{13}$$

Con la ecuación 13 se da importancia a algunos términos por encima de otros, sobre todo cuando la rareza de un término en el corpus es considerable. Además con la frecuencias de las palabras en un documento en particular, se da menos prioridad a palabras que aparecen poco en dicho documento, en contraste con las que aparecen con frecuencia considerable.

4. Experimentos y resultados

Para comprobar si el preprocesamiento del corpus mejora la tarea de agrupamiento, se utilizó un SOM y se realizaron experimentos con lo tres tipos de vocabulario explicados en el apartado de Preprocesamiento. A continuación se describen las condiciones de los experimetos realizados y se presentan los resultados obtenidos.

4.1. SOM con vocabulario sin palabras vacías únicamente

En el primer experimento, los documentos se subministraron al SOM de manera ordenada: los primeros 500 eran referentes a los predadores sexuales y los últimos 500 sobre temas de Ubuntu. El resultado del SOM muestra un agrupamiento de 670 documentos en la primera clase y 330 en la segunda para una primera ejecución (Figura 1.a), mientras que para un segundo experimento, se obtuvo un agrupamiento de 672 para la primera clase y 328 para la segunda clase, como puede observarse en la Figura 1.b.

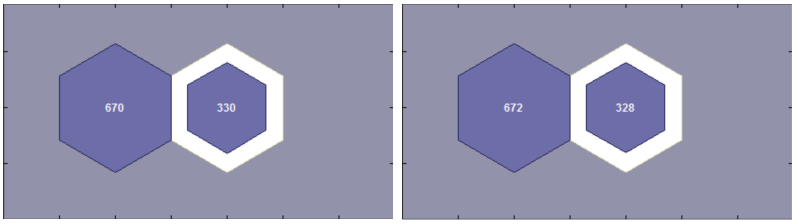


Fig. 1. Resultados de la ejecución del SOM con un corpus ordenado con el Vocabulario 1, a) Agrupamiento en primera corrida, b) Agrupamiento en segunda corrida. En ambas imagenes, el hexágono de la izquierda corresponde al agrupamiento referente a predadores sexuales y el hexágono de la derecha, a documentos de tecnología Linux.

Con este mismo vocabulario los 1000 documentos fueron proporcionados al SOM de manera aleatoria, el resultado fue muy similar a los obtenidos anterior-

mente, siendo el agrupamiento de 671 y 329 elementos para la primera y segunda clase respectivamente, el resultado se muestra en la Figura 2.

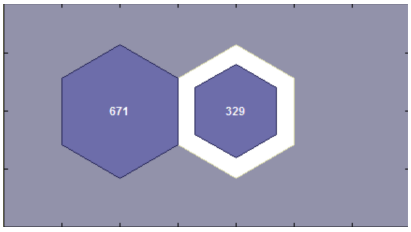


Fig. 2. Resultados de la ejecución del SOM con un corpus en orden aleatorio con el Vocabulario 1. El hexágono de la izquierda corresponde al agrupamiento de documentos sobre predadores sexuales y el hexágono de la derecha, a documentos de tecnología Linux.

En promedio el tiempo de procesamiento que le tomó al SOM para el agrupamiento fue de 125.62 segundos.

4.2. SOM con palabras de longitud acotada y sin palabras vacías

El segundo experimento se realizó con una combinación de dos técnicas, resultando para una corrida con datos ordenados de 500 documentos sobre abuso sexual seguidos de 500 documentos de chats de Ubuntu.

El agrupamiento arrojado en este experimento fue un resultado de 266 y 734 documentos para la primera y segunda clase respectivamente. Como se observa en la Figura 3.

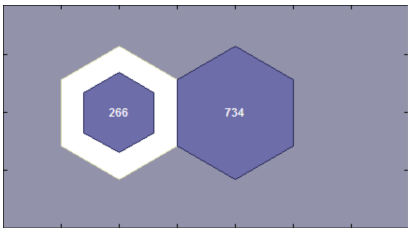


Fig. 3. Resultados de la ejecución del SOM con un corpus ordenado tomando el Vocabulario 2. Hexágono izquierdo pertenece a documentos con contenido sexual y el hexágono derecho a documentos sobre tecnología Linux.

Para ser un corpus balanceado, los resultados obtenidos fueron menos satisfactorios de lo esperado, pues estuvieron por debajo de los resultados al eliminar únicamente las palabras vacías. Analizando el vocabulario utilizado, se encontró

que la eliminación de palabras pertenecientes a direccones web provocó que los resultados no fueran favorables.

El tiempo de procesamiento que le tomó al SOM para el agrupamiento fue de 108.93 segundos.

4.3. SOM con vocabulario sin palabras mal escritas, longitud acotada y sin palabras vacías

La primera prueba del experimento con este vocabulario más reducido se realizó nuevamente con los documentos ordenados en un primer grupo de 500 y un segundo grupo de 500, para los chats de depredadores sexuales y de Ubuntu, respectivamente. El resultado del SOM mostró un agrupamiento de 500 para la primera clase y 500 para la segunda, por lo que se obtuvo el 100 % de agrupamiento correcto. Figura 4.

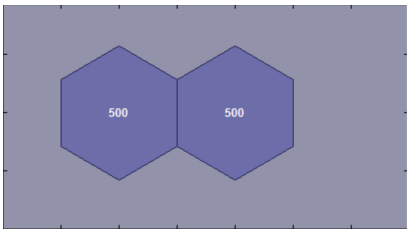


Fig. 4. Resultados de la ejecución del SOM con un corpus ordenado tomando el Vocabulario 3. Hexágono izquierdo referente a documentos con contenido sexual y hexágono derecho sobre documentos de tecnología Linux.

La siguiente prueba fue colocar únicamente 500 documentos de contenido sexual y 205 de Ubuntu, obteniendo nuevamente los resultados esperados, un agrupamiento de 500 en la primera clase y 205 en la segunda. Se decidió hacer una prueba con un corpus desbalanceado ya que los resultados obtenidos en la prueba de la Figura 4 fueron favorables, y queríamos corroborar que se podían producir resultados similares de esta manera.

La última prueba con este vocabulario fue con orden aleatorio de los 1,000 documentos. El resultado fue nuevamente exitoso, logrando agrupar el 100 % de los documentos, esto se observa en la Figura 6.

El tiempo de procesamiento en el SOM durante el agrupamiento fue de 66.24 segundos.

5. Conclusiones y trabajo futuro

Se comprobó que contar con un vocabulario preciso y de dimensión adecuada permite mejorar el desempeño de la tarea de agrupamiento, en particular en nuestro caso se logró agrupar de manera correcta el 100 % de los documentos.

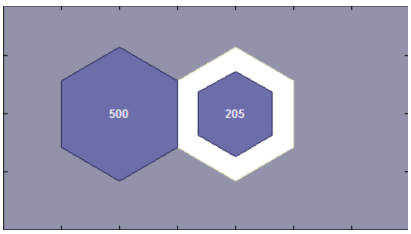


Fig. 5. Resultados de la ejecución del SOM con un corpus desbalanceado ordenado tomando el Vocabulario 3. Hexágono izquierdo referente a documentos con contenido sexual y hexágono derecho sobre documentos de tecnología Linux.

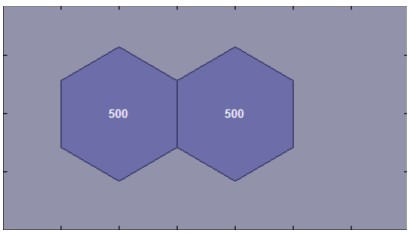


Fig. 6. Resultados de la ejecución del SOM con un corpus en orden aleatorio tomando el Vocabulario 3. Hexágono izquierdo referente a documentos con contenido sexual y hexágono derecho sobre documentos de tecnología Linux.

Aunado a esto, debe mencionarse que el tiempo de procesamiento disminuyó en un 47.27 %, ya que en cada paso del preprocesamiento se redujo la dimensión del vocabulario.

Se realizaron tres pasos de pre-procesamiento, los dos primeros fueron los comunmente usados en el procesamiento de cualquier corpus, con los cuales no obtuvimos resultados extraordinarios. Sin embargo, el tercer paso, la corrección de palabras mediante el teorema de Bayes, permitió obtener mejores resultados, gracias a que se rescataron palabras mal escritas, que son importantes dentro del contexto.

Como trabajo futuro, se pretende experimentar con otros corpus de mayor dimensión y con chats en idioma español, así como probar la eficiencia de los vocabularios generados, en algoritmos genéticos de agrupamiento.

Referencias

1. Perverted Justice, `http://www.perverted-justice.com`
2. Ubuntu Chats Corpus, `http://daviduthus.org`
3. How to write a spelling corrector, `http://norvig.com`
4. Estimación mundial sobre el trabajo forzoso (resumen ejecutivo), `http://www.ilo.org/wcmsp5/groups/public/---ed_norm/---declaration/documents/publication/wcms_182010.pdf`

5. Contralínea, http://contralinea.info/archivo-revista/index.php/2010/09/05/mexico-pasividad-ante-explotacion-sexual-infantil/
6. Terreno ideal para la trata de personas, http://eleconomista.com.mx/sociedad/2013/11/10/terreno-ideal-trata-personas
7. Kohonen, T.: The Self-Organizing Map. Proceedings of the IEEE, Vol. 78, pp. 1464–1480 (1990)
8. Honkela, T.: Emerging categories and adaptive prototypes: Selforganizing maps for cognitive linguistics. Extended abstract, accepted to be presented at the International Cognitive Linguistics Conference (1997)
9. Kohonen, T., Kaski, S., Lagus, K., Honkela, T.: Very large two-level SOM for the browsing of newsgroups. In: Proceedings of ICANN'96, International Conference on Artificial Neural Networks (1996)
10. Kurimo, M.: Fast latent semantic indexing of spoken documents by using self-organizing maps. In: IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP'00), vol. 6, pp 2425–2428, IEEE, Istanbul (2000)
11. Deboeck, G., Kohonen, T.: Visual Explorations in Finance with Self-Organizing Maps. Springer, New York (1998)
12. López, H., Machón, I.: Self-organizing map and clustering for wastewater treatment monitoring. Engineering Applications of Artificial Intelligence 17(3), 215–225 (2004)

Primera aproximación de un sistema de recuperación de información booleano con expansión semántica de consultas

Mireya Tovar Vidal, Ana Laura Lezama Sánchez, Darnes Vilariño Ayala, Beatriz Beltrán, Mauricio Castro Cardona

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Puebla,
Mexico

{mtovar,darnes,bbeltran,mcastro}@cs.buap.mx
yumita1102@gmail.com

Resumen. En el presente trabajo se propone una aproximación que utiliza la expansión de consultas en un Sistema de Recuperación de Información Booleano (SRIB), con la finalidad de mejorar el nivel de precisión de un SRIB sin expansión. Las consultas están formadas por los conceptos y relaciones existentes en ontologías de dominio. El SRIB sin expansión y con expansión asocia a cada consulta la información relevante extraída desde el corpus de dominio. En base a los resultados experimentales obtenidos, se observa que la precisión del SRIB con expansión mejora al SRIB sin expansión, al recuperar más información, incluso al identificar más conceptos con información en el corpus, que el sistema tradicional sin expansión. Se analizaron cuatro ontologías de dominio y los resultados experimentales obtenidos resultan ser satisfactorios con esta aproximación.

Palabras clave: Sistema de recuperación de información, expansión semántica de consultas, ontologías.

1. Introducción

La Recuperacón de Información*(RI)* es el área de la ciencia y la tecnología que trata de adquirir, representar, almacenar, organizar y acceder a elementos de información. Desde el punto de vista práctico, dada una necesidad de información del usuario, un sistema de RI produce como salida un conjunto de documentos cuyo contenido satisface potencialmente esa necesidad. Esta última puntualización es de suma importancia, ya que la función de un sistema de RI no es la de devolver la información deseada por el usuario, sino únicamente la de indicar qué documentos son potencialmente relevantes para dicha necesidad de información.

Hoy en día la búsqueda de información es el eje central de cualquier investigación. Las búsquedas son proporcionadas por el usuario en su lenguaje natural y se

espera que los documentos recuperados sean aquellos que satisfagan la consulta realizada.

Esta investigación parte de un sistema de recuperación de información que permite recuperar documentos de un corpus de dominio, asociados a cada concepto y relaciones de una ontología de dominio. Tales conceptos y relaciones son utilizados como consultas que se emplean en la entrada a dicho sistema. En [16] se emplea un Sistema de Recuperación de Información Booleano y la información recuperada por cada concepto y relación es utilizada posteriormente para la evaluación automática de ontologías de dominio. Con la finalidad de mejorar la precisión de este sistema, se propone la extensión al mismo. En este caso se añade únicamente la expansión semántica de los términos que forman la consulta, en este caso la consulta está formada por los sinónimos exactos de los conceptos de la ontología extraídos desde WordNet [9].

Esta investigación está estructurada de la siguiente manera: en la sección 2 se describe la información general sobre sistemas de recuperación de información, en la sección 3 se presentan algunas propuestas por diversos autores para la expansión de consultas, en la sección 4 se describe la aproximación propuesta, en la sección 5 se presentan los experimentos y el conjunto de datos y finalmente en la sección 6 se discuten las conclusiones y el trabajo a futuro.

2. Sistemas de recuperación de información

La Recuperación de Información (RI) ha sido interpretada por diversos autores. En el caso de Ricardo Baeza-Yates et al. [1] " la Recuperación de Información trata con la representación, el almacenamiento, la organización y el acceso a ítems de información". Salton [12] propuso una definición que plantea que el área de RI "es un campo relacionado con la estructura, ánalisis, organización, almacenamiento, búsqueda y recuperación de información". Croft [15] estima que la recuperación de información es el "conjunto de tareas mediante las cuales el usuario localiza y accede a los recursos de información que son pertinentes para la resolución del problema planteado". Korfhage [7] definió RI como "la localización expresada como una pregunta". De manera más genéral, se puede plantear que la recuperación de información intenta resolver el problema de "encontrar y ordenar documentos relevantes que satisfagan la necesidad de información de un usuario, expresada en un determinado lenguaje de consulta" [15].

Uno de los modelos existentes para la recuperación de información es el modelo booleano que representa la colección de documentos como una matriz binaria documento-término. Los términos son extraídos de los documentos y representan el contenido de los mismos. Se utilizan operadores lógicos: AND, OR y NOT, y los resultados son referencias a documentos, donde la representación de la consulta satisface las restricciones lógicas de la expresión de búsqueda. En el modelo original no hay orden de relevancia sobre el conjunto de respuestas a la consulta, todos los documentos poseen la misma relevancia [15].

La precisión de los sistemas de recuperación de información depende mucho de los términos que se encuentran en la consulta, es por ello que intentar de

manera eficiente expandir la consulta, puede aunmentar la cantidad y calidad de los documentos recuperados y satisfacer la necesidad de información dada por el usuario.

3. Trabajos relacionados

En el caso de la expansión de consultas por sinónimos, algunos autores han recurrido a diferentes técnicas de expansión, así como diferentes modelos de recuperación de información. A continuación se describen algunos trabajos relacionados con esta investigación.

En Cotelo et al. [3] el problema principal consiste en definir un lenguaje de consulta que sea utilizado para recibir consultas con información semántica y un algoritmo de ordenamiento que permita ordenar los documentos. Dentro de las características deseables del lenguaje de consulta se encuentran: identificar objetos y atributos de los mismos, permitir al usuario indicar el significado de una palabra polisémica, incluir semántica temporal en las consultas, expandir la consulta con sinónimos y permitir operadores sobre los predicados.

Kuna et al. [8], utiliza una ontología de dominio específico para la expansión de consultas, además de un sistema de recuperación de información para la búsqueda de documentos científicos.

En Valbuena et al. [18] se propone el uso de ontologías para garantizar que los resultados en una búsqueda hecha por el usuario, correspondan al dominio de la misma.

En Muñoz et al.[4] se propone el desarrollo de un sistema de recuperación de información en Inteligencia Artificial enfocado a textos médicos, con el objetivo de conseguir un sistema destinado a introducirse en el campo de la Medicina Personalizada y en el campo turístico.

En Hernández-Aranda et al.[6] se desarrolló un prototipo que consta de una interfaz web que permite la búsqueda y visualización de resultados a partir de una consulta dada.

Shabanzadeh et al.[14], proponen un algoritmo para la expansión de consultas basado en relaciones semánticas, utilizan Wordnet para extraer las relaciones semánticas entre palabras. Se demostró que las relaciones semánticas pueden mejorar la expansión de consultas, que las palabras vagas reducen el rendimiento de la recuperación de información.

Chauhan et al.[2], proponen la técnica de expansión de consulta semántica que incluye un modelo matemático para calcular la similitud semántica entre conceptos y un algoritmo para la expansión de consultas basado en una ontología de dominio.

En Moreno et al. [13], se implementó una búsqueda textual sobre una ontología, permitiendo obtener los conceptos de la ontología en función de una búsqueda expresada en lenguaje natural.

Neha et al. [10], proponen un algoritmo genético para la expansión de consultas hechas en lenguaje natural, se utiliza el coeficiente de Czekanowski durante el proceso de expansión, para que la recuperación de documentos sea más eficiente.

Finalmente, en Hany et al. [5] se emplea el modelo espacio vectorial que se adaptó en su propuesta de trabajo para la representación de documentos, retira palabras vacías, etc. La consulta es expandida por sinónimos extraídos de Wordnet.

En esta investigación se propone el uso de los sinónimos recuperados desde WordNet de los conceptos que integran a la ontología, para la expansión de consultas. Las consultas están formadas por las palabras de cada concepto de la ontología y por otro lado por los sinónimos de estos conceptos. También se presenta un algoritmo que realiza la unión de los documentos recuperados por el Sistema de Recuperación de Información Booleno con los conceptos y sus sinónimos correspondientes. La finalidad de esta investigación es la de incorporar información adicional, como los documentos que contienen al sinónimo del concepto y al concepto mismo, para la evaluación posterior de los mismos y las relaciones semánticas existentes en la ontología de dominio. A continuación se presenta la aproximación propuesta.

4. Aproximación para la expansión de consultas

En este artículo se plantea la expansión de consultas por sinónimos, la cual se utiliza para recuperar documentos relevantes a la misma, por medio de un sistema de recuperación de información booleano. Las consultas están formadas por las palabras que integran los conceptos extraídos de ontologías de dominio.

A continuación se presentan las etapas de la aproximación propuesta:

1. Extracción de conceptos y relaciones de las ontologías de dominio.
2. Extracción de los sinónimos de los conceptos desde WordNet.
3. Preprocesamiento del corpus de dominio, de los conceptos, de las relaciones y de los sinónimos. Esta etapa incluye las siguientes acciones:
   a) División del corpus en líneas.
   b) Eliminación de símbolos especiales, números y palabras cerradas.
   c) Aplicación de un lematizador, en particular se utiliza el algoritmo de Porter [11].
4. Formación de consultas. Existen tres tipos de consultas:
   a) Consultas formadas con las palabras del concepto.
   b) Consultas formadas con los sinónimos del concepto.
   c) Consultas formadas con los dos conceptos que forman la relación semántica.
5. Aplicación del Sistema de Recuperación de Información Booleno (SRIB) para conceptos.
6. Aplicación del Sistema de Recuperación de Información Booleno (SRIB) para los sinónimos de los conceptos.
7. Mezcla de los resultados obtenidos (posting) por el SRIB de los dos pasos anteriores. La mezcla consiste en la unión de postings sin repetir información.
8. Aplicación del operador AND para la consulta que incluye los dos conceptos que forman la relación semántica. El operador AND realiza la intersección de las líneas que integran los posting de ambos conceptos que forman la relación semántica.

9. Evaluación de resultados obtenidos tanto para los conceptos como para las relaciones. La medida de evaluación que se utiliza en este caso es la de precisión.

$$P_C = \frac{Conceptos\ recuperados}{Total\ conceptos} \tag{1}$$

$$P_R = \frac{Relaciones\ recuperadas}{Total\ relaciones} \tag{2}$$

Donde: *Conceptos recuperados* es el total de conceptos obtenidos por el SRIB, y el *Total conceptos* es el total de conceptos existentes en la ontología de dominio. En el caso de *Relaciones recuperadas* se evalúa por separado las relaciones tipo class-inclusion y las relaciones no taxonómicas (para más información ver [17]). El *Total relaciones* corresponden a las relaciones de cada tipo recuperadas de la ontología de dominio evaluadas de manera independiente.

La Figura 1 muestra el comportamiento de manera gráfica de este algoritmo.

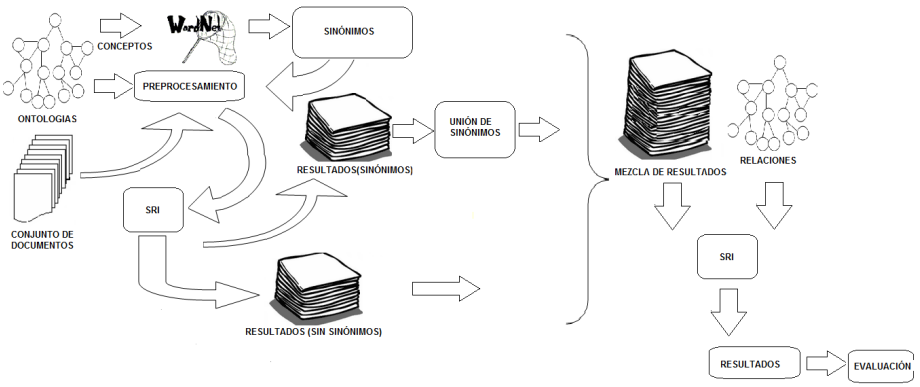


Fig. 1. Primera aproximación para la expansión de consultas en un SRIB.

5. Resultados experimentales

En esta sección, se presentan los datos utilizados (5.1) y los resultados obtenidos en los experimentos (5.2).

5.1. Conjunto de datos

En la Tabla 1 se presenta el número de conceptos (C), el total de relaciones class-inclusion (CI) y el total de relaciones no taxonómicas (NT) de las ontologías evaluadas. También se incluye el número de documentos (D), número de tokens

(T), cantidad de vocabulario (V), y el número de oraciones. Los dominios utilizados en los experimentos son Inteligencia Artificial (IA), Aprendizaje e-Learning (SCORM) [19], ontología del dominio de Petróleo (OIL), y Turismo (Turismo).

Tabla 1. Conjunto de datos.

Dominio	Ontología			Corpus de referencia				
	C	SC	NT	D	T	V	O	S
AI	276	205	61	8	11,370	1,510	475	415
SCORM	1,461	1,038	759	36	1,621	34,497	1,325	1,606
OIL	48	37	-	577	546,118	10,290,107	168,554	157,276
Turismo	963	1,016	-	1,801	877,519	32,931	36,505	31,418

5.2. Resultados obtenidos

A continuación se presentan los resultados experimentales obtenidos por los dos algoritmos desarrollados y su comparación, es decir, resultados del Sistema de Recuperación de Información Booleno (SRIB) sin expansión de consultas y del Sistema de Recuperación Información Booleano (SRIB) con expansión de consultas.

Los resultados obtenidos por ambos algoritmos, para el caso de los conceptos, se muestran en la Tabla 2 para cada ontología revisada (Dominio). En la Tabla 2 también se muestra el total de conceptos extraídos de la ontología (CO), los conceptos recuperados por el SRIB sin expansión (C), los conceptos que no obtuvieron líneas asociadas (F) y la precisión (P); los conceptos recuperados por el SRIB con expansión (CA), los conceptos que no logró recuperar el SRIB con expansión (FA) y la precisión obtenida (PA).

Además, en la tabla se incluye la cantidad de oraciones obtenidas por el SRIB sin expandir (OC), con expansión (OCA), la diferencia del número de líneas recuperadas con expansión y sin ella (OCE) y el porcentaje de incremento (%). En base a los resultados obtenidos para los conceptos, se observa que en los casos de los dominios de SCORM y Turismo principalmente, se incrementó el número de conceptos recuperados que los que se recuperan con el SRIB sin expansión. Además, la cantidad de oraciones que contienen los sinónimos del concepto incrementa la cantidad de líneas u oraciones asociadas a cada concepto de las ontologías, esto ocurre para cada dominio. El porcentaje de incremento de la información recuperada por el SRIB con expansión es mayor al 27 %, lo que indica que el concepto puede ser representado en el corpus por su sinónimo correspondiente y que esta información es adicional a la presentada por el SRIB sin expanción.

En la Tabla 3 se presentan los resultados obtenidos por ambos Sistemas de Recuperación de Información con expansión y sin ella, para relaciones de tipo class-inclusion de cada ontología de dominio. La columna OSC corresponde al

Tabla 2. Resultados del Sistema de Recuperación Booleano con expansión para el caso de los conceptos de cada ontología de dominio.

Dominio	Ontología							SRI			
	CO	C	F	P	CA	FA	PA	OC	OCA	OCE	%
IA	276	274	2	0.992	274	2	0.992	1,992	3,110	1,118	56.12 %
SCORM	1,461	1,443	18	0.987	1,444	17	0.988 %	23,479	31,833	8,354	35.58 %
OIL	48	48	0	1.00	48	0	1.00	232,603	297,234	64,631	27.78 %
Turismo	963	683	280	0.709	711	252	0.736	86,077	232,855	146,778	170.51 %

total de relaciones tipo class-inclusion incluidas en la ontología de dominio correspondiente. La columna SC es el total de conceptos recuperados con información del SRI sin expansión. La columna correspondiente a F es la diferencia de las relaciones recuperadas por el SRI booleano sin expansión y con expansión (FA). La precisión del sistema sin expansión (P) y con expansión (PA). También se incluye la cantidad de oraciones recuperadas en total por el SRIB sin expansión (OSC) y con expansión ($OSCA$) para este tipo de relaciones, la diferencia obtenida (OE) y el porcentaje de la diferencia (%). En base a los resultados obtenidos se observa que el número de relaciones de tipo class-inclusion de las tres primeras ontologías se mantienen por los dos algoritmos diseñados, pero en el caso de la ontología de Turismo el número de conceptos se incrementa de 292 a 387 esto indica que existen conceptos en el corpus que sólo se pueden encontrar por su correspondiente sinónimo y al SRIB sin expansión no le es posible encontrarlo exactamente. También, la cantidad de oraciones asociadas a los SRIB con expansión se incrementa para las cuatro ontologías y más aún para la ontología de Turismo, reforzando nuevamente la existencia de los sinónimos de los conceptos encontrados en el corpus.

Tabla 3. Resultados del Sistema de Recuperación Booleano con expansión para el caso de las relaciones tipo class-inclusion de cada ontología de dominio.

Dominio	Ontología							SRI			
	OSC	SC	F	P	SCA	FA	PA	OSC	OSCA	OE	%
IA	205	205	0	1.00	205	0	1.00	782	824	42	5.37
SCORM	1,038	1,006	32	0.969	1,006	32	0.969	10,624	10,784	160	1.50
OIL	37	32	5	0.864	32	5	0.864	12,691	12,699	8	0.063
Turismo	1,016	292	724	0.287	387	629	0.380	4,886	19,520	14,634	299.5

En el caso de las relaciones tipo no taxonómicas, que sólo las ontologías IA y SCORM tienen, se observa que la cantidad de relaciones recuperadas es la misma para ambos sistemas. Sólo se incrementaron algunas oraciones en las cuales existen el sinónimo correspondiente a cada concepto que forma la relación (ver Tabla 4).

Tabla 4. Relaciones no taxonómicas.

Dominio	Ontología							SRI			
	ONT	NT	F	P	NTA	FA	PA	ONT	ONTA	OE	%
IA	61	61	0	1.000	61	0	1.000	106	121	15	14.15 %
SCORM	759	744	15	0.980	744	15	0.980	8,752	9,589	837	9.56 %

5.3. Análisis de resultados

La aproximación propuesta, sistema de recuperación booleano con expansión semántica por sinónimos, recupera más información que lo que se obtiene con el sistema de recuperación booleano tradicional (ver columna % de cada tabla). La necesidad de incorporar sinónimos en la expansión se debe a que estos son considerados en una de las etapas de diseño de ontologías y el SRIB tradicional no logra identificar los conceptos exactos en el corpus, pero en base a los resultados se observa que el sinónimo correspondiente mantiene una relación semántica con evidencia en el corpus, dando la posibilidad de encontrar más relaciones existentes en la ontología y en el corpus de dominio.

Una de las limitaciones que se identifica en la aproximación es que el recurso semántico (WordNet) no es heterogeneo, es decir, no se obtienen sinónimos para cualquier tipo de dominio. Por lo tanto, se considera el uso de otras alternativas para la extración de sinónimos en el corpus, como es el caso del uso de patrones léxico-sintácticos.

6. Conclusiones

En este artículo se presenta una aproximación que realiza la expansión de consultas con el uso de sinónimos. Las consultas están formadas por los conceptos extraídos de las ontologías de dominio, la aproximación propuesta utiliza un SRIB. En base a los resultados experimentales se observa que la expansión permite recuperar más información del corpus de dominio. En algunos casos el SRIB con expansión permite recuperar más conceptos e información asociada a estos conceptos desde el corpus, al añadir los sinónimos correspondientes obtenidos desde WordNet. En algunas ontologías la cantidad de oraciones recuperadas supera significativamente al SRIB sin expansión. Como trabajo a futuro se propone el diseño de otro algoritmo de expansión que considere el uso de sinónimos por cada palabra que integra al concepto. Se considera que esa propuesta facilitará la incorporación de más información a procesar por cada concepto. También como consecuencia de este tipo de expansión consideramos la propuesta de extensión de las ontologías de dominio al incluir la relación semántica de tipo sinonimía.

Referencias

1. Baeza-Yates, R., Ribeiro-Neto, B., et al.: Modern information retrieval, vol. 463. ACM press New York (1999)

2. Chauhan, R., Goudar, R., Rathore, R., Singh, P., Rao, S.: Ontology based automatic query expansion for semantic information retrieval in sports domain. In: Eco-friendly Computing and Communication Systems, pp. 422–433. Springer (2012)
3. Cotelo, S., Makowski, A., Chiruzzo, L., Wonsever, D.: Búsqueda de documentos utilizando criterios semánticos (2012)
4. Gil, R.M.n., Aparicio, F., de Buenaga, M.: Sistema de acceso a la información basado en conceptos utilizando freebase en español-inglés sobre el dominio médico y turístico. Procesamiento del lenguaje natural 49, 29–38 (2012)
5. Hany, M.H., Khaled, M.F., Nagdy, M.N.: Recuperación semántica enfocada en documentos web. International Journal of Advanced Computer Science and Applications (2011)
6. Hernández-Aranda, D., Granados, R., García-Serrano, A.: Servicios de anotación y búsqueda para corpus multimedia. Procesamiento del Lenguaje Natural 49, 213–216 (2012)
7. Korfhage, R.R.: Information storage and retrieval (2008)
8. Kuna, H.D., Rey, M., Podkowa, L., Martini, E., Solonezen, L.: Expansión de consultas basada en ontologías para un sistema de recuperación de información. In: XVI Workshop de Investigadores en Ciencias de la Computación (2014)
9. Miller, G.A.: WordNet: a lexical database for English. Communications of the ACM 38(11), 39–41 (1995)
10. Neha, S., others: Mejora de la consulta con coeficiente de czekanowski por expansión usando algoritmos genéticos. International Journal of Computer Science and Information Technologies (2014)
11. Porter, M.F.: Readings in information retrieval. chap. An Algorithm for Suffix Stripping, pp. 313–316. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1997)
12. Salton, G., McGill, M.J.: Introduction to modern information retrieval (1986)
13. Schneider, J.M., Declerck, T., Fernández, J.L.M., Martínez, P.: Prueba de concepto de expansión de consultas basada en ontologías de dominio financiero. Procesamiento del lenguaje natural 51, 109–116 (2013)
14. Shabanzadeh, M., Nematbakhsh, M.A., Nematbakhsh, N.: A semantic based query expansion to search. In: 2010 International Conference on Intelligent Control and Information Processing (ICICIP). pp. 523–528. IEEE (2010)
15. Tolosa, G.H., Bordignon, F.R.: Introducción a la recuperación de información (2008)
16. Tovar Vidal, M.: Evaluación automática de ontologías de dominio restringido. Ph.D. thesis, Cenidet (2015)
17. Tovar Vidal, M., Pinto Avendaño, D., Montes Rendón, A., González Serna, J.G., Vilariño Ayala, D.: Evaluation of ontological relations in corpora of restricted domain. Computación y Sistemas 19(1) (2015)
18. Valbuena, S.J., Londoño, J.M.: Búsqueda de documentos basada en el uso de índices ontológicos creados con mapreduce document search supported on an ontological indexing system created with mapreduce. Ciencia e Ingeniería Neogranadina 24(2), 57 (2014)
19. Zouaq, A., Gasevic, D., Hatala, M.: Linguistic patterns for information extraction in ontocmaps. In: Blomqvist, E., Gangemi, A., Hammar, K., del Carmen Suárez-Figueroa, M. (eds.) WOP. CEUR Workshop Proceedings, vol. 929. CEUR-WS.org (2012)

Análisis sobre el idioma español en México, con base en la frecuencia de palabras azules, rojas, obscenas y vulgares en Twitter

Orlando Ramos, Luis Alfredo Moctezuma, Jesús García,
David Pinto, Rodolfo Martínez

Benemérita Universidad Autónoma de Puebla, Puebla
México

{orlandxrf, luisalfredomoctezuma}@gmail.com
gr_jesus@outlook.com, dpinto@cs.buap.mx

Resumen. En este artículo se presenta una comparativa entre estados de la República Mexicana de la frecuencia de palabras azules, rojas, obscenas y vulgares que escriben usuarios de la red social de microblogging Twitter. Se presentan gráficas de los resultados obtenidos. El objetivo es mostrar en mapas del comportamiento de la frecuencia de palabras por cada estado y clasificados por el tipo de palabra analizada. Los experimentos fueron realizados sobre un corpus de tweets.

Palabras clave: Palabras azules, palabras rojas, palabras obscenas, palabras vulgares, tweets.

1. Introducción

Una de las características que definen a México sobre el idioma español es su riqueza lingüística en relación a palabras con connotación sexual y ofensivas que utilizan las personas en la vida cotidiana, muchas veces en doble sentido. Estas expresiones verbales o escritas son consideradas por la sociedad como malas palabras. En cuanto a las palabras azules se utilizan en un contexto positivo, mientras que las palabras rojas son utilizadas en un contexto negativo.

El procesamiento del lenguaje natural en la red social Twitter nos da conocimiento de la dispersión del idioma español a lo largo de la república Mexicana. Al tener los datos se puede buscar información específica sobre temas de interés, en este caso la búsqueda de palabras rojas, azules, obscenas y vulgares. El objetivo de este trabajo es identificar los estados de la República Mexicana en los que se concentra una mayor frecuencia en el uso de ellas y mostrarlas de una manera más clara para el lector.

2. Trabajo relacionado

En esta sección se describen trabajos previos que tratan sobre el análisis de frecuencias de determinadas palabras por ubicación geográfica divididas por entidades federativas.

En [1] se realizó un análisis de la dispersión del idioma español en México con respecto a la frecuencia n-gramas de letras que fueron calculados por regiones geográficas de la República Mexicana y comparados con la media nacional para obtener la frecuencia por entidad federativa, donde se usaron corpus de noticias periodísticas y de tweets. En general, se calcula la frecuencia de cada n-grama (unigrama, bigrama o trigrama) y se ordenan los n-gramas en forma descendente. Se usa una porción de los n-gramas más frecuentes y se calcula el grado de intersección entre los n-gramas calculados a nivel nacional y aquellos calculados a nivel estatal. El grado de traslape indica la cercanía del idioma en el estado con respecto a la media nacional.

En [2] se utilizó un corpus de tweets para clasificar la frecuencia de palabras obscenas y vulgares por entidad federativa de la República Mexicana, se realizó un análisis estadístico sobre los tweets con base en los diccionarios de vulgaridades y obscenidades se obtuvieron las frecuencias. Se realizó un proceso de balanceo quedando la misma cantidad de tweets correspondiente a la clase menos representativa en dicho conjunto. Una vez desarrollado el corpus de entrenamiento, se utilizaron para el desarrollo del modelo de clasificación todas las palabras que aparecen en cada tweet. No se aplicó ningún preprocesamiento al corpus de entrenamiento construido.

En [5] se realizó una propuesta para el estudio de campos semánticos en Twitter, este trabajo se plantea con el fin de dar paso a una serie de investigaciones donde se pueda usar la técnica para extracción de tweets. En este trabajo se usan bases de datos para almacenar los tweets que son extraídos en formado JSON usando la API original de twitter, solo almacenando los datos, el uso que se les pudiera dar es dejado para posibles trabajos futuros. La extracción masiva de tweets no es una opción en este trabajo ya que solo se trabajan sobre perfiles preestablecidos y tomados aleatoriamente.

Se utilizó [3] para complementar los diccionarios de palabras obscenas y vulgares usados en [1], sin embargo, algunas frases que aparecen ya no son utilizadas de manera frecuente en la actualidad por usuarios de la red social de Twitter. En este trabajo se hizo uso de herramientas que permitieron la extracción masiva de tweets y poder analizar una mayor cantidad de perfiles y poder analizar información específica y de relevancia en el procesamiento de lenguaje natural. La contribución más fuerte presentada en este trabajo es que usamos técnicas para tener una representación de las palabras que más se usan por estado de la república mexicana, con esta información podemos ver las tendencias en el uso de ciertas palabras por estado pudiendo comparar con factores ocurridos en cada estado, por ejemplo al analizar un estado con problemas de delincuencia, ver que palabras son las más usadas y proponer una representación de que palabras usan regularmente los delincuentes o gente que convive con ellos.

3. Procesamiento de lenguaje natural

La red social de microblogging Twitter ofrece una API para desarrolladores, que consiste en un conjunto de métodos para poder hacer uso de los datos públicos, en este caso se obtuvieron tweets, que es un mensaje de 140 caracteres compartido públicamente con una comunidad.

A continuación se describe cada tipo de palabras analizadas en datos obtenidos.

- Palabra azul: se refiere a la palabra que expresan en el autor un sentimiento positivo, por ejemplo; alegrar, gozar, respetar, ganar, cuidar etc.
- Palabra roja: este tipo de palabra manifiesta un sentimiento negativo en el autor, por ejemplo; enojar, llorar, juzgar, burlar, gobernar, etc.
- Palabra obscena: se usa generalmente para calificar cierto tipo de lenguaje, sobre todo a las palabras que tienen connotación sexual, por ejemplo; mocos, chichis, culo, huevos, etc.
- Palabra vulgar: hace referencia a palabras de carácter ofensivo, conocidas como palabras altisonantes, por lo regular vistas por la sociedad como malas palabras, por ejemplo; puto, chingada, cabrón, pendejo, etc.

El corpus construido para este experimento consiste de tweets, extraídos por medio de la API mencionada anteriormente, con base en las coordenadas geográficas de cada uno de los estados de la república mexicana, en la Tabla 3.1 se muestra la cantidad de tweets que se recolectaron para este trabajo. Cabe mencionar que la cantidad de tweets depende del estado de la República Mexicana, en algunos estados los tweets que se pueden extraer son menores que en otros, es por ello que se ha tomado al estado con un menor número de tweets y se han recortado los demás estados para que todos tengan la misma cantidad de tweets.

Tabla 3.1. Corpus de Tweets.

Para cada uno de los 32 estados se utilizaron	
Tweets	460
Total	14720

En la Tabla 3.2 se muestran los cuatro diccionarios que se utilizaron para calcular la frecuencia de los diferentes tipos de palabras. Los diccionarios de palabras obscenas y vulgares, se construyeron en base a una encuesta a 20 personas de entre 23 y 34 años, a quienes se les pidió que escribieran las palabras que usaban con más frecuencia. Esta decisión fue tomada porque los diccionarios con los que se contaba sobre palabras obscenas y vulgares estaban desactualizados (pasados de moda), por este motivo se decidió actualizarlos.

Tabla 3.2. Diccionarios utilizados.

Palabras			
Obscenas	**Vulgares**	**Azules**	**Rojas**
158	104	78	94

4. Experimentos

Se realizó una comparativa de los datos guardados en el corpus y los diccionarios descrito en la sección 2, dicha comparativa se realizó contando la frecuencia con la que aparecieron las palabras de los diccionarios en el corpus de cada estado.

4.1 Preprocesamiento del corpus

Para poder comparar las palabras de los diccionarios con los tweets que se extrajeron se realizó un tratamiento a dichos tweets que consistió en eliminación de caracteres especiales, signos de puntuación, cada letra en mayúscula se sustituyó por su correspondiente en minúscula, sustitución de retornos de carro y nueva línea para el reacomodo de los tweets, dejando cada uno en un renglón, ya que la API Twitter proporciona los datos tal como los ingreso el usuario.

Para la comparativa con las palabras vulgares y obscenas se realizó la comparativa de dichas palabras tal y como aparecen en los diccionarios, sin embargo, para las palabras azules y rojas se lematizó el corpus de cada estado, así como los diccionarios correspondientes usando TreeTagger.

4.2 Análisis de palabras

De los diccionarios de palabras se comparó con el corpus de cada estado. El resultado fue un archivo con las palabras encontradas y su frecuencia, lo cual permite hacer una comparativa entre todos los estados.

En la Tabla 4.2.1 se muestra el resultado obtenido de analizar los corpus de palabras obscenas, vulgares, azules y rojas explicadas anteriormente. Para poder comparar los resultados obtenidos entre todos los estados lo que se hizo fue obtener un 100% del total de palabras vulgares encontradas y buscar cuantas palabras del total se encontraron en cada estado. Esto nos permite decir que del total de palabras usadas en todos los estados, las obscenas se usaron en un 0.93% en Aguascalientes y el 0.40% se usaron en Baja california Norte por ejemplo. De igual manera podemos ver que el porcentaje de palabras vulgares para cada estado representa la cantidad de veces que se usaron respecto de los otros estados.

Tabla 4.2.1. Frecuencias encontradas por estados.

Estado	Palabras			
	Obscenas	**Vulgares**	**Azules**	**Rojas**
Aguascalientes	0.93%	1.29%	2.68%	1.42%
Baja California Norte	0.40%	0.89%	1.41%	2.28%
Baja California Sur	1.46%	0.98%	1.09%	1.11%
Campeche	0.67%	1.25%	2.01%	1.92%
Chiapas	2.66%	2.46%	2.42%	0.81%
Chihuahua	0.80%	1.34%	1.30%	1.32%
Coahuila	3.06%	6.03%	8.73%	12.96%
Colima	0.40%	0.58%	1.15%	0.66%
Distrito Federal	7.06%	8.17%	6.43%	6.28%
Durango	1.20%	1.38%	1.97%	1.82%
Guanajuato	2.80%	3.30%	3.05%	1.21%
Guerrero	1.73%	1.47%	1.04%	1.92%
Hidalgo	1.20%	1.92%	1.41%	1.47%

Estado	Palabras			
	Obscenas	**Vulgares**	**Azules**	**Rojas**
Jalisco	8.39%	5.58%	6.92%	6.88%
México	16.38%	9.73%	6.37%	6.02%
Michoacán	0.93%	0.94%	1.45%	1.47%
Morelos	0.67%	0.98%	1.45%	1.82%
Nayarit	0.00%	0.13%	0.62%	0.40%
Nuevo León	0.13%	0.22%	2.14%	0.05%
Oaxaca	0.40%	0.40%	1.15%	0.71%
Puebla	2.26%	1.34%	1.03%	1.87%
Querétaro	3.60%	5.94%	4.48%	4.05%
Quintana Roo	2.80%	2.63%	3.37%	3.24%
San Luis Potosí	2.53%	2.81%	2.50%	4.05%
Sinaloa	3.46%	3.79%	3.88%	2.28%
Sonora	10.65%	11.43%	6.43%	6.02%
Tabasco	7.19%	5.04%	5.34%	6.88%
Tamaulipas	2.53%	3.93%	4.07%	4.00%
Tlaxcala	1.73%	1.29%	1.09%	1.57%
Veracruz	5.19%	5.31%	5.82%	5.21%
Yucatán	5.59%	6.16%	6.17%	6.63%
Zacatecas	1.20%	1.25%	1.01%	1.67%

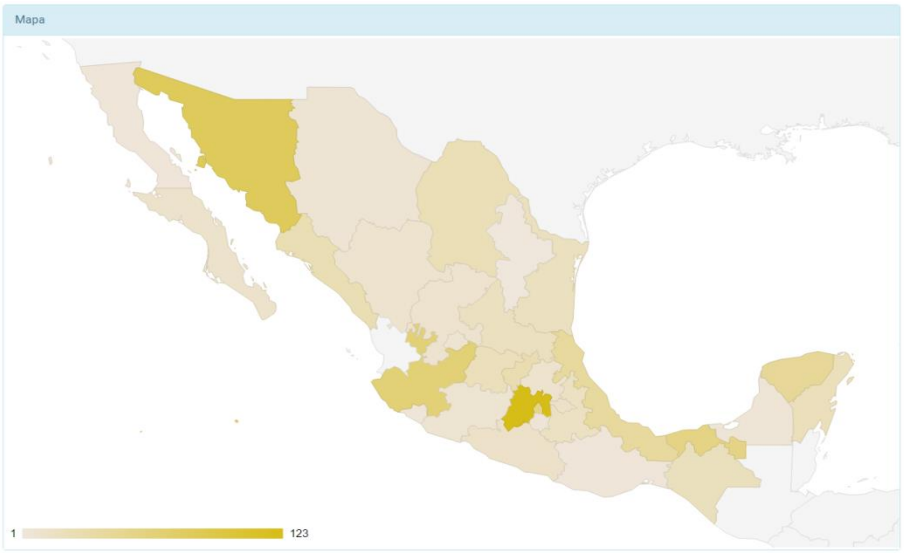


Fig. 1. Mapa para las palabras obscenas.

5. Resultados

Para una mejor comprensión para el lector se muestran los resultados de manera gráfica, en mapas interactivos en los que se aprecian las entidades federativas con la frecuencia con la que se obtuvieron los distintos tipos de palabras. En los mapas se muestra de color más intenso a los estados con una mayor frecuencia, y de color más tenue los de menor frecuencia.

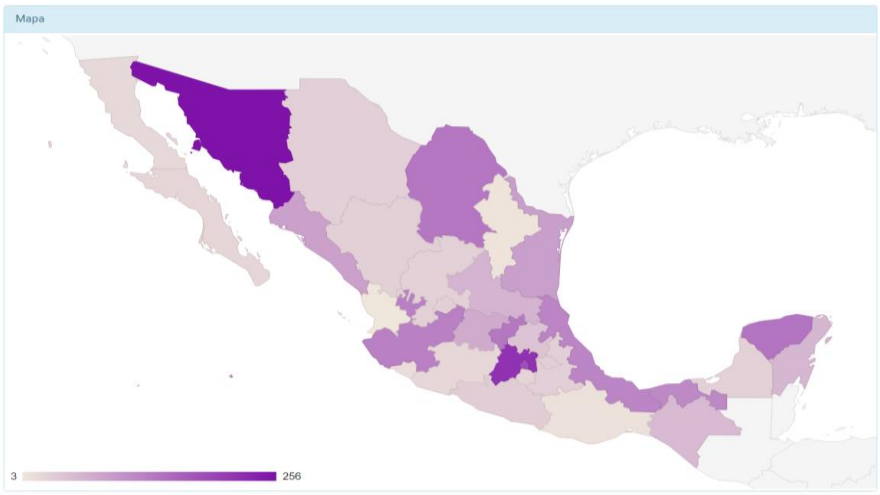


Fig. 2. Mapa para las palabras vulgares.

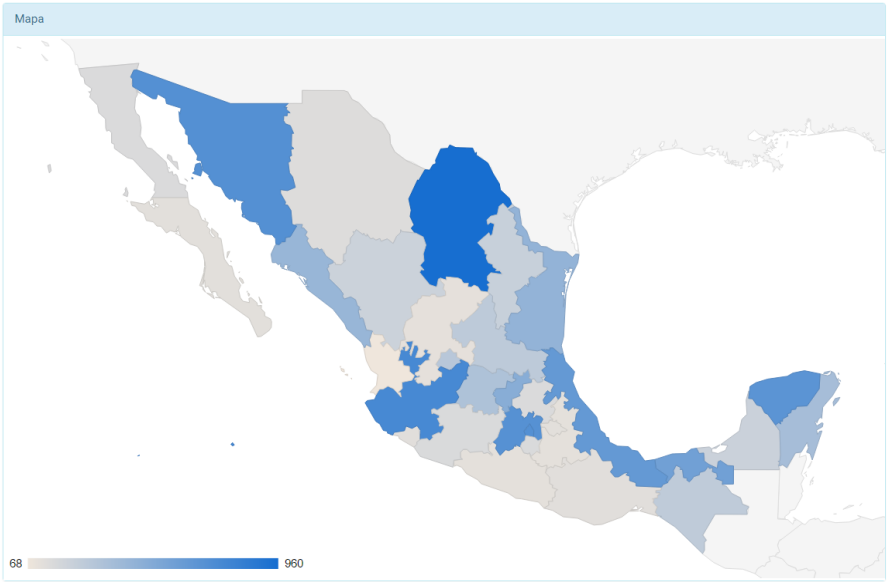


Fig. 3. Mapa para palabras azules.

En la Fig. 1 se aprecia el mapa con la frecuencia obtenida de analizar el uso de las palabras obscenas por cada estado de la República. Los estados de Sonora, Estado de México y Jalisco presentan una mayor frecuencia en el uso de palabras obscenas, mientras que el estado de Nayarit es el único que no presenta incidencias de dichas palabras.

Como se puede observar en la Fig. 2 los estados en los que se encontró un mayor número de incidencias de palabras vulgares es Sonora junto con el Estado de México y los que menos frecuencia obtuvieron fueron los estados de Nayarit, Nuevo León y Oaxaca.

Para la Fig. 3 se muestra la frecuencia de palabras azules en donde se observa claramente que los estados de Coahuila y Jalisco presentan un mayor número de palabras positivas con base en el análisis realizado, mientras que los estados de Nayarit y Zacatecas fueron los que se encontraron dichas palabras con menos frecuencia.

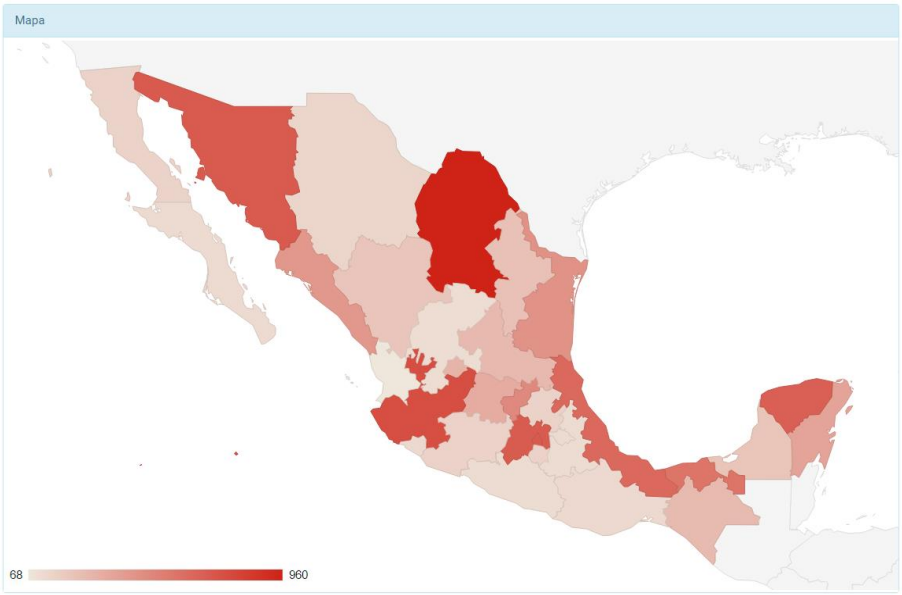


Fig. 4. Mapa para las palabras rojas.

En la Fig. 4 se presentan Coahuila, Sonora y Jalisco como los estados con un mayor número de frecuencias de palabras negativas. Y los estados con menos incidencia de palabras rojas son los estados de Nayarit, Zacatecas y Guerrero.

6. Conclusiones

Los estados de la República Mexicana con una presencia mayor de los cuatro tipos de palabras usados en este trabajo son Sonora, Coahuila y el Estado de México, lo que nos permite concluir que en el norte y centro del país es donde se concentra la mayor frecuencia de palabras azules, rojas, obscenas y vulgares.

Las palabras más utilizadas de los diferentes tipos son:

- Azules: *hacer*, *decidir*, *ganar*.
- Rojas: *ultimar*, *jugar*, *fallar*.
- Obscenas: *verga*, *hueva*, *huevos*.
- Vulgares: *pinche*, *pedo*, *pendejo*.

Como trabajo futuro se pretende incrementar los diccionarios de los cuatro tipos de palabras, así como del corpus de tweets obtenido con base en las mismas coordenadas utilizadas en el presente trabajo, se propone analizar diferentes redes sociales como Facebook, para así comparar la red social en la que se prefiere utilizar los diferentes tipos de palabras.

Referencias

1. Ramos, O., Pinto, D., Priego, B., Olmos, I., Beltrán, B.: Análisis empírico de la dispersión del español mexicano. Research in Computing Science (2014)
2. Guzmán, E., Beltrán, B., Tovar, M., Vázquez, A., Martínez, R.: Clasificación de frases obscenas o vulgares dentro de tweets. Research in Computing Science, Vol. 85, pp. 65–74 (2014)
3. d. l. L. Academia Mexicana: Diccionario de mexicanismos. Siglo XXI Editores México (2010)
4. Gupta, N.K.: Extracting Phrases Describing Problems with Products and Services from Twitter Messages. Computación y Sistemas, Vol. 17, No. 2, pp. 197–206 (2013)
5. Alonso Berroca, J.L.: Propuesta de estudio del campo semántico de los libros electrónicos en Twitter (2012)
6. Fainholc, B.: Un análisis contemporáneo del Twitter. RED – Revista de Educación a Distancia
7. Pla, F., Hurtado, Ll-F.: Análisis de Sentimientos en Twitter. In: Proceedings of the TASS workshop at SEPLN (2013)
8. Alegria, I.: Introducción a la tarea compartida Tweet-Norm 2013: Normalización léxica de tuits en español (2013)

Desarrollo de un modelo para encontrar la similitud semántica multilingüe

Emanuel Aguilar, Darnes Vilariño, Claudia Zepeda,
Mireya Tovar, Beatriz Beltrán

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Puebla,
México

{emanuel.aguilar.benitez}@hotmail.com, {dvilarinoayala, czepedac}@gmail.com,
{mtovar, bbeltran}@cs.buap.mx

Resumen. En el presente trabajo se desarrollan dos modelos para detectar el grado de similitud semántica entre pares de sentencias. El primer modelo está basado en aprendizaje supervisado, este utiliza un vector compuesto por dieciséis características para la representación de cada par de sentencias, con el que se entrena un clasificador. El segundo es un modelo no supervisado, el cual, basa su funcionamiento en la reconstrucción de una de las sentencias por medio de la otra, apoyándose para esto en los sinónimos de las palabras que las componen. Los dos modelos fueron probados para los idiomas inglés y español, y presentan un desempeño aceptable para ambos idiomas.

Palabras clave: Similitud semántica multilingüe, grafo de co-ocurrencia, intertextualidad, reconstrucción de sentencias.

1. Introducción

La similitud semántica tiene como objetivo determinar que tan semejantes son los sentidos de dos textos, y es por esto que se ha convertido en objeto de estudio durante muchos años dentro del área del Procesamiento de Lenguaje Natural (PLN). La similitud semántica cuenta con un amplio rango de aplicaciones, por ejemplo: máquinas de traducción, construcción automática de resúmenes, atribución de autoría, pruebas de lectura comprensivas, recuperación de información y muchas otras que necesitan medir el grado de similitud entre dos textos dados.

Existen diversos sistemas para la detección de similitud semántica, pero la mayoría sólo se enfocan en el idioma inglés. Por esta razón, se pretende el desarrollo de un modelo capaz de encontrar el grado de similitud semántica entre pares de sentencias que logre un buen desempeño en los idiomas inglés y español. El objetivo del modelo es dado un par de sentencias S_1 y S_2, ofrecer un valor entre 0 y 5 que represente el grado de similitud semántica entre ambas sentencias considerando los siguientes valores:

– 5 Las dos sentencias son completamente equivalentes, ya que significan lo mismo.

- 4 Las dos sentencias son en su mayoría equivalentes, pero difieren algunos detalles sin importancia.
- 3 Las dos sentencias son más o menos equivalentes, pero alguna información importante difiere o está ausente en una de las sentencias.
- 2 Las dos sentencias no son equivalentes, pero comparten algunos detalles.
- 1 Las dos sentencias no son equivalentes, pero tratan sobre el mismo tema.
- 0 Las dos sentencias tratan sobre temas distintos.

En el presente trabajo se presentan dos modelos que permiten detectar la similitud semántica entre pares de sentencias. Este artículo se encuentra estructurado de la siguiente manera: en la sección 2 se exponen algunos trabajos relacionados a este tema, en la sección 3 se presentan los modelos desarrollados para resolver la problemática que se plantea, posteriormente en la sección 4 se muestran los resultados obtenidos y por último en la sección 5 se abordan las conclusiones de la investigación.

2. Trabajo relacionado

Se han desarrollado diversas investigaciones para detectar el grado de similitud semántica entre pares de sentencias, a continuación se mencionan algunas asociadas a los trabajos presentados en el marco de la Conferencia SemEval 2013[1].
En el trabajo desarrollado en [1], se describe un sistema en el cual para estimar la similitud semántica entre dos enunciados se usan modelos de regresión que incluyen las siguientes características: n-gramas repetidos entre enunciados, similitud léxico-semántica entre palabras distintas, métricas de similitud de cadenas, similitud de contenido afectivo y longitud del enunciado. Todas estas características se combinan usando un modelo, ya sea de regresión múltiple lineal, o bien un modelo que detecta la relación que en cuanto a longitud poseen cada una de las sentencias. Dicha propuesta obtiene un 47% de aciertos reportados con los datos del SemEval 2013.

En el trabajo presentado en [2], se presenta un modelo de estimación lineal usando regresión *ridge*. Se analiza el comportamiento de la propuesta aplicando validación cruzada de 5 pliegues con los datos de entrenamiento. Esto permite afinar la sanción α que se emplea en la regresión *ridge*, con $\alpha \in 2^{\{-5,-4,\ldots,4\}}$. Dadas dos sentencias de entrada el sistema extrae: características de superposición de n-gramas, características de longitud y características de sentimientos. Posteriormente el sistema estandariza los valores de las características, substrayendo la media de las características y dividiendo por su desviación estándar. Se usan adaptaciones al dominio para facilitar la generalización a nuevos dominios. Este sistema presenta en promedio una tasa de aciertos del 45.03%.

El sistema que se describe en [3] desambigua el sentido de cada palabra en el enunciado empleando el contexto de cada una de ellas. La similitud de enunciados es calculada con el número de sentidos que comparten, es razonable asumir que enunciados similares deben tener más sentidos superpuestos. Para determinar la superposición del sentido se comparan las características de las palabras, primero se conceptualizan las palabras y después se calcula su similitud basándose en una estructura jerárquica, que a su vez se basa en WordNet; cada palabra en un enunciado

[1] http://www.cs.york.ac.uk/semeval-2013/index.php?id=tasks

es asignada a un sentido en el diccionario WordNet. Para reducir las limitaciones del diccionario, se utilizan los términos de los sentidos relacionados con la palabra objetivo, se asume que las palabras que co-ocurren en un enunciado comparten relación del sentido, y mientras más similares sean los enunciados compartirán más términos en las definiciones de sus palabras. Por ello no sólo se extraen los términos del enunciado principal, sino que también, para cada palabra del enunciado principal, se extraen los términos de su *hypernym, hyponym, meronym, holonym y troponym*; formando con esto el conjunto contexto. Finalmente se compara el contexto del enunciado con diferentes conjuntos de contextos para determinar cuál sentido debe ser asignado a las palabras. La implementación de este sistema reporta un 41.51% de aciertos.

Otro trabajo que es importante destacar es el propuesto en [4], que procesa características que se obtienen de diferentes bases de conocimiento como son WordNet, Wikipedia y Wiktionary. Las puntuaciones de similitud derivadas de estas características son introducidas dentro de varios perceptrones multicapa. Dependiendo del tamaño de los textos a comparar se usan diferentes parámetros para las redes neuronales; esto es que para cada grupo de longitud de enunciados, los pesos del perceptrón multicapa se calcularon separadamente. Cada perceptrón fue definido con 48 capas de entrada, correspondientes a las puntuaciones extraídas de las características, 4 capas ocultas y una capa de salida que representa la puntuación de similitud entre los enunciados. Además de las características que se obtienen de las bases de conocimiento aplicaron características de números y expresiones financieras y características de n-gramas. El sistema se desempeña bastante bien cuando los textos de entrada son cortos y de tamaños semejantes, pero cuando existe una variación considerable de tamaño entre estos, o son textos muy largos el desempeño del sistema cae considerablemente. La media de aciertos en este sistema es de 0.16.

Por último, en el sistema *Textual Similarity based on Lexical-Semantic Features* que se presenta en [5], se usan diferentes tipos de características léxicas y semánticas para entrenar un clasificador de bolsas de palabras que se utiliza para decidir la similitud entre los enunciados. Se implementaron tres variaciones del sistema cada una de estas variaciones utiliza un grupo particular de características. Cada par de enunciados es tokenizado, lematizado y post-etiquetado. Posteriormente varios métodos y algoritmos son aplicados para extraer todas las características necesarias para el sistema de máquina de aprendizaje. La primera variación llamada MultiSemLex, toma en cuenta todas las características extraídas y entrena un modelo con un clasificador de bolsas, la segunda variación llamada MultiLex y la tercera llamada MultiSem usan el mismo clasificador, pero incluyen diferentes características. MultiLex utiliza características extraídas de métricas léxico-semánticas y alineamiento léxico-semántico. Por otra parte, MultiSem, utiliza características extraídas solamente de alineación semántica. El sistema obtuvo un coeficiente de correlación general de 0.61.

Para esta investigación se proponen dos modelos, el primero basado en aprendizaje supervisado y un segundo modelo no supervisado. A continuación se presentan ambos modelos.

3. Metodología propuesta

Para resolver la problemática que se plantea en este artículo se desarrollaron dos modelos. El primer modelo extrae un vector de 16 características con el que se

representa cada par de sentencias, el cual es utilizado para entrenar un clasificador, ya sea máquina de soporte vectorial, Naïve Bayes o perceptrón simple; que posteriormente será empleado para determinar el grado de similitud semántica entre las sentencias. En el caso del segundo modelo, la similitud semántica se obtiene por medio de la reconstrucción de la sentencia de menor longitud, utilizando para esto la sentencia de mayor longitud. Para llevar a cabo los modelos se utilizaron las herramientas Clips Pattern[2], Network X[3], WordNet[4] y OpenThesaurus-es[5]. Ambos modelos se explican a continuación.

3.1. Modelo basado en aprendizaje supervisado

La primera parte de este modelo consiste en un pre-procesamiento de los datos de entrada. Este pre-procesamiento se encarga de sustituir los caracteres que no entran en la codificación ANSI por sus correspondientes, luego las sentencias se convierten a su representación en minúsculas, además se eliminan símbolos, signos de puntuación y las *stopwords* que se muestran en la Tabla 1, por ultimo se eliminan las palabras repetidas de cada sentencia.

Tabla 1. Stopwords a eliminar en el pre-procesamiento.

Idioma	Stopwords
Inglés	the, a, of, in, and, to, or, is, for, on, with, that, by, as, at, from, an, was, are, said, be, has, it, this, its, not, after, us, which, will, have, his, were, but, into, over, who, new, up, two, more, he, some, had, i, also, about, their, something, one, we, no, out, can, man, against, they, you, would, being, all, s, may
Español	de, y, la, en, el, que, a, del, los, un, se, por, es, para, una, con, las, al, o, su, como, más, no, entre, ha, fue, sus, desde, son, este, sobre, está, también, según

En la segunda parte de este modelo se obtienen las características de cada par de sentencias de entrada, con lo que se consigue el *Vector de Características*, que junto con su grado de similitud son utilizados por el clasificador para entrenar y así obtener el modelo utilizado para asignar un grado de similitud a las sentencias de prueba.

Por ser este un modelo de aprendizaje supervisado, la parte mas importante es la obtención de las características. En este modelo se proponen dos variaciones, la primera denominada mínimos y la segunda promedios, dado un par de sentencias S_1 y S_2, se extraen las características que se muestran en la Tabla 2, y con estas se forma el vector que representa a cada par de sentencias. Las primeras tres características (TTR[6], número de palabras en la sentencia y número de palabras clave[7] en la sentencia) son propias de cada sentencia, y sobre estas es calculado el mínimo o el promedio según sea el caso.

[2] http://www.clips.ua.ac.be/pattern
[3] https://networkx.github.io/
[4] https://wordnet.princeton.edu/
[5] http://openoffice-es.sourceforge.net/thesaurus/
[6] Type-Token Ratio, medida de variación léxica utilizada en varios tipos de análisis lingüísticos.
[7] Se consideran palabras clave todas aquellas que no son consideradas *stopwords* por la herramienta Clips Pattern.

Tabla 2. Características usadas para la representación de cada par de sentencias.

Variación Mínimos	**Variación Promedios**
TTR mínimo entre S_1 y S_2	TTR promedio de S_1 y S_2
Número mínimo de palabras en S_1 y S_2	Promedio de palabras en S_1 y S_2
Número mínimo de palabras clave en S_1 y S_2	Promedio de palabras clave en S_1 y S_2
Similitud coseno entre S_1 y S_2	
Similitud euclidiana entre S_1 y S_2	
Porcentaje de palabras clave que comparten S_1 y S_2	
Porcentaje de palabras clave que comparten S_1 y S_2 utilizando como *stemmer* el algoritmo de Porter	
Promedio de la intertextualidad entre S_1 y S_2	
Isomorfismo de las representaciones como grafos de S_1 y S_2(ancho de ventana igual a 1)	
Isomorfismo de las representaciones como grafos de S_1 y S_2(ancho de ventana igual a 2)	
Isomorfismo de las representaciones como grafos de S_1 y S_2(ancho de ventana igual a 1, sólo palabras clave)	
Isomorfismo de las representaciones como grafos de S_1 y S_2(ancho de ventana igual a 2, sólo palabras clave)	
Similitud Levenshtein entre S_1 y S_2	
Similitud Dice entre S_1 y S_2	
Distancia coseno entre las características[8] de S_1 y S_2	
Distancia euclidiana entre las características de S_1 y S_2	

En el caso de las características de isomorfismo el valor que pueden tomar es: 1 si las representaciones como grafo de co-ocurrencia de ambas sentencias son isomorfas, y 0 en caso contrario. La representación de cada sentencia por medio de un grafo de co-ocurrencia se llevó a cabo utilizando el siguiente algoritmo:

1. Se crea un grafo vacío no dirigido.
2. Se obtienen las palabras de la sentencia, o sólo las palabras clave, dependiendo el caso, estas pasarán a ser los nodos dentro del grafo.
3. Para cada palabra que se obtuvo se agrega una arista que una a esta palabra y a las siguientes *n* palabras, donde *n* es el valor del ancho de ventana.
4. Se remueven los auto-ciclos[9].

Como se puede notar, cada palabra distinta en la sentencia se convertirá en un nodo dentro del grafo, así bien los nodos se conectarán con otros nodos si las palabras que representan dichos nodos co-ocurren dentro del valor del ancho de la ventana.

3.2. Modelo basado en reconstrucción

El segundo modelo propuesto se basa en un algoritmo simple que no requiere entrenamiento, en este se determina la similitud semántica de dos sentencias por medio de la intertextualidad, que existe entre la sentencia que posee el menor número de palabras distintas y la reconstrucción de está utilizando la sentencia con mayor número de palabras distintas.

[8] TTR, número de palabras de la sentencia, número de palabras clave de la sentencia.
[9] Se considera auto-ciclo cuando una arista conecta a un nodo consigo mismo.

Este modelo está formado por cuatro etapas, tras las que una vez terminado el proceso se obtendrá un grado de similitud representado por un valor decimal entre cero y uno, donde, un valor igual a cero significa que las sentencias son totalmente distintas y un valor igual a uno que ambas sentencias representan el mismo concepto.

La primera etapa se encarga del pre-procesamiento de los datos de entrada, aquí las sentencias se convierten a su representación en minúsculas, posteriormente se eliminan los símbolos y signos de puntación, además de las palabras repetidas y las *stopwords* correspondientes al idioma de la sentencia (ver Tabla 1). En la segunda etapa se determina cual de las dos sentencias se intentará reconstruir, se propone que dicha sentencia sea la de menor longitud en términos de palabras distintas. La sentencia que se intentará reconstruir se conoce como *sentencia objetivo*.

En la tercera etapa, usando la sentencia de mayor longitud se intentará reconstruir la sentencia objetivo aplicando el siguiente algoritmo:

```
for w in Sm:
      if(w in So):
            Sr=Sr+w
      else:
            if(existeSinonimo(w,So)):
                  Sr=Sr+sinonimo(w,So)
```

donde:

w es una palabra que pertenece a *Sm*.
Sm representa la sentencia de mayor longitud.
So representa la sentencia objetivo.
Sr representa la sentencia resultado de la reconstrucción.
La función *existeSinonimo()* determina si está presente un sinónimo de *w* en *So*.
La función *sinonimo()* devuelve el sinónimo de *w* presente en *So*.

Para determinar la sinonimia entre las palabras se usó, WordNet para el idioma inglés, y para el idioma español se realizó una adaptación del OpenThesaurus-es.
Por último, en la cuarta etapa, se utiliza la intertextualidad para medir cuanto de la sentencia objetivo se pudo reconstruir en la sentencia resultado, el cálculo de la intertextualidad se realizó mediante la función *intertextuality()* que proporciona la herramienta Clips Pattern, tomando el resultado de esta operación como el grado de similitud semántica entre el par de sentencias de entrada.

4. Resultados

Durante el desarrollo y prueba de ambos modelos planteados, se han utilizado los corpus proporcionados en la tarea 10 (*Multilingual Semantic Textual Similarity*) de la conferencia SemEval 2014[10]. A continuación se describen los datos empleados durante la presente investigación.

4.1. Conjunto de datos

Se dispone de un corpus conformado por 6,627 pares de sentencias en idioma inglés, el cual se utiliza para el entrenamiento del primer modelo, tanto para el idioma inglés

[10] http://alt.qcri.org/semeval2014/task10/

como español. También se cuenta con dos corpus de prueba, el primero conformado por 3,000 pares de sentencias en idioma inglés y el segundo de 65 pares de sentencias en idioma español.

4.2. Resultados obtenidos

Los resultados obtenidos en las pruebas del modelo basado en aprendizaje supervisado se muestran en la Tabla 3. Cabe aclarar que los clasificadores usados fueron los que ofrece la herramienta Clips Pattern. Como se puede observar, en este modelo el mejor resultado para el idioma inglés se obtuvo por medio de la variación denominada promedios, utilizando como clasificador una máquina de soporte vectorial con un kernel lineal, logrando un porcentaje de 35.16% de aciertos. Por otra parte el mejor desempeño en el idioma español se dio utilizando la variación denominada mínimos junto con una máquina de soporte vectorial usando un kernel lineal obteniendo un 64.61% de aciertos.

Tabla 3. Resultados del modelo basado en aprendizaje supervisado.

Clasificador	Variación	Idioma	Aciertos
SVM kernel lineal	Mínimos	Inglés	34.30%
		Español	64.61%
	Promedios	Inglés	35.16%
		Español	52.30%
SVM kernel polinomial grado 2	Mínimos	Inglés	16.76%
		Español	0%
	Promedios	Inglés	16.76%
		Español	0%
SVM kernel polinomial grado 3	Mínimos	Inglés	16.76%
		Español	0%
	Promedios	Inglés	16.76%
		Español	0%
SVM kernel radial	Mínimos	Inglés	16.76%
		Español	0%
	Promedios	Inglés	16.76%
		Español	0%
Naïve Bayes	Mínimos	Inglés	29.73%
		Español	13.84%
	Promedios	Inglés	29.73%
		Español	13.84%
Perceptrón simple	Mínimos	Inglés	24.70%
		Español	01.53%
	Promedios	Inglés	27.66%
		Español	10.76%

En el caso del modelo basado en reconstrucción, tomando en cuenta que dicho modelo representa la similitud semántica con un valor en el rango [0,1], se ha

multiplicado el valor que retorna por cinco, el cual se comparó con los resultados proporcionados en los corpus de prueba.

Como se observa en la Tabla 4, este modelo presenta una mayor tasa de aciertos con respecto al primer modelo. En este segundo modelo, el porcentaje que se obtiene para el idioma inglés fue de 37.86%, mientras que para el idioma español se obtiene un porcentaje de 70.76%.

Tabla 4. Resultados de precisión para el modelo basado en reconstrucción.

Idioma	Relación de Aciertos	Porcentaje
Inglés	1136/3000	37.86%
Español	46/65	70.76%

5. Conclusión

En este artículo se presentan dos modelos para encontrar la similitud semántica multilingüe. Se observa que el modelo basado en la reconstrucción de las sentencias obtiene un mejor desempeño para ambos idiomas, en contraste con el modelo basado en aprendizaje supervisado. Se cree que los resultados para el primer modelo pudieran mejorar si se construyen representaciones por medio de grafos que incluyan los sinónimos de las palabras que componen a cada sentencia.

Con los resultados obtenidos en el modelo 1 queda claro que las características extraídas no logran descubrir realmente el grado de similitud entre las dos sentencias, pues se basan en la similitud entre las palabras que las componen, sin lograr descubrir el significado de cada una de ellas.

El segundo modelo tiene la ventaja de ser no supervisado, sin embargo, se considera que la diferencia tan grande entre el porcentaje de aciertos se debe a las características del conjunto de datos, pero con los experimentos desarrollados no se puede afirmar que ofrezca mayor precisión.

Se planea extender la reconstrucción de sentencias de manera que no solo busque sinónimos de palabras, si no también sinónimos sobre series de palabras, con lo cual se espera aumentar el grado de precisión.

Referencias

1. Malandrakis, N., Iosif, E., Prokopi, V., Potamianos, A., Narayanan, S.: Lexical, String and Affective Feature Fusion for Sentence-Level Semantic Similarity Estimation. In: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity, Vol. 1, pp. 103–108 (2013)
2. Heilman, M., Madnani, N.: Domain Adaptation and Stacking for Text Similarity. In: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity. Vol. 1, pp. 96–102 (2013)
3. Xu, J., Lu, Q.: Computing Semantic Textual Similarity using Overlapped Senses. In: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity, Vol. 1, pp. 90–95 (2013)
4. Ziak, H., Kern, R.: Semantic Text Similarity by use of Knowledge Bases. In: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity, Vol. 1, pp. 138–142 (2013)

5. Chávez, A., Fernández, A., Dávila, H., Gutiérrez, Y., Collazo, A., Abreu, J., Montoyo, A., Muñoz, R.: Textual Similarity Based on Lexical-Semantic Features. In: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity, Vol. 1, pp. 109–118 (2013)

Hacia la comparación precisa de productos a partir de fuentes de datos distintas en la Web

J. Guadalupe Ramos[1], Ricardo A. Solís[1], Juan Carlos Olivares[2], Luis Alfredo Moctezuma[3], Maya Carrillo[3]

[1] Instituto Tecnológico de La Piedad, La Piedad, Michoacán, México

[2] Instituto Tecnológico de Morelia, Morelia, Michocán, México

[3] Benemérita Universidad Autónoma de Puebla, Puebla, México

jgramos@pricemining.com,solis.itlp@gmail.com, {luisalfredomoctezuma,crrllrzmy}@gmail.com, juancarlosolivares@hotmail.com

Resumen. Cuando un consumidor desea comprar un producto vía comercio electrónico, es común que enfoque sus esfuerzos en localizar la tienda con el mejor precio y con las características concretas de su interés. Desafortunadamente la mayoría de las tiendas emplean un catálogo propio de productos y atributos lo que impide elaborar reportes automáticos de productos de proveedores diversos. En este trabajo hacemos una revisión de la forma en que se ha atendido el problema de la comparación de precios, enseguida caracterizamos algunos sitios de comercio para conocer cómo buscan resolver la comparación. Después presentamos un modelo de organización de información basado en la Web Semántica para posibilitar el diseño de reportes de productos y atributos de diferentes proveedores y detallamos los diferentes roles que pueden asumir los componentes del modelo. La elaboración de reportes precisos puede favorecer la mejor decisión de compra de los clientes e incorporar de manera masiva a negocios diversos a la necesaria comparación de ofertantes.

Palabras clave: Web semántica, ontología, RDF.

1. Introducción

Cuando un cliente desea comprar un producto por medio de una tienda en línea en la Web, regularmente se ve obligado a buscar y evaluar la oferta de diferentes sitios de comercio electrónico. Es común que el interés del cliente se enfoque en un tipo de producto y algunos de sus atributos, por ejemplo el precio.

La acción de evaluar diversos productos no es fácil si consideramos los siguientes aspectos:

- Cada tienda en línea emplea una interfaz distinta, i.e., los productos y propiedades se presentan de manera arbitraria.
- El nombre de los productos y atributos es distinto por cada vendedor, esto es, emplean catálogos con el vocabulario particular de la tienda, por ejemplo podríamos encontrar el producto televisión en un proveedor y el mismo como televisor en otro.
- Los atributos de interés para un producto varían de vendedor a vendedor, por ejemplo para alguien podría ser importante el peso de un producto pero para otro no.
- El procedimiento de búsqueda varía de una tienda a otra, en algunos casos es posible ir refinando la búsqueda, haciendo selección en casillas de verificación, hasta llegar a un producto muy específico. En otros, un producto está asociado a muchas palabras clave. De este modo, como resultado de las consultas aparecen regularmente productos que no interesan al usuario. Si bien, esto constituye un gancho para el consumidor, también es cierto que se debe a la carencia de reportes precisos a partir de la Web.

Expuesto lo anterior, resulta evidente la necesidad de medios tecnológicos que permitan automatizar la tarea de búsqueda y filtrado de información para poder presentar a un cliente un reporte simple con información concreta, por ejemplo, de producto, precio y tienda ofertante. Esto es más urgente si consideramos que la siguiente generación Web estará constituida por agentes que apoyen al ser humano en tareas de localización de datos precisos para la compra de productos y servicios para las personas, conceptualizando así a la Web como un ente inteligente [1] que responde a peticiones concretas.

De hecho existen en la literatura trabajos que se han propuesto establecer modelos para construir herramientas para comparación de precios globales, por ejemplo [11] emplea lenguajes de la Web Semántica (precursores de los estándares actuales) y una interfaz gráfica para lanzar consultas. Por otro lado en [18] se hace un estudio profuso acerca de los problemas a resolver para establecer comparadores de precios globales, en el trabajo se centran fundamentalmente en resolver la heterogeneidad de la información. Para implementar un comparador de precios industrial de diversas fuentes, en línea, de proveedores de productos diversos, sería necesario resolver la heterogeneidad de la tecnología empleada por las empresas para almacenar los datos, además de que habría que compartir un modelo de bases de datos por lo que este enfoque representa grandes retos. Un enfoque en el que los proveedores publiquen información y que existan agentes o intermediarios compartiendo vocabularios semánticos que colecten información y entreguen reportes sería deseable.

En este trabajo presentamos una revisión del estado actual de un conjunto de sitios de comercio electrónico en relación al asunto específico de mostrar un reporte automático de precios de productos de diferentes ofertantes. Para ello, en la Sección 2 definimos e introducimos un conjunto de aspectos a considerar para luego analizar cada uno de los sitios escogidos en relación a dichos aspectos.

Posteriormente en la Sección 3 haremos mención de propuestas tecnológicas que podrían emplearse para filtrar y obtener información de documentos web y

Tabla 1. Caracterización de sitios de comercio electrónico.

Sitio	productos	inventario	filtrado	especificidad
shopping.com	de todo tipo	terceros	detallado	medio
shopzilla.com	de todo tipo	terceros	detallado	medio
google.com/shopping	de todo tipo	terceros	detallado	medio
pricegrabber.com	de todo tipo	terceros	detallado	medio
idealo.co.uk	de todo tipo	terceros	detallado	medio
getprice.com	de todo tipo	terceros	detallado	medio
shopbot.com.au	de todo tipo	terceros	general	difuso
staticice.com	de todo tipo	terceros	general	difuso
ebay.com	de todo tipo	propio	general	difuso
linio.com.mx	de todo tipo	propio	detallado	difuso
bestbuy.com	de todo tipo	propio	general	difuso
carritus.com	supermercado	terceros	general	difuso
mysupermarket.com	supermercado	terceros	detallado	medio
ciao.es	mundo digital	terceros	general	difuso
gocompare.com	seguros	terceros	general	difuso
buscape.com.mx	de todo tipo	terceros	general	difuso

presentar un reporte a un consumidor, pasando por métodos probabilísticos, por técnicas para la gestión de fragmentos enriquecidos (web semántica en minúsculas) y finalmente, aproximaciones basadas en la Web Semántica.

Mas adelante, como propuesta del trabajo, presentaremos en la Sección 4, un modelo de organización de la información basado en la Web Semántica para ofrecer al consumidor reportes precisos de productos, precios y proveedores. Finalmente en la Sección 5 mencionamos trabajo relacionado y concluimos.

2. Análisis de sitios de comercio electrónico

Con el objeto de establecer un marco de análisis de sitios de comercio electrónico, consideramos tres criterios de clasificación: a) origen del inventario, b) las capacidades del filtrado de información en las búsquedas y c) la especificidad del reporte. En cuanto al inventario definimos dos tipos:

1. Sitios con inventario propio.
2. Sitios con inventario de terceros.

Los primeros ofrecen productos a nombre propio y asumen la responsabilidad del cobro y la entrega. En cuando a los segundos incorporan a su catálogo productos de otros proveedores y mediante un clic conducen al vendedor original, en la Tabla 2 se presentan algunos casos.

En cuanto a la calidad de las búsquedas establecemos dos tipos:

1. Filtrado general.
2. Filtrado detallado.

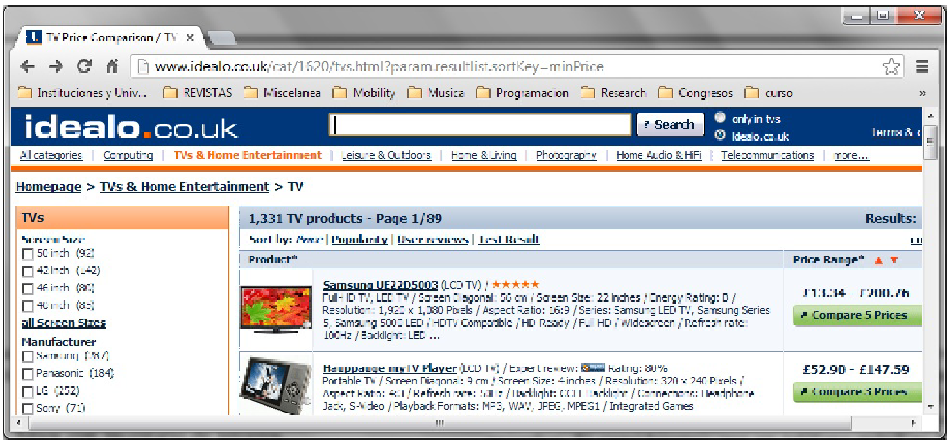


Fig. 1. Sitio para comparación de productos y sus precios con filtrado detallado.

Un filtrado general no permite conocer características precisas de un producto. Por ejemplo al buscar televisiones hay tiendas que permiten buscar a aquellas que tienen un tamaño de pantalla entre 50 y 60 pulgadas, pero cuando el usuario quiere sólo a las que tienen 55 pulgadas, no le es posible recuperarlas.

Hoy en día los sitios más elaborados permiten hacer un filtrado detallado, esto es, ofrecen un amplio conjunto de atributos para consultar artículos con características muy específicas. Regularmente la búsqueda se inicia con una palabra clave, por ejemplo "televisiónz la interfaz permite detallar por marca, tamaño de pantalla, tecnología (LCD, LED, Plasma), resolución, etc. En la medida que se escogen atributos los resultados se refinan, en la Figura 1 se observa una página web del sitio `www.idealo.co.uk` que permite filtrado detallado. Amazon y Linio ofrecen un filtrado detallado y cuentan con inventario propio.

El tercer criterio es la especificidad del reporte.

1. Difuso.
2. Medio.
3. Preciso.

En este aspecto queremos clasificar como preciso a un reporte que es capaz de traer datos de productos pero sólo aquellos solicitados, un reporte medio cuando presenta información de los atributos seleccionados pero también información no solicitada, y finalmente un reporte difuso presenta información de atributos distintos entre productos que componen un reporte.

En la tabla 2 se presenta un resumen con las características de un conjunto de sitios de comercio electrónico. Como se observa en la tabla, ninguno de los sitios ofrece una especificidad de reporte precisa. Existen sitios como `idealo.co.uk` (Figura 1) que presenta un reporte detallado de productos pero que incluye atributos no pedidos, por ejemplo al consultar "television"presenta una foto

por cada registro del reporte y que, por tanto, exige espacio no disponible en dispositivos pequeños. Si consideramos que un teléfono móvil es hoy en día un dispositivo para desplegar resultados es prácticamente incómoda la presentación de tales reportes.

El caso de `idealo.co.uk` es paradigmático, si bien es un sitio con una gran posibilidad de comparación a partir de la cantidad de datos que concentra, también presenta áreas de oportunidad. Por ejemplo, en su sitio se puede leer una invitación a los negocios afiliados a que envíen sus datos de productos como archivos CSV, y de preferencia, diariamente. De aquí deducimos que el sitio obtiene la información a partir de un gran volumen de información entrante que se concentra y sirve para establecer las comparaciones. Si bien es una solución práctica, la posibilidad de actualizar automáticamente la fuente de datos no es evidente.

3. Enfoques tecnológicos

Encontrar nombres de productos y su precio es una inquietud bien identificada en los clientes. La tarea de encontrar páginas web que contengan la información puede ser abordada desde distintos enfoques tecnológicos.

Procesamiento de texto en lenguaje natural

Existe un conjunto de técnicas bien definidas, tales como los métodos de clasificación automática de texto, que pueden aplicarse al texto de la Web una vez que se ha retirado el conjunto de etiquetas (X)HTML que lo componen [12]. Métodos de clasificación tales como: Naive Bayes, Redes Neuronales, Máquinas de Vectores de Soporte [19], etc., presentan un alto grado de madurez y permiten localizar con precisión aceptable texto asociado a alguna categoría de búsqueda, por ejemplo, artículos electrónicos.

Sin embargo, cuando se pretende extraer datos concretos (*tokens*), como por ejemplo el nombre de un producto y su precio, localizado entre una gran cantidad de líneas y textos de páginas de proveedores diversos, los métodos de procesamiento de lenguaje natural, si bien ofrecen un resultado probabilístico aproximado, no garantizan una certeza del 100 % además de que se adolece de un lenguaje de consulta para recuperar unidades de datos: *tokens* específicos.

Fragmentos enriquecidos

Cuando se tienen datos estructurados, por ejemplo el nombre de un evento, su fecha, lugar, hora de inicio, etc., es conveniente utilizar algún esquema que permita registrar la información sin perder la estructura.

Hoy en día existen tres medios para agregar contenido y su semántica incrustado en una página web, estos son: Microformatos, RDFa y Microdata. Los microformatos [10] son anotaciones semánticas que se hacen sobre el atributo `class`, `rel` o `rev` de etiquetas estándar (X)HTML tales como `div` y `span`.

Por otro lado RDFa (*Resource Description Framework-in-attributes* [6]) es la recomendación de la W3C (*World Wide Web Consortium*) para añadir metadatos enriquecidos a nivel de atributos en páginas Web. RDFa permite que, usando algunos atributos se puedan marcar datos legibles por humanos con indicadores permitiendo que los navegadores los entiendan. Una página Web puede contener metadatos que expresen información personal, de productos, eventos, etc. Finalmente, los microdatos (`Microdata` [5] de HTML 5) se pueden añadir a las páginas usando los atributos `itemscope` (para definir un nuevo elemento), `itemtype` (para especificar el tipo de elemento que se describe) e `itemprop` (para especificar una propiedad de ese elemento). Existen iniciativas como `schema.org` para definir vocabularios a emplear para agregar fragmentos enriquecidos.

Si bien se pueden construir rastreadores (*crawlers*) para extraer fragmentos estructurados de las páginas web [14] o bien pasar dichos formatos a un estándar semántico mediante GRDDL [3], también es cierto que se adolece de un mecanismos de consulta que permitan lanzar una petición a la Web y que se devuelva un reporte a partir de la información explorada. Nuevamente para el problema planteado no es la solución apropiada.

Lenguajes semánticos

La respuesta adecuada para estar en la posibilidad de ofrecer reportes que incluyan conceptos y atributos de los productos está en las tecnologías semánticas tales como RDF (*Resource Description Framework*), RDFS (RDF-*Schema*) y OWL (*Ontology Web Language*) [17], ya que proporcionan mecanismos sólidos para la presentación de la información de manera estructurada, sin ambigüedad y por otro lado permite la recuperación de las información a través del lenguaje de consulta SPARQL [4].

4. Modelo de datos

En la presente sección introducimos un modelo de datos con base en la Web Semántica que hace posible el lanzamiento de consultas, la recuperación de información de interés (en función de la consulta), la actualización automática de información y la disposición de reportes precisos para presentarse en el dispositivo emisor.

4.1. Elementos del modelo de datos

El modelo pues, se compone de los siguientes elementos:

Cliente gestor de consultas

La mayoría de las consultas que se lanzan a la Web se expresan en lenguaje natural. Es necesario convertir dicha expresión en una consulta formal que

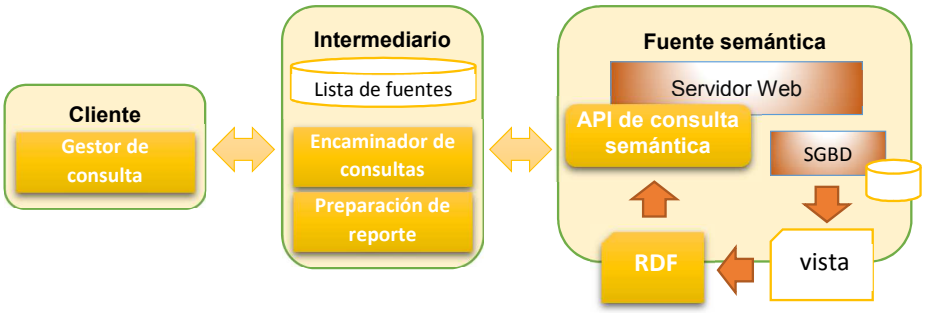


Fig. 2. Modelo de datos para comparación de precios.

puedan computarse. En este sentido es posible construir una consulta compleja a partir de los atributos definidos para cada clase de la ontología. Dicha consulta finalmente se debe expresar en SPARQL.

Un mecanismo simple para formular una consulta SPARQL puede conducirse a partir la representación de los atributos de las entidades como casillas de verificación en una interfaz.

Intermediario

De la misma manera que la Web anterior a la Semántica, la existencia de intermediarios (servidores de búsqueda) es necesaria. La función del intermediario debe ser concentrar un listado de sitios que ofrecen información con una ontología bien determinada, esto es, vínculo web y ontologías empleadas. El intermediario debe proveer un conjunto de servicios web libres para que cualquier empresa se dé de alta en dicha lista de sitios.

Eventualmente un intermediario puede concentrar volúmenes de información provenientes de los sitios registrados, de la misma manera que un servidor de búsqueda actual mantiene un *caché* de información, con la diferencia que este *caché semántico* admitiría consultas complejas con resultados precisos.

Fuentes de datos semánticas

La información que consultan los usuarios a través de un buscador hoy en día, reside en los servidores web empresariales. En un escenario semántico el modelo debería ser el mismo, pero, con características adicionales. Si bien un Servidor Web actual responde a peticiones `http`, una fuente de información semántica debe recibir una consulta SPARQL, gestionar la petición a partir de un conjunto estándar de Servicios Web y regresar el conjunto de datos RDF.

Una posible modificación al Servidor Web es concebible, de la misma manera que se busca el archivo `index.html` como una configuración por omisión, también es posible considerar una adecuación estándar a sus funciones. No obstante

para prueba del modelo los Servicios Web se pueden implementar de manera independiente. La implementación de los Servicios Web puede llevarse a cabo haciendo uso de la librería para gestión de información semántica JENA [9].

Cada empresa posee una infraestructura de cómputo distinta, por ejemplo en cuanto al gestor de base de datos empleado, aún con ello, es posible concebir la preparación de un conjunto de vistas de las bases de datos, a partir de ontologías determinadas, para generar archivos RDF que constituyen la información sujeto de publicación.

Cuando decimos generación de vistas no nos referimos a un sofisticado mapeo de datos como el propuesto en [2], nos referimos a la conformación de archivos RDF que contengan los atributos mínimos de interés para la comparación y en relación a la ontología del dominio de interés a partir de las bases de datos relacionales de la empresa. Por ejemplo dada la tabla Producto y el atributo color, forzosamente se generaría un triple RDF con una instancia de Producto, color como predicado y una instancia de color como valor, por ejemplo, una sentencia semántica podría ser: <tsuru ii><color><azul>.

Una ventaja de colocar la generación de vistas y la posterior generación de archivos RDF es que ésta puede ser una tarea automática a nivel de disparadores (*triggers*) en la base de datos, por lo que la actualización de información sería automática.

Vocabularios formales

Sin duda la utilización de vocabularios formales, esto es, ontologías, es un elemento necesario para la materialización del modelo.

Las consideraciones arriba mencionadas se plasman en el modelo de datos propuesto. En el modelo de la Figura 2 El consumidor (Cliente) emite una consulta mediante interfaz de gestor de consulta. La interfaz debe permitir la selección de atributos a presentar, de modo tal que sea posible construir la consulta SPARQL y obtener sólo los atributos deseados.

La consulta se envía al intermediario, quien a partir de la lista de fuentes de información semántica encaminará la consulta hacia aquellas APIs que conocen la ontología de la consulta origen. La fuente de datos semántica recibirá la consulta, la procesará en sus archivos públicos RDF y devolverá los triples calculados. Finalmente el intermediario recibirá los reportes parciales de las fuentes semánticas y les encaminará hacia el cliente solicitante, quien desplegará el reporte detallado.

4.2. Comportamiento de los componentes del modelo de datos

Existen variaciones en las funciones que pueden asumir los componentes del modelo de datos en su implementación. A continuación se explica:

Fuentes de datos pasivas

Se trata de servidores que sólo publican archivos RDF en sus servidores Web. No ejecutan ninguna acción de filtrado ni gestión de consultas del exterior. En este caso el intermediario y el cliente pueden asumir distintos roles:

intermediario dominante: El intermediario recibe una consulta del cliente, explota la consulta a partir de los archivos RDF de las fuentes de datos. Posee una lista dinámica de vínculos a fuentes de datos. Prepara reporte y lo regresa al cliente solicitante.

intermediario y cliente actuantes: El intermediario administra lista de URLs dinámicas de fuentes de datos. El cliente obtiene lista de fuentes y gestiona reportes por si mismo.

cliente dominante: El cliente conoce los URLs de las fuentes de datos y se comunica directamente con ellas para obtener datos para confeccionar el reporte, el intermediario desaparece.

Con una fuente de datos pasiva, cuando un elemento del modelo requiere realizar una consulta tiene que obtener todo el archivo RDF para construir un grafo RDF, este enfoque hace viajar información que tal vez no se usará.

Fuentes de datos activas

En este caso cada fuente de datos oferta Servicios Web para realizar operaciones de filtrado de datos, recibirá las consultas las procesará y regresará solo los datos necesarios, es un enfoque muy eficiente.

También puede ocurrir un intermediario o cliente dominante, o ambos actuantes.

Fuentes de datos *end-points*

Un enfoque tradicional consiste en instalar un gestor de consultas estándar en cada fuente de datos, estos son llamados *end-points*. En esta parte hace falta que la herramienta empleada soporte características técnicas tales que permita la interacción con agentes artificiales para la confección conveniente de reportes.

5. Trabajo relacionado y conclusiones

Sin duda el problema de comparación de productos a partir de varios proveedores sigue siendo de interés común, como evidencia podemos citar a dos *start ups* destacadas como `FindBest.com` [15] y `Save22` [7] que han obtenido apoyos relevantes para su operación a partir de su propuesta de valor.

Si nos referimos a casos de éxito como los citados en [8], i.e., `Shopping.com`, `Getprice`, `Myshopping`, `Shopbot`, `StaticICE` ellos se basan en el principio de

colectar fuentes de datos (`feeds`), sin embargo la pertinencia de la información depende de la frecuencia de actualización que se haga a partir de la lectura de las fuentes: Nosotros en contraparte proponemos un modelo de consultas en línea.

Por otro lado, muchos vendedores al detalle (*retail*) usan vendedores de terceros para colocar su oferta de productos en algunos sitios como `channeladvisor` (`http://ssc.channeladvisor.com/howto`) para optimizar los datos. Es decir, se depende de un gestor o presentador de información en contraparte con la disposición independiente y en línea que propone nuestro modelo.

Si bien nosotros hacemos referencia a un intermediario, nos referimos a un ente similar a un buscador, como ocurre en la web actual. Además concebimos a las fuentes de datos semánticas como entidades independientes de la misma manera que operan hoy en día los Servidores Web que publican información empresarial y son independientes del buscador.

Como puede verse, la alternativa de ofrecer los datos en formato RDF y en línea, sigue siendo una posibilidad que ofrece muchas ventajas, por ejemplo, los datos están disponibles y actualizados para poder ser recuperados y no se requiere un pago a terceros. Y, la más importante, el reporte puede definirse de acuerdo a la necesidad del cliente o de las necesidades de tamaño del propio dispositivo de despliegue de resultados.

El modelo que proponemos es útil también por ejemplo para la construcción de agentes que recaben información y la presenten en dispositivos móviles en un concepto similar al denominado economía de la intención [16], de cara a un nueva generación de la Web, la Web 4.0, como se propone en [13].

Agradecimiento. Este trabajo ha sido realizado en el marco de proyectos financiados con registro oficial del Tecnológico Nacional de México.

Referencias

1. Aghaei, S., Nematbakhsh, M.A., Farsani, H.K.: Evolution of the world wide web: From web 1.0 to web 4.0. International Journal of Web & Semantic Technology (IJWesT) 3(1) (Jan 2012)
2. Buil-Aranda, C., Corcho, O., Krause, A.: Robust service-based semantic querying to distributed heterogeneous databases. In: Proceedings of the 2009 20th International Workshop on Database and Expert Systems Application. pp. 74–78. DEXA '09, IEEE Computer Society (2009)
3. Consortium, W.W.W.: Gleaning Resource Descriptions from Dialects of Languages (GRDDL). `http://www.w3.org/2004/01/rdxh/spec` (2008)
4. Consortium, W.W.W.: SPARQL Query Language for RDF. `http://www.w3.org/TR/rdf-sparql-query/` (2008)
5. Consortium, W.W.W.: HTML Microdata. `http://www.w3.org/TR/2011/WD-microdata-20110525/` (2011)
6. Consortium, W.W.W.: RDFa 1.1 Primer - Second Edition. `http://www.w3.org/TR/xhtml-rdfa-primer/` (2013)
7. Ho, V.: Asian Price Comparison Site Save 22 Gets Angel Round Of "Mid Six Figures" . `http://techcrunch.com/2013/05/11/asian-price-comparison-site-save-22-gets-angel-round-of-mid -six-figures/` (2013)

8. Huppatz, N.: ECommerce Technology Basics: Part Two Comparison Shopping Engines and Marketplaces. http://www.powerretail.com.au/getting-started/comparison-shopping-engines-and-marketplaces/ (2013)
9. JENA, A.: A free and open source Java framework for building Semantic Web and Linked Data applications. https://jena.apache.org/ (2015)
10. Khare, R.: Microformats: The Next (Small) Thing on the Semantic Web? IEEE Internet Computing 10(1), 68–75 (2006)
11. Lee, H., Yu, Y., S., Jo, G.: Comparison Shopping Systems Based on Semantic Web - A Case Study of Purchasing Cameras. In: Li, M., Sun, X.H., Deng, Q., Ni, J. (eds.) GCC (1). Lecture Notes in Computer Science, vol. 3032, pp. 139–146. Springer (2003)
12. Qi, X., Davison, B.: Web Page Classification: Features and Algorithms. ACM Comput. Surv. 41(2) (2009)
13. Ramos, J., Ramos, R., Navarro, I.: Modelo computacional semántico para la economía de la intención. In: Proceedings of the 10o Congreso Estatal de Ciencia, Tecnología e Innovación, CECTI, Michoacán. CECTI, Michoacán (2015), por aparecer
14. Ramos, J., Solis, R., Oceguera, H., Silva, J.: A Practical Approach to Modeling and Extracting Information from Semantic Web Based on Microformats. In: ENC '09 Proceedings of the 2009 Mexican International Conference on Computer Science. pp. 65–74. IEEE Computer Society (2009)
15. Rao, L.: Data-Driven Comparison Shopping Platform FindTheBest Raises 11M From New World, Kleiner Perkins And Others. http://techcrunch.com/2013/03/05/data-driven-comparison-shopping-platform-findthebest-raises-11m-from-new-world-kleiner-perkins-and-others/ (2013)
16. Searls, D.: The Intention Economy: When Customers Take Charge. Harvard Business School Press, Harvard Business Review Press (2012)
17. Yu, L.: A Developer's Guide to the Semantic Web. Springer (2011)
18. Zhu, H., Madnick, S., Siegel, M.: Enabling Global Price Comparison through Semantic Integration of Web Data. IJEB 6(4), 319–341 (2008)
19. Zubiaga, A.: Comparativa de Aproximaciones a SVM Semisupervisado Multiclase para Clasificación de Páginas Web. Master's thesis, Master Thesis, UNED (Septiembre 2008)

Relación contextual de palabras en libros de Shakespeare usando mapas auto-organizados

Luis Alfredo Moctezuma[1], Jessica López[1], Caleb Jiménez[1], Maya Carrillo[1], Luis Enrique Colmenares-Guillen[1], J. Guadalupe Ramos[2]

[1] Benemérita Universidad Autónoma de Puebla, Puebla, México

[2] Instituto Tecnológico de La Piedad, La Piedad, Michoacán, México

luisalfredomoctezum@gmail.com,{cmaya,lecolme}@cs.buap.mx, {acissejol, calebji,joguar}@hotmail.com

Resumen. El significado de las palabras puede encontrarse por el contexto en el cual estas ocurren. Los contextos de las palabras pueden ser reflejados y agrupados mediante una red SOM (*Self- Organizing Map*). En los experimentos que se muestran en el presente trabajo, se aprecian los diferentes grupos de palabras, obtenidos al capturar información de contexto, empleando representaciones vectoriales de palabras, en tres obras de Shakespeare: Macbeth, Julio Cesar y Hamlet. Los resultados obtenidos muestran que los SOM funcionan de manera adecuada para identificar los diferentes conceptos que los autores manejan en sus obras.

Palabras clave: Redes neuronales, mapas auto-organizados, relación contextual, Shakespeare.

1. Introducción

El procesamiento de lenguaje natural (PLN) se ocupa del reconocimiento y utilización de la información expresada en lenguaje humano para ser empleada en sistemas computacionales. En su estudio intervienen diferentes disciplinas tales como: lingüística, inteligencia artificial, filosofía, matemáticas, psicología, y ciencia cognitiva. En PLN para analizar los textos generalmente se consideran cuatro niveles de análisis: léxico, sintáctico, semántico y pragmático. Si bien la mayoría de la técnicas están basadas en análisis léxico y sintáctico, es evidente la necesidad de comprender el significado de los textos para lograr elevar el desempeño de diversas tareas de PLN como análisis de sentimiento, la generación automática de reportes, recuperación de información, búsqueda de respuestas, por mencionar algunas. Sin embargo, las técnicas de representación del significado no han obtenido los resultados deseados, y numerosas cuestiones continúan sin encontrar soluciones satisfactorias.

Definir qué es el significado no es una tarea sencilla, y puede dar lugar a diversas interpretaciones. Es posible distinguir entre significado independiente y significado dependiente del contexto. El primero, tratado por la semántica, hace referencia al significado que las palabras tienen por sí mismas sin considerar el significado adquirido según el uso en una determinada circunstancia. Por otra parte, el componente significativo de una frase asociado a las circunstancias en que ésta se da, es estudiado por la pragmática y conocido como significado dependiente del contexto.

Un mapa auto-organizado (SOM) es una herramienta que analiza datos en muchas dimensiones con relaciones complejas entre ellos y los presenta en una visualización sencilla en sólo dos dimensiones. La propiedad más importante de una SOM es que preserva las propiedades topológicas de los datos. Dichos mapas pueden utilizarse para visualizar roles contextuales de las palabras, es decir similitud en su uso en contextos cortos formados por las palabas adyacentes, este es el objetivo del presente trabajo tomando como fuente obras de Shakespeare.

En diferentes aplicaciones, los datos de entrada a un SOM, son numéricos. Sin embargo al trabajar con palabras y considerar su contexto, el orden en el que aparecen dichas palabras es importante y no basta con representarlas por un número, como por ejemplo: frecuencia de términos o frecuencia inversa. En este trabajo, cada palabra se representó como un vector cuyas entradas representan una secuencia única de unos y ceros. El contexto de una palabra *a* fue capturado sumando los vectores únicos de las palabras adyacentes con las que aparecía a lo largo del texto, considerando una ventana simétrica de dimensión uno. Los resultados obtenidos muestran que la similitud entre palabras puede definirse indirectamente, capturando su significado en función del contexto en el que aparecen (palabras vecinas).

Este artículo está organizado de la siguiente manera en la sección 2 se introducen algunos trabajos relacionados, en la sección 3 se define lo que es un SOM y el procedimiento seguido para el aprendizaje de la red, en la 4 se describe la representación utilizada para las palabras, en la 5 las condiciones de operación del SOM, en la sección 6 se presentan los resultados y finalmente en la 7 las conclusiones.

2. Trabajo relacionado

En [1] se utilizar un SOM para organizar las palabras en categorías gramaticales representadas en una matriz bidimensional. La similitud de las categorías se refleja en función de su distancia sobre la matriz. Este tipo de mapa de categorías de palabras, puede utilizarse en aplicaciones para analizar colecciones grandes de documentos. En [3] se identifican categorías gramaticales empleando un corpus en chino, se emplean vectores de contexto de dimensión 650 y para capturar la información de contexto ventanas simétricas de dimensión 2.

3. Mapas auto-organizados

Un SOM es un tipo de red neuronal, útil para tareas de agrupamiento y auto-organización de grandes cantidades de datos de manera eficiente. T. Kohonen [8] los

presentó por primera vez en 1982. La principal característica de los SOM es que preservan las relaciones topológicas. Su finalidad es descubrir la estructura subyacente de los datos introducidos en él. Este tipo de red consiste de un conjunto de neuronas sobre una cuadricula de $N = \{n_1, n_2, \dots, n_k\}$ que se conectan de manera idéntica a la entrada *X*. La localización de cada neurona sobre la cuadricula está representada por el valor de entrada asociado. Para determinar si una neurona *i* está cerca de una neurona *j*, se calcula la distancia euclidiana, generalmente:

$$d(n_i, n_j) = || r_i - r_j ||.$$

En el proceso de entrenamiento de la SOM las neuronas interactúan entre ellas, la relación entre ellas está regulada por una *función de vecindad* $H(|| r_i - r_j ||)$ que mide la intensidad de la relación entre neuronas, la interacción o intercambio de información es más fuerte cuando la distancia entre neuronas es pequeña. Se tiene que *H* tiene un parámetro *p* el cual representa el radio de la vecindad de las que serán consideradas neuronas cercanas. Una neurona es vecina de otra si la *función de vecindad* es pequeña de acuerdo a la función Gaussiana.

Para las neuronas que son vecinas se realiza un proceso de cooperación y competencia para determinar que neurona es la ganadora y modificar los pesos propios y de neuronas vecinas calculadas con la función de vecindad. Para esta fase puede utilizarse el modelo de *sombrero Mexicano*. Todas las neuronas tienen cierta influencia sobre sus neuronas vecinas, esto es por la *función de vecindad*.

En el proceso de entrenamiento se utiliza un conjunto finito de datos de entrada $X = \{x_0, \dots, x_{m-1}\} \subset R^n$. Lo que busca la red es encontrar neuronas que tengas pesos similares e ir modificándolos en cada iteración, para que si las neuronas están cerca según la distancia euclidiana se junten o separen. Como se especifica en los siguientes pasos:

1. Cada nodo se inicializa con un peso (aleatorio) (W)
2. Se selecciona al azar un vector del conjunto de entrenamiento.
3. Se calcula el nodo de la red que tiene el peso más similar al vector anterior. Para ello, simplemente se calculan las distancias euclidianas entre los vectores W de cada nodo y el vector de entrenamiento.
4. Se calcula el radio de la función de vecindad. Este radio comenzará siendo grande (como para cubrir la red completa) y se va reduciendo en cada iteración.
5. Cada nodo en el radio de la vecindad ajusta su peso para parecerse al vector de entrenamiento seleccionado en el paso 2, de forma que los nodos que son vecinos se vean más modificados siguiendo la siguiente formula.

$$W_j(n+1) = W_j(n) + \wedge_{ij}(n)\eta(n)\Big(X(n) - W_j(n)\Big),$$

donde η es la taza de aprendizaje y $\wedge_{ij}(n) = e^{\left[\frac{-d^2_{ij}}{2\sigma^2(n)}\right]}$ es la función gaussiana que calcula el radio de la vecindad, d es la distancia entre las neuronas y σ disminuye en cada iteración.

6. Repetir desde el paso 2 (el número de iteraciones que se considere necesario).

Las SOM se han utilizado en diversos trabajos por ejemplo para simular la adquisición del lenguaje [4], visualizar agrupamiento [5], su aplicación en procesamiento de lenguaje natural puede comprobarse en [6, 7].

4. Representación vectorial

En el presente trabajo, se analizaron tres libros de William Shakespeare en idioma inglés: Macbeth, Julio Cesar y Hamlet. El número de palabras analizadas fueron 67,805, obteniendo un vocabulario de 13,118 palabras. El vocabulario es poco común, ya que por el año en que fueron escritos, se usan palabras en diferentes idiomas dentro de conversaciones.

Cada uno de los tres libros se analizó por separado. Se eliminaron marcas de puntuación y caracteres especiales, todas las letras en mayúsculas fueron sustituidas por su correspondiente letra minúscula. Los artículos y preposiciones también fueron eliminados de los textos, así como las palabras de frecuencia menor a 3.

Para una palabra *a* que denominaremos clave, el contexto fue capturado considerando la palabra que la precede y sucede, así se formaron tramas de la forma ("predecesor", "clave", "sucesor"). Cada palabra fue representada con una sucesión de 24 dígitos binarios únicos. Para capturar el contexto se crearon vectores de dimensión 72. En los primeros 24 dígitos se almaceno la suma vectorial de todas las representaciones de las palabras que precedían a la palabra clave en el texto, y en los últimos 24 la suma vectorial de todas las palabras que sucedían a la palabra clave. Los 24 dígitos intermedios representaron las diferentes palabras del vocabulario. En la Tabla 1 se puede apreciar un ejemplo de los valores únicos en los vectores para cada una de las palabras, la parte inicial y final a ceros antes de iniciar la captura del contexto.

Tabla 1. Vector de ejemplo con 8 valores binarios creado para cada palabra clave.

Palabra	Clave
reason	00000000 00010000 00000000
beare	00000000 01101000 00000000
heart	00000000 00010100 00000000
roome	00000000 00110001 00000000

La Tabla 2 presenta un ejemplo de dos contextos para la palabra clave *beare*, donde se tienen los predecesores diferentes. Como las palabras tienen representaciones únicas asociadas, se utilizan dichas representaciones para formar los contextos sumando la representación de los antecesores y sucesores a *beare*, Tablas 3 y 4.

Tabla 2. Contextos para la palabra *beare*.

Predecesor	Clave	Sucesor
reason	beare	heart
roome	beare	reason

Tabla 3. Vector de contexto para la palabra "beare", sumando predecesor y sucesor, las 24 posiciones mostradas constituyen ahora la representación de beare.

Predecesor	Clave	Sucesor	Palabra
00010000	01101000	00010100	beare
00110001	01101000	00010000	beare

Tabla 4. Vectores únicos de cada palabra a ocho dígitos, sin presentar los ocho ceros anteriores y posteriores.

Predecesor	Clave	Sucesor	Palabra
01000001	11010001	00100100	beare

Una vez obtenidos los vectores de contexto para las palabras del vocabulario, estos fueron la entrada al SOM, a continuación se describe el proceso seguido para obtener el agrupamiento contextual de las palabras.

5. Proceso de aprendizaje de la red neuronal

Los vectores de contexto fueron usados como entrada en la red neuronal. Las SOM utilizada fue de dimensiones 7 por 9 neuronas, cada neurona representa una colonia o cluster de palabras. Este paso permitió encontrar la relación contextual de las palabras clave de los vectores de contexto. Para etiquetar las palabras en la red se usaron las técnicas presentadas por Teuvo Kohonen en [3]. La representación de las etiquetas en la red fue realizada con 794 palabras obtenidas de cada libro. El algoritmo se ejecutó tres veces, una vez por cada libro.

En un SOM al cabo de suficientes iteraciones, el espacio de datos de entrada es cubierto por el mapa y cada dato del espacio multidimensional de entrada tiene una proyección en el espacio bidimensional de salida, es decir, cada palabra puede ser ubicada en una celda del mapa y en celdas vecinas palabras con vectores similares, como puede observarse en las Figura 1,2, y 3.

6. Resultados

En Fig. 1 se pueden apreciar los resultados de la obra Macbeth de Shakespeare. En el cluster de la parte superior izquierda de la figura se observa que la red asoció palabras que tienen que ver con el cuerpo humano, por ejemplo: *sangre*, *ojos*, y *corazón*. En la parte central superior de la imagen el agrupamiento se realizó a partir de diferentes roles de las personas en la sociedad: *esposo, persona*, *amigo*, *hijo, hermano, papá*. El tercer agrupamiento de la red SOM, está directamente relacionado con muerte y dolor, siendo algunas de las palabras encontradas las siguientes: *muerto, muerte, noche, peligro, trueno*, *tirano*, entre otras. El cuarto agrupamiento corresponde a palabras de valentía y lealtad, por ejemplo: *verdad, honor, espíritu, fortuna,* entre otras. Por último, el quinto agrupamiento para esta obra es un conjunto

de palabras de enfrentamientos: *traición*, *antorcha*, *espada*, *contender*, *fantasma*, *traidor*, etc.

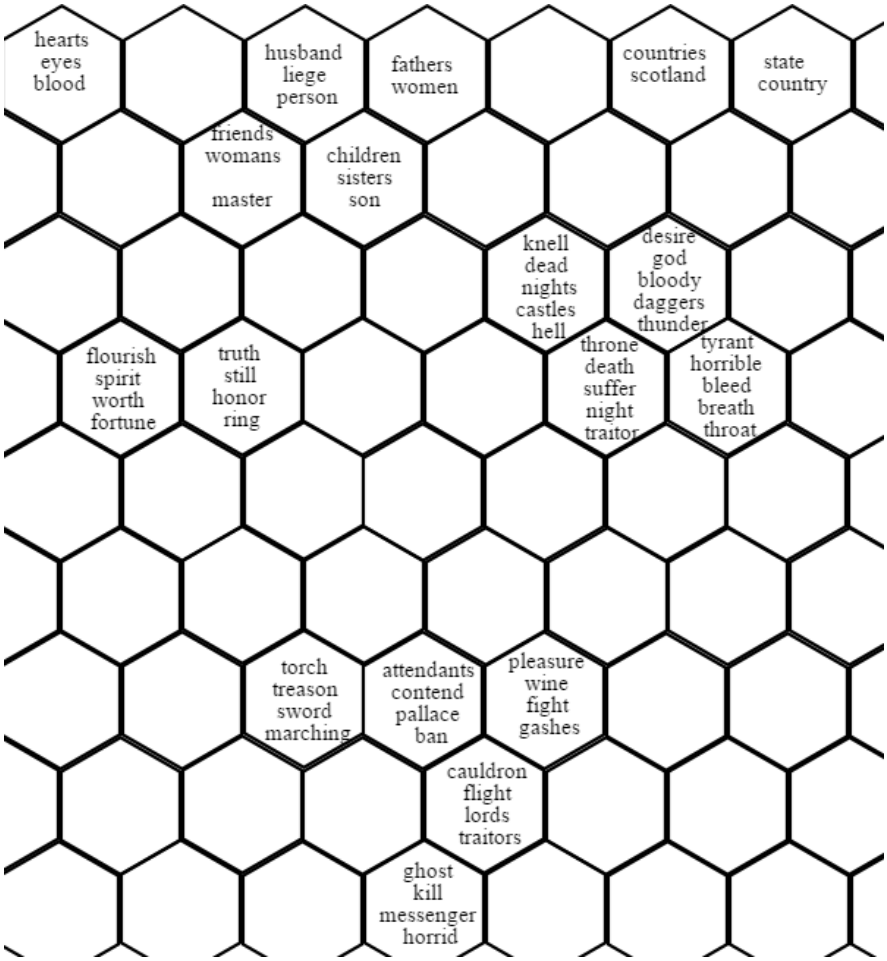


Fig. 1. Resultados del agrupamiento de la red SOM para la obra Macbeth de Shakespeare.

La Fig. 2 muestra el agrupamiento que la red SOM obtuvo de la obra Julio Cesar de Shakespeare. El primer grupo localizado en la parte superior izquierda de la imagen reconoce palabras relacionadas con enfrentamientos, por ejemplo: *pelea*, *débil*, *conspiración*, *soportar*, *cabezas*, *gloria*, entre otras.

Otro grupo, localizado en la parte superior izquierda de la imagen, contiene palabras de independencia, tal es el caso de: *honorabilidad*, *poder*, *libertad*, *conquista*, *fortuna*, *ambición* y *nobleza*. Un tercer grupo alberga palabras de guerra: *corazón*, *honor*, *sangre*, *muerte*, *enojo*, *peligro*, *culpa*. Un cuarto agrupamiento es de palabras directamente relacionadas con los humanos, por ejemplo: *hombres*, *hermano*, *soldado*, *hijo*, *mujer*, *mujeres*, *esposa*, *gente*, *hombre*, *rey*, *señor*, etc.

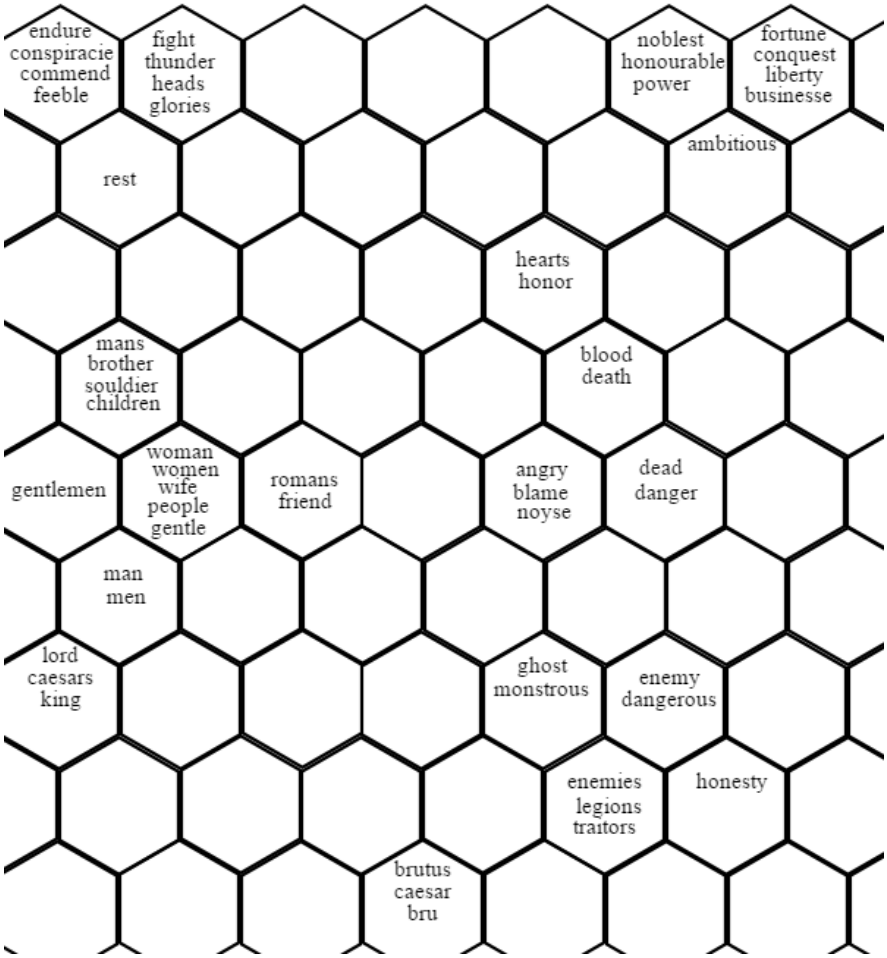


Fig. 2. Resultados del agrupamiento de la red SOM para la obra Julio Cesar de Shakespeare.

En la Fig 3 se muestra el agrupamiento realizado por la red SOM en la obra literaria Hamlet de Shakespeare. Uno de los agrupamientos identificados para esta obra fueron palabras de adjetivos, por ejemplo: *sonriente*, *amable*, *libre*, *humilde*, *extraño*, etc. Un segundo agrupamiento se refiere a palabras que tienen que ver con los humanos: *hermano*, *hermana*, *esposa*, *marido*, *hijo*, *hombre*, *rey*, *criatura*, *joven*, *chicas*, *rey*, *señores*, entre otras. El tercer agrupamiento de la imagen mostrada refleja palabras asociadas con terminología policiaca: *ley*, *crimen*, *causa*, *fuerza*, *violencia*.

7. Conclusiones

El análisis de las obras de Shakespeare, tratando de identificar los principales conceptos empleados, utilizando mapas auto-organizado resultó una técnica adecuada.

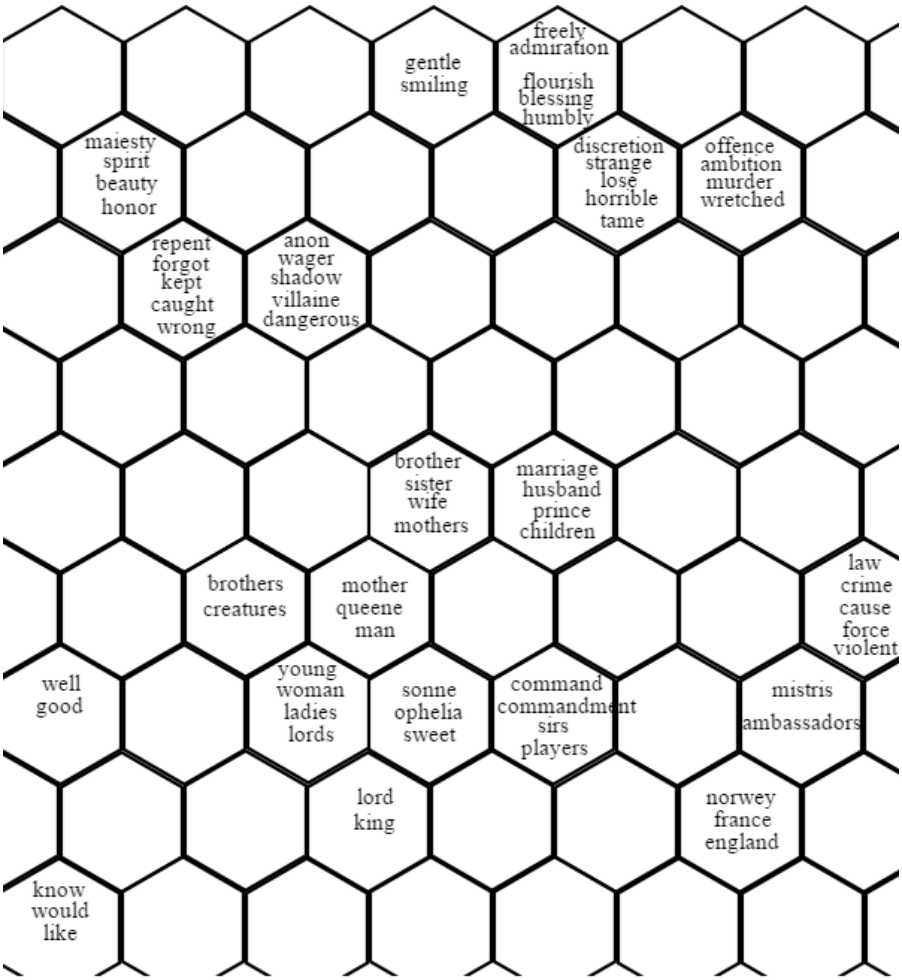


Fig. 3. Resultados del agrupamiento de la red SOM para la obra Hamlet de Shakespeare.

Construir modelos simbólicos explícitos como el análisis del contexto es más fácil para los lingüistas y analistas de obras, debido a que pueden saber de manera fácil y rápida, que temas son los abordados por un determinado autor, cual es la tendencia de palabras, y también las categorías de palabras que existen en un conjunto de obras literarias, sin embargo, esto no es fácil de automatizar. Con los experimentos presentados comprobamos que los SOM pueden utilizarse con este fin.

En el presente experimento se analizaron 3 obras de Shakespeare, como trabajo a futuro se propone analizar todas las obras de Shakespeare y encontrar la tendencia de las palabras que utiliza en sus obras, el uso de algunas palabras por su contexto en la narrativa de cada una de las obras, y además analizar un corpus más grande de otros autores de la época para obtener más información de tipo de narración de ese tiempo. Con esto posiblemente podamos aplicar la misma técnica desarrollada en tareas de atribución de autoría.

Referencias

1. Honkela, T.: Self-Organizing Maps of Words for Natural Language Processing Applications (2012)
2. Shakespeare, W.: Internet Shakespeare Editions. [Online] http://internetshakespeare.uvic.ca/Library (2013)
3. Kohonen, T.: Matlab implementations and applications of the self organizing map. Helsinki, Finland (2014)
4. Li, P., Zhao, X.: Self-organizing map models of language acquisition. Frontiers in psychology 4 (2013)
5. Vesanto, J., Alhoniemi, E.: Clustering of the self-organizing map. IEEE Transactions on Neural Networks 11(3), pp. 586–600 (2000)
6. Miikkulainen, R.: Subsymbolic natural language processing: An integrated model of scripts, lexicon, and memory. MIT press (1993)
7. Scholtes, J.C.: Kohonen Feature Maps in Natural Language Processing (1991)
8. Kohonen, T.: Self-organized formation of topologically correct feature maps. Biological Cybernetics 43 (1982)
9. Krista, L.: Self-organized Maps of Documents Collections: A new Approach to Interactive Exploration. Association for the advancement of artificial intelligence (1996)

An Image Rotation Approach for Hand Dorsal Vein Recognition

Ignacio Irving Morales-Montiel, J. Arturo Olvera-López, Ivan Olmos-Pineda

Benemerita Universidad Autonoma de Puebla,
Facultad de Ciencias de la Computacion, Puebla, Pue.,
Mexico

cpycon@hotmail.com,{aolvera,iolmos}@cs.buap.mx

Abstract. In Biometric recognition, commonly the information about the biometric to be analyzed is contained in digital images, in particular this work is focused on analyzing hand dorsal veins as biometric. A basic process during the recognition stage is the feature (minutiae) extraction; when images are captured from several people it is very difficult to obtain a global alignment and as consequence different orientation images are obtained which makes more difficult the feature extraction. In this work we propose an approach based on central hand points for auto-rotating hand thermal images; based on the results our approach is able to rotate images from different orientations and obtains homogeneous alignment of the tested images.

Keywords: Hand dorsal thermal image rotation, segmentation, biometric systems, hand vein recognition.

1 Introduction

Nowadays several aspects in the real life are aided by computer technology; in particular some difficult tasks for humans can be solved using computer systems. Mainly, the difficult tasks for humans are related to those problems involving huge amounts of information to be analyzed in short periods of time. As a particular case of this kind of difficult human tasks, we can state the person recognition problem in public places (during the whole day i.e. twenty four hours) where the goal is the identification of at least one person in a scenario in which there exists not only the person to identify but also other people and some other objects.

In Computer Science, there exists an area which is focused on solving the person recognition problem , this area is named Biometric Recognition which is defined as the science of establishing the identity (set of attributes associated with a person) of an individual based on physical or behavioral characteristics of the person in either automatic or semi-automatic manner; each one of these characteristics are known as

biometric, in particular the following factors must be considered in order to determine the suitability of the biometric for being considered as descriptive characteristic of the person: *Universality* (Every person must posses it), *Uniqueness* (different across individuals in the population), *Permanence* (time invariant) and *Circumvention* (difficult to be imitated by either impostor or artifacts). Person identification is useful in some fields such as security for access control, corpse identification, criminal investigation, missing children, parenthood determination, among others [1].

Some examples of biometrics commonly used are: fingerprint, palm print, iris, voice, face, ear structure, gait, among others [2-6] which fulfill the descriptive characteristics mentioned above. In general, a Biometric system (Figure 1) consists of the following modules:

-*Sensor Module:* In this module the capture of the biometric is carried out and the digital sensor used depends on the biometric to be analyzed. In the case of voice the required sensor is a microphone but in the case of other biometrics such as iris, face, gait, fingerprint and hand the sensor used is a device based on CCD (Charge-Coupled Device), particularly digital cameras are commonly used.

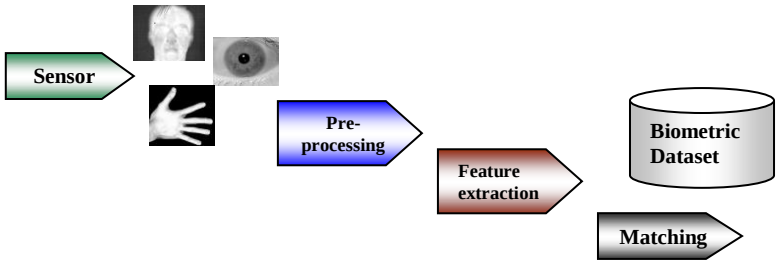


Fig. 1. Modules in a Biometric System: Sensor, Pre-processing, Feature extraction and Matching. The biometric images (face, hand, iris) were obtained from [7].

-*Pre-Processing module.* Once the biometric has been digitalized (captured) it is necessary to prepare the information provided by the biometric. When the biometric is captured as a digital image, typical algorithms applied in this module are focused on denoising data, enhancing contrast, equalization, rotation, among others. This pre-processing phase is fundamental in any biometric system since the better quality of the biometric information the better identification accuracy is obtained.

-*Feature extraction module*. This component is focused on detecting and extracting the most relevant components from the biometric which are enough descriptive for characterizing the person's identity. In particular, for the case of digital images, the first process applied in this stage is the extraction of ROI (Region of Interest) that is the sub region (from the whole image) in which the main descriptors of the biometric are located. The ROI extraction is followed by a segmentation process in order to isolate regions which indeed are relevant for biometric purposes (i.e. isolating them and discarding other non useful components in the image such as the background).

-*Matching module.* In this module, based on the extracted features, they are analyzed in order to determine the person's identity i.e. recognize the person. In

biometric systems the main task in the context of recognition is to establish the association between an individual and his personal identity. For establishing this association some algorithms known as supervised classifiers are used. The classifier learns patterns from the biometric and matches them with other samples of valid users for determining whether the person's identity is accepted. In the context of supervised classifiers, this process is named classification and the database used for matching is named training set; the classifier inputs corresponds to the biometric descriptors obtained in the previous biometric modules. When the matching process is not successful then it is said that the person trying to recognize is an impostor or intruder.

In the literature, several biometrics have been analyzed for person recognition purposes but recently the hand dorsal veins also are used as biometric; in this work an approach for auto-rotating hand dorsal vein images is proposed which is applicable in the pre-processing module of any biometric system.

The content of this paper is organized as follows: Section 2 describes some relevant concepts about hand veins as biometric; in section 3 the proposed approach for rotating is described and some experimental results are presented; finally in section 4 the conclusions from this work are given.

2 Hand Dorsal Veins as Biometric

As mentioned in section 1, there exists several biometrics which can be considered as baseline for person recognition; recently the hand dorsal veins has been analyzed and used as biometric [8, 9]; for capturing this biometric a digital camera is used but these kind of devices operates in the visible range (400-700nm wavelength) of the electromagnetic spectrum i.e. illuminating the region to capture using visible light. In the captured images it is difficult to discern the hand dorsal veins since in some cases the person's hand veins are not superficially visible due some corporal factors such as complexion or skin color. On the other hand, the Infrared (IR) light (2.5-50μm wavelength) is a non visible light (for the human eye) that penetrates into the skin about 3mm depth. Due to the hemoglobin properties in the veins, it absorbs the infrared radiation and as consequence, the veins are contrasted in a dark color. When IR light is used to capture dorsal vein images, it is necessary to capture via IR cameras which are known as thermal cameras and the captured images are named IR images or thermal images since the temperature (thermal) is a kind of radiation in the IR band.

The vein pattern in the dorsal hand is a vast network of blood vessels underneath a person's skin. Similarly to fingerprints, the shape of vascular patterns is different among different people and it is stable over a long period of time. In addition, as the blood vessels are hidden underneath the skin and they are mostly not visible to the human eye and as consequence they are difficult to copy by impostors. Another characteristic of the hand vein recognition is that it requires analyzing alive people because when person is not alive the vein pattern would change, also temperature, color and contrast (in the IR band).

For processing hand thermal images as initial step they must be denoised since in the IR band the images are contaminated by salt and pepper noise. In order to reduce

the amount of noise some softening filter (mean, median, alpha-trim, Gaussian, etc.) can be applied.

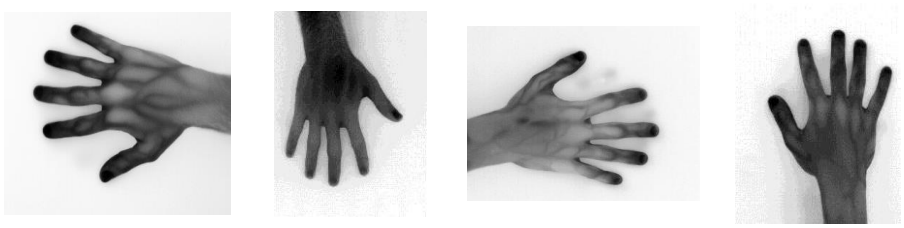


Fig. 2. Some hand thermal images where the hands were captured with different orientations [10].

Before the feature extraction step, an image rotation method must be used in order align all the images because usually when the hand images are captured, not all the hands are oriented in exactly the same direction. In figure 2, some hand images in different orientations are depicted; it can be seen than the images are not oriented in the same direction which makes difficult the ROI and feature extraction processes. Due the orientation problem, in this work we propose an approach for rotating hand thermal images in order to align them in the same direction; in the next section our approach is described.

3 Image Rotation through Central Hand Points

As it was stated before, the feature extraction is a crucial stage in a biometric system because the extracted features are the baseline (input) in the training set used by classifiers. In the case of hand dorsal veins as biometric the patterns learned by classifier are similar to that learned in fingerprints; the hand veins are vast net of veins interconnected therefore the bifurcations and intersections (minutiae) from them (Fig. 3) are distinctive, unique and different from each person to other [11-13]. Due the nature of minutiae in hand vein, the spatial description is sensible to minimal changes in rotation which impacts the feature extraction phase and an automatic image alignment method is required.

In this work, we present an auto-rotating method for thermal hand vein images; this method is named Rotation through Central Hand Points (RCHP) and its main idea consist in finding reference points from which the rotation is carried out in order to align the image. RCHP is mainly useful in the cases where there are not land marks or pivot points taken as reference to rotate an image.

RCHP finds two central points $R_{p1}=(x_{r1},y_{r1})$ and $R_{p2}=(x_{r2},y_{r2})$ which are the central points of the hand and wrist respectively and they both are common regions in any dorsal hand image captured. Through R_{p1} and R_{p2} the reference line $\overline{R_{p1}R_{p2}}$ is considered to rotate the image θ degrees (Figure 4). For computing the θ value the tangent definition is used as in expression (1) where d_x and d_y are the differences

between x_{r2},x_{r1} and y_{r2},y_{r1} respectively (d_x= x_{r2}-x_{r1}, d_y=y_{r2}-y_{r1}). These differences let us know how the hand is initially oriented (Fig. 4) and it must be considered when assigning the θ value.

$$\theta = \begin{cases} \tan^{-1}\left(\frac{dy}{dx}\right) & if \ \ (dy \geq 0 \wedge dx \geq 0) \vee (dy < 0 \wedge dx < 0) \\ 180 + \tan^{-1}\left(\frac{dy}{dx}\right) & if \ \ (dy \geq 0 \wedge dx < 0) \vee (dy < 0 \wedge dx \geq 0) \end{cases} \quad (1)$$

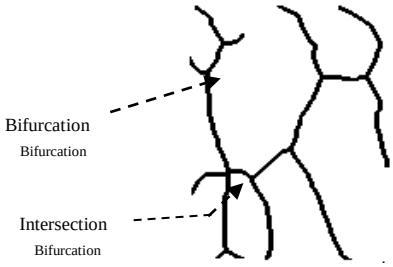


Fig. 3. Hand vein minutiae: bifurcations and intersections, which are distinctive, unique and different among people.

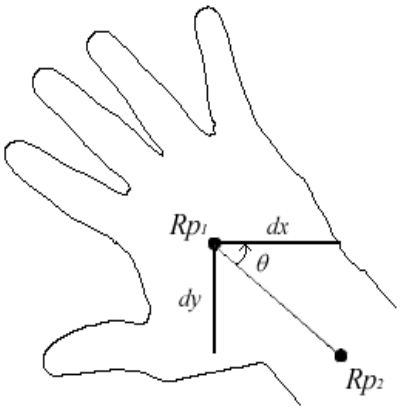


Fig. 4. Hand central reference points R_{P1}, R_{P2} and the reference line $\overline{R_{P1}R_{P2}}$ used for rotating.

According to expression (1) it can be noticed that as result of the rotation, for the four cases, the image is vertical oriented i.e. the fingers will be located in the bottom region and the wrist in the top region.

In figure 5, the RCHP auto-rotation step by step process is shown. As initial step, the hand must be segmented through binarization (Fig. 5b) using as threshold the median of the gray scale in the image and a mask is obtained. The central point R_{P1} of the hand (fig. 5c) is computed as the mean point (x_m,y_m) among all the white pixels in the mask. The central point of the wrist R_{P2} is obtained by finding the region where the wrist is located.

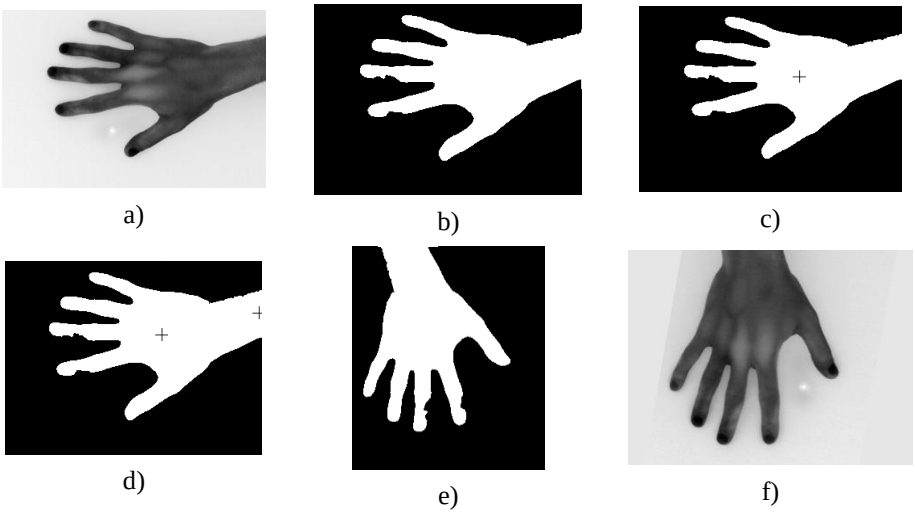


Fig. 5. RCHP step by step process. **a)** Hand thermal image. **b)** Mask obtained after binarization **c)** Hand central point **d)** Hand central and wrist central points obtained. **e)** Mask rotated θ degrees. **f)** Final result obtained by RCHP.

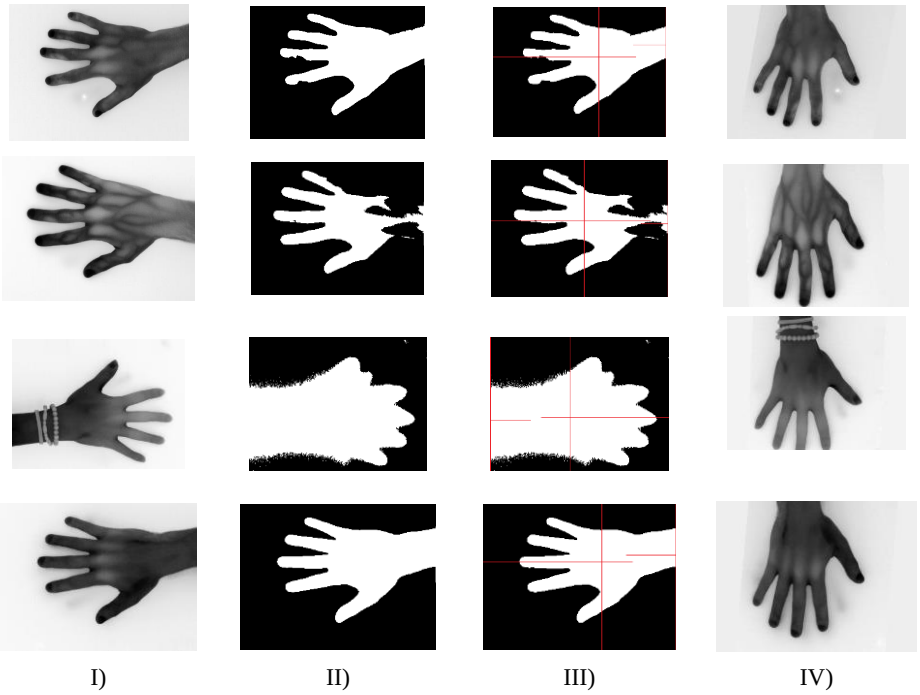


Fig. 6. Results obtained by RCHP. **I)** Hand thermal images. **II)** Central points found. **III)** Mask obtained alter binarization. **IV)** Final image rotated.

For finding the wrist, the border of the mask is processed in order to obtain the two lateral points $l_{w1}=(x_{w1},y_{w1})$, $l_{w2}=(x_{w2},y_{w2})$ that constitute the wrist; l_{w1}, l_{w2} are found by searching the two pixel over the mask border such that they have the most white pixels connected i.e. lateral points defining the wrist and R_{P2} is approximated as the mean of the lateral points (fig 5d) using (2).

$$R_{P2} = \left(\frac{x_{w1} + x_{w2}}{2}, \frac{y_{w1} + y_{w2}}{2} \right) \quad (2)$$

After R_{P2} and R_{P2} are obtained, the mask is rotated based on (1) and the final result of the rotation by RCHP is depicted in figures 5e and 5f.

RHCP was applied over hand thermal images taken from the Technocampus dataset [10] which contains hand thermal images from 100 users and the size of each image is 320x240 pixels. The images shown in figure 2 are some examples of the images from Technocampus dataset.

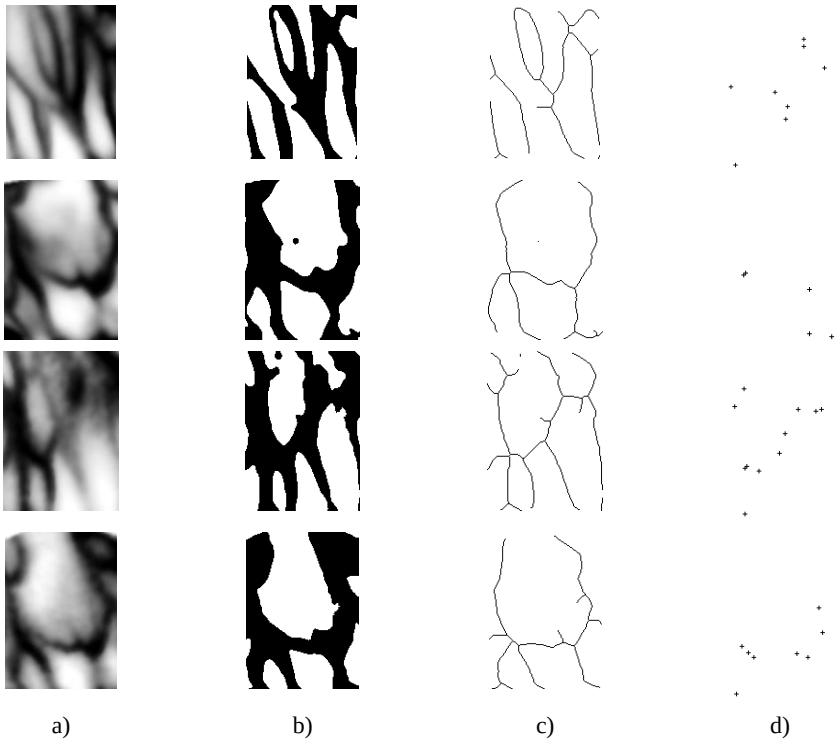


Fig. 7. Minutiae extraction from hand dorsal veins. **a)** Hand thermal images enhanced. **b)** Hand veins segmented. **c)** Thinned veins. **d)** Minutiae extracted.

In figure 6, the first column depicts the hand image to be rotated by RCHP; the second column depicts the mask found after binarization. The third and fourth columns show the central points found and the results of the rotation obtained by RCHP. Based on the experimental results, it can be seen that RCHP obtains vertical

and homogeneous alignment from hands originally captured in different orientations. The RCHP usefulness is mainly the benefit for the feature extraction module in a biometric system because of the spatial description of the minutiae in hand vein patterns.

The next phase of the pre-processing is the minutiae extraction, as another experiment we segmented veins from images applying dynamic enhancement and 2-D Wiener filters as suggested and tested in [13, 14]. In figure 7a, the results of vein enhancement are reported. For segmenting veins the binarization applied takes as threshold the mean and deviation of sub-regions of size 5x5 over the image; this result is depicted in figure 7b. Once the veins are segmented, they must be thinning to be reduced to their minimal structure representation; for thinning we used morphological operators using 3x3 identity structural element (results are shown in fig 7c). Finally, the minutiae were extracted by analyzing the neighborhood of each pixel in the thinned veins such that intersections and bifurcations have at least one neighbor defining either bifurcation or intersection forms.

4 Conclusions

In biometric systems one of the modules is the minutiae (feature) extraction which is a crucial phase since the minutiae are the baseline for the classifiers used in the person recognition stage. Due the kind of minutiae considered in hand dorsal veins (intersections, bifurcations) their spatial/geometric description is highly sensitive to minimal rotation, which in most of the cases is not controlled when capturing. In this work we proposed an approach named RCHP and based on our experimental results it is able to auto-rotate hand dorsal images in order to obtain images vertically oriented and as result the minutiae process is much less sensitive to the original orientation of the image. RCHP is based on finding central points of the hand and through them the rotation is carried out. A special characteristic of RCHP is that it does not require landmarks or reference points specified during the capture as parameters for auto-rotating hand dorsal vein images.

As future work we will evaluate the minutiae (bifurcations, intersections) extracted though RCHP rotation not only using statistical classifiers but also analyzing the hand recognition as a problem in the graph domain where the recognition is mapped to a graph matching problem therefore we will work on defining a representation of the veins features in the graph space.

Acknowledgment. This research is supported through the project OLLJ-ING15-I from VIEP-BUAP, Puebla, Mexico.

References

1. Jain, A.K., Ross, A.A., Nandakumar, K.: Introduction to Biometrics. Springer (2012)
2. Aboshosha, A., El Dahshan, K.A., Karam, E.A., Ebeid, E.A.: Score Level Fusion for Fingerprint, Iris and Face Biometrics. International Journal of Computer Applications 111(4), pp. 47–55 (2015)

3. Xing, X., Wang, K., Lv, Z.: Fusion of Gait and Facial Features using Coupled Projections for People Identification at a Distance. Signal Processing Letters 22(12), pp. 2349–2353 (2015)
4. Menotti, D., Chiachia, G., Pinto, A., Robson Schwartz, W., Pedrini, H., Xavier Falcao, A., Rocha, A.: Deep Representations for Iris, Face, and Fingerprint Spoofing Detection. IEEE Transactions on Information Forensics and Security 10(4), pp. 864–879 (2015)
5. Shinzaki, M., Iwashita, Y., Kurazume, R., Ogawara, K.: Gait-based person identification method using shadow biometrics for robustness to changes in the walking direction. In: IEEE Winter Conference on Applications of Computer Vision (WACV), pp. 670–677 (2015)
6. Jacob, L., Raju, G.: Ear Recognition Using Texture Features-A Novel Approach. In: Advances in Signal Processing and Intelligent Recognition Systems, pp. 1–12, Springer International Publishing (2014)
7. Computer Vision Research Laboratory dataset. University of Notre Dame. http://www3.nd.edu/~cvrl/CVRL/CVRL_Home_Page.html
8. Heenaye, M., Khan, M.: Representation of Dorsal Hand Vein Pattern Using Local Binary Patterns (LBP). In: S. El Hajji et al. (Eds.), C2SI 2015, LNCS 9084, pp. 331–341, Springer-Verlag (2015)
9. Djerouni, A., Hamada, H., Loukil, A., Berrached, N.: Dorsal Hand Vein Image Contrast Enhancement Techniques. International Journal of Computer Science Issues 11(1), pp. 137–142 (2014)
10. Faundez-Zanuy, M., Mekyska, J., Font-Aragonés, X.: A new hand image database simultaneously acquired in visible, near-infrared and thermal spectrums. Cognitive Computation 6(2), pp. 230–240 (2014)
11. Jain, A.K., Bolle, R.M., Pankatni, S.: Biometrics: Personal Identification in Networked Society. Kluwer Academic Publishers, Dordrecht (1999)
12. MacGregor, P., Welford, R.: Veincheck: imaging for security and personnel identification. Adv. Imaging 6(7), pp. 52–56 (1991)
13. Wang, L., Leedham, G., Siu-Yeung, C.D.: Minutiae feature analysis for infrared hand vein pattern biometrics. Pattern recognition, 41(3), pp. 920–929 (2008)
14. Morales-Montiel, I.I., Olvera-López, J.A., Martín Ortiz, M., Orozco-Guillén, E.E.: Hand Vein Infrared Image Segmentation for Biometric Recognition. Research in Computing Science 80, pp. 55–66 (2014)

Elaboración de una ontología para apoyar el diseño de secuencias didácticas basadas en competencias en la práctica del docente de educación media superior

Carmen Cerón Garnica, Etelvina Archundia Sierra, Beatriz Beltrán Martínez, Patricia Cervantes Márquez, José Luis Galindo Cruz

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Puebla,
México

{mceron,etelvina,bbeltran}@cs.buap.mx,cervantes.patty@gmail.com

Resumen. La web semántica ha contribuido en la educación mediante la creación de ontologías para la conceptualización y modelado del conocimiento de temas como recursos de aprendizaje y procesos de aprendizaje. El propósito de este artículo es presentar el diseño y construcción de una Ontología para apoyar el diseño de secuencias didácticas con un enfoque en competencias que utiliza un docente de educación media superior. En esta investigación se presentan los resultados obtenidos en la construcción de la ontología, al utilizar la herramienta Protégé y una prueba de aplicación. Finalmente se presenta las conclusiones y el trabajo a futuro de esta investigación.

Palabras clave: Ontología**,** planeación didáctica**, c**ompetencias, docente.

1. Introducción

En el 2008 se lleva a cabo la Reforma Integral de Educación Media Superior (RIEMS) estableciendo un nuevo plan curricular basado en competencias, un perfil del egresado y nuevo perfil del docente compuesto de ocho competencias para fortalecer su práctica docente [1]. Una de las competencias enfatiza que el docente "Planifica los procesos de enseñanza y de aprendizaje atendiendo al enfoque por competencias, y los ubica en contextos disciplinares, curriculares y sociales amplios. Esta planeación es para apoyar el desarrollo de las competencias en el estudiante y forma parte de sus actividades como docente.

Actualmente el uso de las Tecnologías de Información y Comunicación (TIC) en el campo de la educación son usadas como herramientas para apoyar procesos educativos centrados en el alumno y la utilización de la Web se ha integrado a distintas modalidades de educación como son semi-presencial y a distancia (bachillerato digital) siendo una gran ventaja para que los estudiantes tengan acceso a una gran cantidad de contenidos, información y actividades de aprendizaje de acuerdo a sus necesidades e intereses personales [2].

Una de las problemáticas que se han identificado en la Educación Media Superior es la necesidad de formar al docente en el enfoque educativo basado por competencias

para que puedan diseñar las secuencias didácticas de acuerdo a las competencias y propósitos de aprendizaje de la asignatura, lo cual ha producido confusión en la terminología de la planeación de las secuencias didácticas y de sus elementos que se deben definir para desarrollar las competencias disciplinares y genéricas. Por otra parte el docente tiene el interés de incorporar recursos educativos y las TIC para generar ambientes virtuales de aprendizaje que puedan utilizarse en la modalidad presencial que permitan enriquecer el desarrollo de las competencias en el alumno y los procesos de aprendizaje.

Las ontologías definen los términos básicos y las relaciones que comprenden el vocabulario de un tema en específico de alguna área, las reglas para combinar términos y las relaciones para definir entidades, clases, propiedades, predicados y relaciones entre estos componentes. La ontología es un sistema de representación del conocimiento que resulta de la selección de un ámbito o dominio del conocimiento, las ontologías se pueden organizar en estructuras jerárquicas, las cuales se consideran como una de las mejores formas de representar el conocimiento. Por eso la web semántica ofrece un área valiosa para avanzar en la construcción de una integración entre tecnología, contenidos y pedagogía mediante el uso de ontologías que permitan una mejor estructuración del dominio de la educación basada en competencias y sus relaciones con los objetivos educativos, el diseño de unidades de aprendizaje, recursos, evidencias y evaluación.

Esta investigación tiene como propósito el diseño y construcción de una Ontología para apoyar la Planeación de Secuencias Didácticas basada en Competencias en la práctica docente. Su aportación permitirá inferir conocimientos para apoyar las actividades del proceso de enseñanza-aprendizaje y fortalecer las competencias del perfil del docente de educación media superior.

Para la construcción de la ontología se utilizó la herramienta Protégé [3] versión 4.3.0, ya que además de ayudar a construir la ontología, proporciona información de referencia sobre el soporte para los datos manejados en las tecnologías semánticas, incluido el almacenamiento, la inferencia, y la capacidad de consultar los datos, es adaptable a la hora de trabajar con RDF Schema (RDFS) y Web Ontology Language (OWL) y lo cual permitió al final realizar una prueba de aplicación en Mysql y Java.
Por lo cual en este documento se organiza de la siguiente manera: En la sección 2 se presenta la fundamentación teórica y el trabajo de otros investigadores sobre las ontologías aplicadas a la educación, de la educación basada en competencias y elementos de una secuencia didáctica. En la sección 3 se define el análisis y diseño de la ontología y se muestran la implementación de la ontología al utilizar la herramienta Protégé. En la sección 4 Se presenta la implementación de una prueba de aplicación en NetBeans IDE 7.0 Apache Tomcat 7.0.11 como Servidor web y Mysql como gestor de base de datos. Finalmente se presentan las conclusiones y el trabajo a futuro de esta investigación.

2. Marco teórico

2.1. Ontologías aplicadas a la educación

En las últimas décadas el diseño y construcción de las ontologías han tomado relevancia en el mundo de la Ingeniería del Conocimiento, la Inteligencia Artificial y en las Ciencias de la Educación.

2.1.1. Definición de ontología

Para definir la Ontología partimos desde dos perspectivas filosóficas y de la Inteligencia Artificial. Desde la perspectiva filosófica la Ontología es una palabra que deriva del griego "ontos" (estudio del ser) y "logos" (palabra), es una ciencia que trata de dar una explicación sistemática de la existencia de tipos de estructura, categorías de objetos, propiedades, eventos, procesos y relaciones en cada área de la realidad [4]. Una ontología es una especificación explicita y formal de una conceptualización compartida, donde dicha conceptualización es una vista simplificada y abstracta del mundo que deseamos representar mediante el control de un vocabulario definido y detallado. Desde el campo de la Inteligencia Artificial, podemos encontrar una de las definiciones es la de Gruber [5] una ontología "es una especificación explícita de una conceptualización", y además, señala que el conocimiento en las ontologías se formaliza a través de seis componentes: clases, atributos, relaciones, funciones, axiomas e instancias. Por otra parte para Guarino [6] una ontología "es un artefacto ingenieril constituido por un vocabulario específico para describir una cierta realidad, más un conjunto de supuestos explícitos concernientes al significado pretendido de las palabras del vocabulario" una ontología describe una jerarquía de conceptos relacionados por relaciones de subsunción; en los casos más sofisticados, se añaden axiomas para expresar otras relaciones entre conceptos y restringir la posible interpretación."

2.1.2. Elementos básicos de una ontología

Las ontologías están formadas de los siguientes componentes que servirán para representar el conocimiento de algún dominio en específico [7].

- **Conceptos:** que son las ideas básicas que intentan formalizar, estos conceptos pueden ser clases de objetos, métodos, planes, estrategias, procesos de razonamiento, etc.

- **Relaciones:** que representan la interacción y el enlace entre los conceptos del dominio. Suelen formar la taxonomía del dominio. Por ejemplo: subclase-de, parte-de, etc.

- **Funciones:** que son un tipo concreto de relación, donde se identifica un elemento mediante el cálculo de una función que considera varios elementos de la ontología. Por ejemplo, categorizar-clase, etc.

- **Instancias:** utilizadas para representar objetos determinados de un concepto.

- **Axiomas:** que son teoremas que se declaran sobre relaciones que deben cumplir los elementos de la ontología. Por ejemplo: *"Si X y Y son de la clase W, entonces X no es subclase de Y"*

2.1.3. Clasificación y características más representativas de las ontologías

Existen varios tipos de clasificación de acuerdo a Barry [4] hay cuatro tipos de ontologías en función de su alcance y posibilidad de aplicación siendo las siguientes:

- Ontología de la aplicación: usadas por la aplicación. Por ejemplo, ontología de procesos de producción, de diagnóstico de fallas, de diseño intermedio de barcos, etc.
- Ontología del dominio: específicas para un tipo de artefacto, generalizaciones sobre tareas específicas en algún dominio concreto del conocimiento. Por ejemplo, ontología del proceso de producción.
- Ontologías técnicas básicas: describen características generales de artefactos. Por ejemplo: componentes, procesos y funciones.
- Ontologías genéricas: describe la categoría de más alto nivel, describiendo conceptos generales (como tiempo, espacio, objeto, etc.).

Otra clasificación se basa en el grado de formalidad de la ontología y distingue tres tipos de ontologías: Ontología descriptiva, formales formal, lógica.
Por otra parte algunas de las características más representativas de las ontologías se mencionan las siguientes:

- Niveles de abstracción de las ontologías: Estos niveles de generalización o abstracción nos dan una topología de ontologías. La idea es caracterizar una red de ontologías con el uso de multiplicidad y abstracción. Puesto que no podemos aspirar a tener una descripción completa del mundo, se puede pensar en una estrategia de construcción gradual de abajo hacia arriba.

- Multiplicidad de la representación: Un concepto puede ser representado de muchas formas, por lo que pueden coexistir múltiples representaciones de un mismo concepto.
- Mapeo de ontologías: Establecer relaciones entre los elementos de una o más ontologías, para establecer conexiones, especializaciones, generalizaciones, etc.
- Uso de Ontologías Múltiples: Esto permite realizar conceptualizaciones específicas.

2.1.4. Diseño de ontologías

Para el diseño de ontologías existen una serie de criterios de diseño y un conjunto de principios de utilidad para el desarrollo de ontologías [8]:

- Claridad y objetividad, que significan que la ontología debería proporcionar el significado de los términos definidos al proporcionar definiciones objetivas y también documentación en lenguaje natural.
- Completitud, esto es, que se prefiere una definición expresa da en términos de condiciones necesarias y suficientes a una definición parcial (por ejemplo, sólo en base a condiciones necesarias).
- Coherencia, para permitir inferencias consistentes con las definiciones.
- Extensibilidad monótona máxima. Significa que los términos nuevos o especializados deben ser incluidos en la ontología de forma que no requiera la revisión de las definiciones existentes.

- Compromiso ontológico mínimo. Se refiere a acordar el uso de una terminología compartida de forma coherente y consistente. Garantiza la consistencia, que no la completitud, de la ontología.
- Principio de Distinción Ontológica que significa que las clases en una ontología deberían ser disjuntas.
- Diversificación de jerarquías para aumentar la potencia proporcionada por los mecanismos de herencia múltiple.
- Modularidad para minimizar el acoplamiento entre módulos.

De acuerdo a la literatura podemos encontrar diferentes metodologías para diseñar y construir ontologías, se pueden distinguir metodologías para construir ontologías desde cero, metodologías para construir ontologías a través de procesos de reingeniería y metodologías para la construcción cooperativa de ontologías.
Para esta investigación se selecciona una metodología para construir ontologías desde cero como es la METHONTOLOGY [7]. Esta metodología se usa para construir ontologías tanto partiendo desde cero como reusando otras ontologías, o a través de un proceso de reingeniería. La construcción de ontologías a nivel de conocimiento incluye:

1. Identificación del proceso de desarrollo de la ontología donde se incluyen las principales actividades (evaluación, gestión, etc.).
2. Un ciclo de vida basado en prototipos evolucionados.
3. La metodología propiamente dicha, que especifica los pasos a ejecutar en cada actividad, las técnicas usadas, los productos a obtener y cómo deben ser evaluados.

Esta metodología ha sido usada en la construcción de múltiples ontologías, como una ontología química, ontologías hardware y software, etc. Las etapas son: (1) Especificación, (2) Conceptualización, (3) Formalización, (4) Implementación y (5) Mantenimiento. Esta metodología está parcialmente soportada por el entorno de desarrollo ontológico Protégé, WebODE y OntoEdit.

Para la implementación las técnicas de programación orientadas a objetos están siendo más comunes, debido a que su representación en términos de clases, atributos de las clases, objetos y la jerarquía de la herencia de clases, ha influido a un número de lenguajes y esquemas que se utilizan para la representación de conocimiento digital. Ontología es la teoría de objetos en términos de criterios, que nos permiten distinguir entre diferentes tipos de objetos y sus relaciones, dependencias y propiedades [5].

2.1.5. Utilidad de las ontologías en la educación

Las ontologías favorecen la comunicación entre personas, organizaciones y aplicaciones porque proporcionan una comprensión común de un dominio, de modo que se eliminan confusiones conceptuales y terminológicas, siendo en el campo de la educación muy útiles, principalmente en la educación a distancia.

En la literatura se pueden encontrar diversas investigaciones que tienen por objetivo la creación de ontologías del estudiante o del contexto para entornos de aprendizaje soportados por computadoras. Otras investigaciones enfatizan que se han creado las ontologías para escenarios colaborativos para el aprendizaje y que se han realizado con estudiantes universitarios usando el sistema DEGREE [9] y AT [10], los resultados de estos trabajos muestran que los escenarios de aprendizaje colaborativos se describen en términos de personas con metas de aprendizaje, estructuras de grupos, herramientas disponibles, roles que se asumen al realizar las tareas y restricciones en el uso del sistema (todos dentro de un dominio y contexto particular). Pramitasari y su equipo de investigadores [11] exponen una ontología del modelo del estudiante para personalización en sistemas de e-learning, los componentes centrales de la misma son el rendimiento del estudiante y el estilo de aprendizaje del mismo.

El diseño de la ontología en esta investigación es ofrecer una herramienta al docente que le apoye en simplificar el diseño de la planeación de secuencias didácticas y ayude a la recuperación de los elementos que la integran, así como información recursos digitales que pueda usar de acuerdo a las necesidades de aprendizaje.

2.2. Generalidades de la educación basada en competencias (EBC) y planificación de secuencias didácticas

Según Argudín [12] considera que este enfoque de EBC es holístico, trata de integrar las experiencias de la vida real, con un propósito de desarrollar habilidades mediante tareas y funciones básicas donde se domine los conocimientos, habilidades, actitudes y valores que determinan el logro de la competencia.

Son varios autores y organizaciones como la UNESCO y ANUIES y autores como Tobón, Argudín, Chomsky, Boyatzis, Marelli, que han propuesto sus definiciones. Para Tobón [13] define las competencias "como procesos complejos de desempeño integral con idoneidad en determinados contextos, que implican la articulación de diversos saberes, para realizar actividades y/o resolver problemas con sentido de reto, motivación, flexibilidad, creatividad y comprensión, dentro de una perspectiva de mejoramiento continuo y compromiso ético" y tienen tres dimensiones las competencias: "la cognitiva (Conocimiento, habilidades cognitivas) Habilidades procedimentales y técnicas (Hacer, actuar) actitudinal y afectivo motivacional (Actitudes y valores) se puede afirmar que estas dimensiones se activan al mismo tiempo, con procesos internos para demostrar la competencia adquirida o desarrollada.

En la EBC se definen claramente las competencias a desarrollar en el estudiante y se establecen objetivos de aprendizaje medibles. Este modelo educativo está enfocado en el éxito de los estudiantes, en la demostración del aprendizaje y en alcanzar el nivel definido de competencia [14].

En la literatura existen varias definiciones de "secuencia didáctica" aportadas por distintos autores como son: Zabala, Frade, Fans y Tobón, pero encontramos que siempre se refieren al conjunto de actividades encaminadas para los aprendizaje en los alumnos. Tobón la define como "…conjuntos articulados de actividades de aprendizaje y evaluación que con la mediación de un docente, buscan el logro de determinadas metas educativas, considerando una serie de recursos" [13].

Por lo cual el modelo de planificación de secuencia didáctica propuesto tiene los siguientes elementos Figura 1:

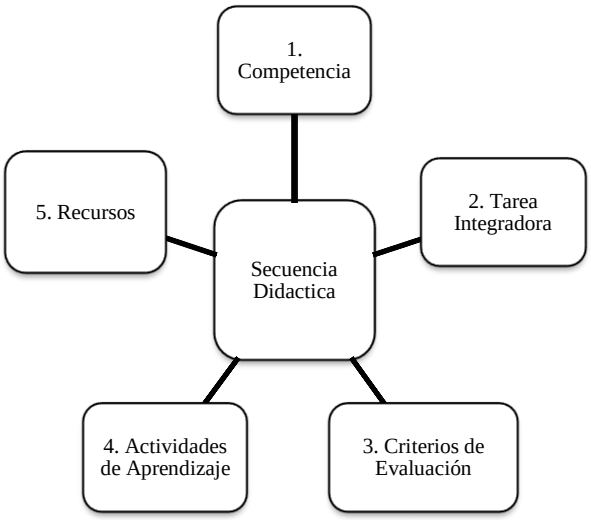


Fig. 1. Elementos Secuencia Didáctica según Tobón.

3. Construcción de la ontología para el apoyo del diseño de la secuencia didáctica y uso de recursos educativos digitales

3.1. Pasos para la creación de una ontología

Al crear una ontología, se hace explícita la categorización de elementos y relaciones que intervienen en un modelo de conocimiento. Los pasos que se usan en general:

(1). Definir el dominio y alcance de la ontología. Esto es definir el dominio que cubrirá la ontología, definir el uso la aplicación final de dicha ontología. (2). Enumerar términos importantes en la ontología. Definir una lista de los términos que se quieren almacenar en la ontología, así como de sus propiedades y las relaciones que existen entre ellos. (3). Definir las clases y la jerarquía de clases. Esto es definir la estructura conceptual del dominio, siguiendo el método top-down. (4). Definir las propiedades de las clases, definir la estructura interna de los conceptos. (5). Creación de instancias. Creación de instancias de clases individuales en la jerarquía.

Para lo cual se utilizó la metodología Methontology para determinar el modelo ontológico para definir los conceptos, relaciones y reglas que rigen el dominio. El proceso de creación de la ontología semántica se realizó a partir del análisis de cada uno de los conceptos identificados con respecto saberes de las competencias y elementos de las secuencias didácticas en educación media superior. Una vez se identificaron las tripletas, se definió cada uno de los objetos para hacer claridad en el hecho que representa cada tripleta, y ayudar a determinar si se acepta o se elimina como se muestra algunas tripletas que se obtuvieron como se muestra en Tabla 1.

Tabla 1. Tripletas identificadas para la Ontología.

Sujeto	Predicado	Objeto	Descripción
Estudiante	Cursa	Asignatura	Pertenece al Plan Curricular
Asignatura	tiene	Competencias	Conjunto de saberes Conocimientos (Saber), habilidades (saber-hacer) y actitudes (saber-ser) que serán evaluadas.
Secuencia _Didáctica	tiene	Etapas	Inicio, desarrollo y cierre.
Tarea _ Integradora	tiene	Escenario _ Aprendizaje	Son escenarios del contexto de las contenidos/unidades.
Etapa	tiene	Actividades Aprendizaje	Son definidas por el docente para alcanzar competencias.
Actividad_ Aprendizaje	tiene	Estrategias_ Aprendizaje	Son métodos y técnicas para propiciar el aprendizaje
Evaluación_ Competencias	tiene	Criterios de desempeño	Son los niveles dominio del logro de la competencia.
Evaluación_Act	Tiene	Evidencias	Son productos de forma individual y/o grupal que muestran la solución de Actividades.
Actividad_ Aprendizaje	tiene	Recursos	Son recursos educativos digitales y herramientas de comunicación.
Docente	dirige	Estudiante	Aprendizaje centrado en el estudiante.
Escenario_ Aprendizaje	tiene	Unidades_ Aprendizaje	Son contenidos representado por objetos de aprendizaje.

Posteriormente se establecieron las relaciones, cada uno de estos conceptos tiene axiomas entre ellos que permiten establecer el dominio y comportamiento establecidos para el modelo ontológico de la secuencia didáctica. En la mayoría de los casos, los axiomas en la ontología solo expresan relaciones de (es-un) y (tiene). A continuación se definen algunas relaciones generales para las clases y las subclases siendo las siguientes:

1. Existe solo un clase de nivel 0, la cual no deriva de ninguna clase.
2. La clase de un mismo nivel n que pertenece a una misma clase X de nivel n-1 no se intersectan entre sí.
3. Una clase X de nivel n, pertenecen solamente a una clase Y de nivel n-1.

Por medio de los siguientes enunciados se establecen las relaciones entre conceptos que definen el dominio de acción. Algunos de los axiomas de la ontología se pueden ver en la Tabla 2 y Tabla 3.

Tabla 2. Axiomas de la Ontología del Diseño de Secuencias Didácticas.

Enunciado	Axiomas
Una asignatura tiene competencias y tiene contenidos y tiene criterios de evaluación y tiene evidencias de aprendizaje y tiene un nivel del logro	∀ x Asignatura (x) => tiene(x, Competencias) Λ tiene(x, Contenidos) Λ tiene (x, Criterios_Evaluación) Λ tiene (x, Evidencias_Aprendizaje) Λ tiene (x,Nivel_ Competencia)

de la competencia	
Una secuencia didáctica tiene etapas y tiene competencias disciplinares y genéricas y tiene actividades de aprendizaje y tiene estrategias de aprendizaje, tiene recursos y tiene evaluación de la actividad.	∀ x Secuencia_Didactica (x) => tiene (x, Etapas_Aprendizaje) Λ tiene (x, Competencias_Disciplinares) Λ tiene (x, Competencias_Genericas) Λ tiene (x, Actividades_Aprendizaje) Λ tiene (x, Estrategias_Aprendizaje) Λ tiene (x,Recursos) Λ tiene (x, Evaluacion_Act)
Las actividades de aprendizaje tienen estrategias de aprendizaje, tiene recursos y tiene evidencia de aprendizaje y tiene evaluación de la actividad.	∀ x Actividad_Aprendizaje (x) => tiene (x, Estrategias_Aprendizaje) Λ tiene (x, Recursos) Λ tiene (x, Evaluacion_Aprendizaje) Λ tiene (x, Evidencia) Λ tiene (x, Evaluación_act)
Los recursos tienen recursos digitales y herramientas de comunicación. Los recursos digitales como animaciones, videos, u objetos de aprendizaje	∀ x Recursos (x) => tiene (x, Recursos_Digitales) Λ tiene (x, Herramienta_comunicación) ∀ x Recursos_Digitales (x) => es_un (x, Recurso_Animación) V es_un (x, Recurso_Video) V es_un (x, Objeto_Aprendizaje)
Una actividad tiene descripción de la actividad y tiene alcance y tiempo y tiene aprendizajes previos.	∀ x Actividad (x) => tiene (x, descrpción_Actividad) Λ tiene (x, Alcance) Λ tiene (x, Tiempo)Λ tiene (x, Aprendizajes_Previos)
La tarea integradora tiene escenarios y tiene contenidos y tiene recursos.	∀ x Tarea_Integradora (x) => tiene (x, Escenarios_Aprendizaje) Λ tiene (x, Contenidos) Λ tiene (x, Recursos)
Una estrategia de aprendizaje tiene métodos y tiene técnicas y tiene recursos.	∀ x Estrategia_Aprendizaje (x) => tiene (x, Métodos) Λ tiene (x, Técnicas)) Λ tiene (x, Recursos)

Tabla 3. Axiomas de la Ontología del Diseño de Secuencias Didácticas.

Enunciado	**Axiomas**
Una herramienta de comunicación puede ser un chat, o un correo electrónico, o un wiki, o un blog o un foro, o una videoconferencia.	∀x herramienta _comunicación (x) => es_un (x, chat) V es_un (x, correo_electronico) V es_un (x, wiki) V es_un (x, blog) V es_un (x, foro) V es_un (x, videoconferencia)
Una evidencia del aprendizaje tiene productos de trabajos. Los productos pueden ser reporte de la actividad o un de informe de trabajo o un proyecto de trabajo integrador.	∀ x Evidencia(x) => tiene (x, Producto) ∀x Producto (x) => es_un (x, reporte_actividad) V es_un (x, informe_trabajo) V es_un (x, proyecto_integrador)
Una estrategia de aprendizaje que se puede utilizar es un aprendizaje basado en problemas o un caso de estudio o un aprendizaje basado en proyecto o un aprendizaje colaborativo.	∀ x Estrategia_Aprendizaje (x) => es_un (x, ABP) V es_un (x, _Caso_estudio) V es_un (x, ABProyecto) V es_un (x, Aprendizaje_Colaborativo)

La evaluación tiene criterios de evaluación, tiene evidencias y tiene instrumentos de evaluación	∀ x Estrategia_Aprendizaje (x) => es_un (x, ABP) V es_un (x, Caso_estudio) V es_un (x, ABProyecto) V es_un (x, AColaborativo)

El modelo ontológico obtenido a través de Methontology, se implementó utilizando el la herramienta del editor de ontologías Protégé-OWL (Protégé). Esto permite generar una base de conocimientos para diseño de Secuencias Didácticas. Como se muestra en la Figura 2.

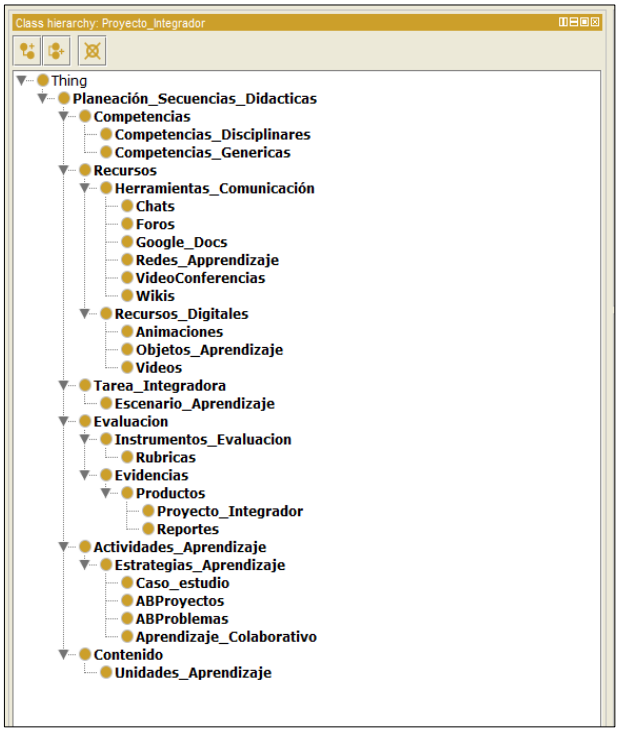


Fig. 2. Conceptos de la Secuencia Didáctica y propiedades.

4. Desarrollo y pruebas de la ontología

Al crear una ontología, se hace explícita la categorización de elementos y relaciones que intervienen en un modelo de conocimiento. Para implementar el prototipo, se analizaron algunas de las herramientas, y se seleccionó de acuerdo a las características necesarias con base en la información y el conocimiento adquiridos en las fases anteriores de la investigación. Las herramientas seleccionadas fueron: Protégé versión 4.3.0 para el diseño y construcción de la ontología, NetBeans IDE 7.0 para la creación y programación del prototipo utilizando como Servidor web Apache Tomcat 7.0.11 y MySql para la conexión con la base de datos.

La herramienta de Protégé, permite almacenar datos semánticos del modelo creado tiene un nombre y se refiere a tripletas almacenadas en el archivo “owl” generado por

la ontología. Se debe tener en cuenta que los datos semánticos se gestionan de manera más eficaz a través de la librería Jena-2-6-4 mediante Netbeans.

Para efectos de la prueba de la ontología, se enfocó a una aplicación específica de tareas de las búsquedas específicas, por lo cual se creó la base de datos en MySql con otras entidades como docente y alumno para poder realizar parte de relaciones (tripletas) y que son inferidas mediante la aplicación de las reglas del modelo, utilizando encadenamiento hacia adelante y reduciendo el tiempo de búsqueda cuando el usuario utiliza la ontología.

5. Conclusiones

Una de las principales contribuciones de este trabajo es el diseño de la Ontología para el Diseño de Secuencias Didácticas basadas en Competencias que permitirá recuperar información de forma más precisa y a partir de la realización se pudo comprobar que es factible representar el modelo de un diseño de una secuencia didáctica través de una ontología, ya que los mecanismos de representación que las mismas proveen resultan suficientes para modelar distintos tópicos en el campo de la educación.

La ontología de las Secuencias Didácticas basada en competencias permite identificar las necesidades reales de la planeación del proceso de aprendizaje, de tal forma que se puedan crear de manera sistemática cada una de las actividades de aprendizaje, garantizando una estructuración que permita la integración con los diferentes recursos digitales y el uso de herramientas de las Tecnologías de Información y Comunicación para que el docente pueda mejorar el perfil requerido en Educación Media Superior con respecto a las competencias pedagógicas y tecnológicas.

Una de las principales perspectivas de este trabajo es enriquecer las ontologías en el campo de la educación basada en competencias que permitan apoyar el diseño de sistemas en otros niveles educativos integrando nuevas tecnologías que logren tareas más específicas y puedan establecer relaciones con otros aspectos del aprendizaje (estilos de aprendizaje, estrategias de estudio, evaluaciones) por lo cual el trabajo a futuro es la evaluación de la ontología para determinar la calidad de la misma.

Referencias

1. Subsecretaría de Educación Media Superior. Reforma Integral de la Educación Media Superior en México, la Creación de un Sistema Nacional de Bachillerato en un marco de diversidad. SEP, México (2008). Disponible en: http://idbdocs.iadb.org/wsdocs/getdocument.aspx?docnum=38043188, http://redalyc.org/articulo.oa?id=99815899016. Fecha de consulta: 20 enero de 2014 (2014)
2. Coll, C.: Psicología de la educación y prácticas educativas mediadas por las tecnologías de la información y la comunicación. Una mirada constructivista, Revista Electrónica Sinéctica, pp. 1–24. Disponible en: http://redalyc.org/articulo.oa?id=99815899016. Fecha de consulta: 10 noviembre de 2014 (2004)
3. Protégé 4.3.0 Disponible en http://protege.stanford.edu/ Fecha de consulta: noviembre de 2014 (2014)

4. Smith, B.: Ontology and information systems, forthcoming in Stanford Encyclopedia of Philosophy. Recuperado de http://ontology.buffalo. edu/ontology(PIC).pdf. Fecha de consulta: 20 de enero 2015 (2003)
5. Gruber, T.: What is an Ontology? Disponible en http://www-ksl.stanford.edu/kst/what-is-anontology.html. Fecha de consulta: 20 de enero 2015 (2015)
6. Guarino, N.: Formal Ontology, Conceptual Analysis and Knowledge Representation. International Journal of Human-Computer Studies, 43, pp. 625–640 (1995)
7. Fernandez, L., Gomez-Perez, A., Jurista, N.: METHONTOLOGY: From Ontological Arts Towards Ontological Engineering. In: Symposium on Ontological Engineering of AAAI, Stanford University, pp. 33–40 (1997)
8. Lozano, A.: Ontologías en la Web Semítica. In: Cuadernos en Investigación en Ingeniería Informática, No. 5, Ingeniería Web: Nuevos retos Tecnológicos en la Era de la Información, Ediciones Servitec, pp.7–11, Oviedo, España (2001)
9. Barros, B., Verdejo, M.: Analysing students interaction process for improving collaboration: The DEGREE approach. International Journal of Artificial Intelligence in Education (2000)
10. Barros, B., Verdejo, F., Read, T., Migozuchi, R.: Applications of Collaborative Learning Ontology. In: Proc. of the 2nd MICAI2002, Yucatan, Mexico, pp. 301–310 (2002)
11. Pramitasari, L., Hidayanto, A., Aminah, S., Krisnadhi, A., Ramadhani, M.: Development of Student Model Ontology for Personalization in an e-Learning System based on Semantic Web. Disponible en http://citeseerx.ist. psu.edu/viewdoc/download?doi=10.1.1.301.3689&rep=rep1&type=pdf. Fecha de Consulta: 25 de enero 2015 (2015)
12. Argudín, Y.: Educación Basada en Competencias. Ed. Trillas, México (2005)
13. Tobón, S., Pimienta, J., García, J.: Secuencias didácticas: Aprendizaje y Evaluación de Competencias México. Pearson-Prentice Hall (2010)
14. Peñalosa, E.: Estrategias docentes con tecnologías. Pearson Educación de México, México (2013)

Hacia la definición de una representación para líneas de razonamiento

José Alfonso del Carmen Garcés-Báez

Benemérita Universidad Autónoma de Puebla,
Facultad de Ciencias de la Computación, Puebla,
México

agarces@cs.buap.mx, alfonso.garcesb@gmail.com

Resumen. Son más conocidos los métodos para la representación del conocimiento, tales como redes semánticas, el cálculo de predicados y el cálculo de situaciones que sirvió para introducir el problema del marco en Inteligencia Artificial [3] que los esquemas para representar razonamiento. En este trabajo se define una forma de representar el razonamiento o el plan a seguir para resolver un problema tal y como lo hacemos de forma cotidiana. La estructura que aquí se presenta es una quíntupla que contiene una condición que detona la valoración de evidencias para llevar a cabo ciertas acciones encaminadas al logro de un objetivo. La idea de la presente propuesta, bio-inspirada, es que permita hacer fácil y reducida la programación de tareas con base en la estructura *LR* que se presenta.

Palabras clave: Representación, conocimiento, razonamiento, inteligencia artificial, objetivo, evidencia y programación.

1. Introducción

Una de las tareas importantes que tiene la lógica, es la de encontrar un lenguaje común no sólo a gente de diferentes especialidades sino de diferentes idiomas y culturas. Norbert Wiener dijo en relación a una reunión con diferentes especialistas en México [6]:

Las discusiones eran interesantes y, en realidad, si aprendimos a hablar más o menos en el lenguaje de cada uno; pero existían grandes obstáculos que impedían una comprensión completa. Estas dificultades semánticas residían en el hecho de que en general no existe otro lenguaje que pueda sustituir la precisión de las matemáticas, y de que gran parte del vocabulario de las ciencias sociales es y debe ser empleado para la expresión de cosas que aún no sabemos expresar en términos matemáticos...me he dado cuenta en muchas otras ocasiones, de que uno de los principales deberes del matemático, al fungir como consejero de científicos de campos menos precisos, es el de desanimar a dichos hombres de ciencia de esperar demasiado de las matemáticas...

La experiencia de Wiener en 1944 sigue siendo la experiencia de muchos grupos interdisciplinarios de trabajo en la actualidad y en este sentido proponemos una forma

de representar la solución de problemas considerando el entendimiento y utilización en diversas áreas del conocimiento.

Son frecuentes los casos en los que ejecutamos una *acción* dependiendo del cumplimiento o no de un suceso (*valoración* de *evidencias)* en un momento determinado (*condición en el tiempo*), con el propósito de alcanzar un *objetivo*. Así mismo, no podemos dejar de considerar la posibilidad de que para la solución de un problema puede ser necesaria la solución previa de otros problemas que pudieran ser resueltos uno tras otro o al mismo tiempo.

El modelo que se presenta en este trabajo es recursivo y tiene los elementos que caracterizan a las estrategias de "divide y vencerás" [1]:

a. Divide.- Si el problema lo amerita puede dividirse en "problemitas".
b. Resuelve.- Se busca alcanzar los objetivos particulares.
c. Combina.- Las soluciones parciales ayudan a resolver los objetivos más generales.

2. Desarrollo

2.1 Estado del arte

Son numerosos los esfuerzos realizados en lo que se refiere a la representación del conocimiento y considero que en menor cantidad aquéllos dedicados a la representación del razonamiento por lo cual no debemos perder de vista lo que menciona Brachman [5]:

A widely recognized goal of artificial intelligence (AI) is the creation of artifacts that can emulate humans in their ability to reason symbolically, as exemplified in typical AI domains such as planning, natural language understanding, diagnosis, and tutoring.

Un avance especializado del uso de la representación del razonamiento y del conocimiento se encuentra en [7]:

KnowLang is a formal language providing a comprehensive specification model that must be able to address all the aspects of an ASCENS Knowledge Corpus and eventually some of the ASCENS Knowledge Base Inference Engine (ASCENS – Autonomic Service-Component Ensembles. 2010. http://www.ascens-ist.eu/) .

Every ASCENS Knowledge Corpus is structured into a domain specific ontology [5], logical framework and inter-ontology operators. The domain-specific ontology gives a formal and declarative representation of the knowledge domain in terms of explicitly described domain concepts, individuals and the relationships between those concepts/individuals:

- *facts – define true statements in the knowledge domains that can be used to discover situations;*
- *x rules – express knowledge such as: 1) if H than C; or 2) if H than C1 else C2; where H is hypothesis of the rule and C is the conclusion of the rule;*

- *x constraints – used to validate knowledge, i.e., to check its consistency. Can be positive or negative and express knowledge of the form: 1) if A holds, so must B; or 2) if A holds B must not.*

Por su parte, la programación lógica ha tenido importantes avances que facilitan la implementación de soluciones como se puede ver en [8]:

… At the same time, the fact that the typical applications of logic programming frequently involve irregular computations, make heavy use of dynamic data structures with logical variables, and involve search and speculation, makes the techniques used in the corresponding parallelizing compilers and run-time systems potentially interesting even outside the field.

2.2 Hacia una definición de LR

En [2] se encuentran algunos antecedentes del presente trabajo.

La estructura *LR* representa la posible solución a un problema *P* y se define mediante la quíntupla siguiente:

$$[Cond(t_0), v(e, s), a, o, m],$$

donde:

Cond(t_0) contiene una función booleana, muchas veces en términos del tiempo. También puede contener la inicialización de parámetros previamente a la *Condición* correspondiente.

- *v(e, s)* es una función de entero que valora la(s) evidencia(s), para el cumplimiento o no, del suceso *s* cuando *Cond(t_0)* es verdadera.
- *a* es la acción que se realizará en caso de que *v(e, s)* sea positivo.
- es el objetivo a lograr, puede ser una meta o la impresión de un texto.
- *m* es un arreglo opcional de *P´s* que puede ser vacío o puede verse como una matriz, esta opción se activa en caso de que *v(e, s)* sea negativo, refiriéndose a la posible necesidad de resolver uno o una secuencia de problemas previos.

Es necesario señalar que el valor de las evidencias que nos permite tomar una decisión es relativo y configurable ya que por claridad y conveniencia a veces debemos decidir teniendo evidencias con valor positivo o con valor negativo, es decir, si el valor de las evidencias es igual a lo que esperamos entonces decimos que la evaluación fue positiva en otro caso decimos que fue negativa.

Con respecto al arreglo *m*, se ha dicho que tiene la misma estructura que *P* y podrá tener las configuraciones siguientes:

a. m puede ser vacía (m = []). Esto significa que al cumplirse *Cond(t_0)*, se evaluarán las evidencias y de ser positivas se realizará la acción *a*, si las evidencias son negativas el procedimiento termina o, si se prefiere, se enviará una leyenda como una señal que ayude a la toma de decisiones o nos obligue a un nuevo acuerdo.

En este caso el tiempo total (T) para resolver el problema es igual a t_0 ($T = t_0$).

b. m puede estar compuesta de un solo elemento:

$$[\ [Cond(t_1), v(e_1, s_1), a_1, o_1, []]\ ].$$

Lo cual quiere decir que de ser negativa la valoración de la(s) evidencia(s) en el nivel anterior para realizar la acción *a*, es necesario realizar primero la acción a_1 y por lo tanto se requiere valorar e_1 con respecto a s_1 agotado el tiempo t_1 ($v(e_1, s_1)>0$) para alcanzar el objetivo o_1.

En este caso el tiempo total que tardamos en tomar la decisión para ejecutar la acción *a* es:

$$T = t_0 + t_1$$

c. m, con un renglón, se podrá interpretar como un conjunto de acciones en serie:

$$[$$
$$[Cond(t_{11}), v(e_{11}, s_{11}), a_{11}, o_{11}, [Cond(t_{12}), v(e_{12}, s_{12}), a_{12}, o_{12},$$
$$[\ldots[Cond(t_{1n}), v(e_{1n}, s_{1n}), a_{1n}, o_{1n}, []]\ldots]]]$$
$$]$$

Y en este caso el tiempo total que tardamos en tomar la decisión para realizar la acción *a* es:

$$T = t_0 + t_{11} + \ldots + t_{1n}$$

las $a_{i's}$ son acciones particulares y los $o_{i's}$ son objetivos particulares.

d. m podrá ser interpretado como un conjunto de elementos en paralelo:

$$[$$
$$[Cond(t_{11}), v(e_{11}, s_{11}), a_{11}, o_{11}, [\ldots[Cond(t_{1n}), v(e_{1n}, s_{1n}), a_{1n}, o_{1n}, []]\ldots]],$$
$$\ldots$$
$$[Cond(t_{m1}), v(e_{m1}, s_{m1}), a_{m1}, o_{m1}, [\ldots[Cond(t_{mn}), v(e_{mn}, s_{mn}), a_{mn}, o_{mn}, []]\ldots]]$$
$$]$$

Ahora, el tiempo total que tardamos en tomar la decisión de realizar la acción *a* es:

$$T = t_0 + \max\{t_{11}, \ldots, t_{m1}\} + \ldots + \max\{t_{1n}, \ldots, t_{mn}\}$$

2.3 Aplicaciones

1. Operaciones básicas

Substracción de dos números naturales a través de la suma:

$$R =$$
$$[(a=Rdm(100), b=Rdm(100),\ If\ a \neq b);\ v(e, a > b);\ c = 0,\ While\ (a \geq b+c):c{+}{+};$$
$$Print\ a, \text{“-“}, b, \text{”=”}, c;\ R]$$

- Se generan dos números aleatorios naturales entre 1 y 100, diferentes.
- Se valora la única evidencia ($a > b$). Si es positiva se ejecuta la *acción (While)*, si es negativa la estructura se llama a sí misma.
- Se termina con la impresión del resultado.

2. Sensores y acciones.

[(Sensor_humo_OK), v(e,"Incendio"), Activar: alarma y aspersores, "Dar protección",[]].

Si un sensor detecta cierta densidad de humo, se valoran las evidencias de incendio y en caso de ser positiva la valoración se activa la alarma y los aspersores con el objetivo de dar protección a las personas. Si la valoración de evidencias es negativa no procede acción alguna.

3. Cierre de operaciones de cobranza diaria.

Utilizando la estructura podemos indicar que:

"A las 20:00 hrs. se iniciará la verificación de programas en ejecución (procesos) y cuando todas las terminales hayan concluido sus operaciones se iniciarán los trabajos de cierre diario de la cobranza".

[(System_time = 20:00),v(e, Off_term_all), Close_job.batch, "Concluido el cierre diario de la cobranza",[]].

4. Venta de una casa.

Supongamos que queremos hacer la venta de un bien inmueble en un tiempo determinado, si el valor de la evidencia es diferente a lo que esperamos, es decir negativa, para ejecutar la acción requerimos nuevas evidencias y la valoración de las mismas que pudieran reunirse en forma seriada o paralela con acciones y objetivos particulares con la intención de obtener evidencias positivas que nos lleven a una valoración global positiva y como consecuencia la correspondiente ejecución de la acción principal.

Para indicar que deseamos tener la documentación completa en 25 días de un inmueble y poder buscar cliente para la venta, lo expresamos como:

[(Cond(25_días)), v(e, tener_doc), buscar_cliente, venta_casa,[]].

Si *v(e, tener_doc) > 0*, el valor de las evidencias coinciden con lo que esperamos. Eso quiere decir que "como máximo en 25 días fue posible tener toda la documentación necesaria y podemos buscar cliente para la venta de una casa"

Si *v(e, tener_doc) < 0*, hubiese sido necesario replantear la(s) acción(es) para lograr el objetivo, por ejemplo, de la siguiente manera:

[Cond(25_días), v(e, tener_doc), buscar_cliente, "venta de casa",
[
[Cond(20-días), v(e, tener_doc), tramitar, "eliminación de embargo",[]] $_{11}$
[Cond(7_días), v(e, tener_doc), pagar_serv, "pagos al corriente",[]] $_{21}$
[Cond(15_días), v(e, tener_doc), conseguir_ presupuesto, "gastos notaría",
[Cond(10_días), v(e, tener_doc), trámitar, "liberación de hipoteca", []] $_{32}$*]* $_{31}$
]

Como podemos darnos cuenta, nos vemos obligados a identificar sucesos para reunir nuevas evidencias en ciertos periodos, definir nuevas acciones y objetivos que se pueden ver en algunos casos como componentes de los principales y en otros como ajenos pero con un objetivo general común.

Ahora hemos agregado:

- Tener la documentación para el trámite de eliminación de un embargo en 20 días.
- Tener la documentación para el pago de los servicios y estar al corriente en 7 días.
- Tener la documentación para hacer un presupuesto de gastos notariales y cubrir el requisito de notaria en 15 días, para esto es necesario tener la documentación para el trámite de eliminación de hipoteca que tardará otros10 días.

Ejemplo para el cálculo del tiempo total en días:

T=25+ | 20 | 0 |
| 7 | 0 |
| 15 | 10 |

25 + max{20, 7, 15} + max{0, 0, 10} = 55 días

De un estimado inicial de 25 días pasamos a un estimado de 55 considerando más actividades por realizar. Se ha mencionado que, por la estructura recursiva de *LR*, la ejecución de tareas se puede hacer en serie o en paralelo, iniciando con una "raíz", de tal suerte que el trazo de la ejecución se puede interpretar como una estructura irregular multidimensional tipo fractal donde cada nodo tiene la misma estructura.

2.4 Relación entre *LR* y lenguajes de programación

La forma de escribir la quíntupla *LR* es natural y entendible para las diversas áreas del conocimiento. Una línea de la estructura se tendrá que traducir en más instrucciones en el lenguaje de programación que se tenga al alcance y dicha labor será transparente para quién use la estructura. Como una forma de aproximar la semántica de *LR*, se presenta el ejemplo que sigue.

Recordemos la primera aplicación de *LR* referente a operaciones básicas (1), donde:

$$R = [(a=Rdm(100), b=Rdm(100), \ If\ a \neq b);\ v(e, a > b);\ c = 0,$$
$$While\ (a \geq b+c):c{+}{+};\ Print\ a,"-",b,"=",c;\ R]$$

Esta instrucción se traduce en el siguiente programa en Python Ver. 2.7.6 (http://www.python.org):

```
#                  Programa LR
#     Substracción de dos números naturales
# aleatorios entre 1 y  100 a través de la suma
#                                        GB-JAC
#
# Primer parámetro
def Condicion(a, b):
      if (a != b):
            return 1
      elif (a == 0):
            return 0
#
# Segundo parámetro
```

```
def Verificacion(e, (a, b)):
      if (e==1):
            if (a >b):
                  return 1
            elif (a < b):
                  return -1
#
# Tercer parámetro
def Accion(a, b):
      c=0
      while (a > (b+c)):
            c=c+1
      return c
#
# Cuarto parámetro
def Objetivo(a, b, c):
      print a,"-",b,"=", c
#

# Quinto parámetro
# y procedimiento recursivo
#
def R():
  a, b = random.randint (1, 100), random.randint (1, 100)
  if (Condicion(a, b)):
        if (Verificacion(1, (a, b)) > 0):
              c=Accion(a, b)
              Objetivo(a, b, c)
        elif (Verificacion(1, (a, b)) < 0):
              print a,"-",b," ¡No es un numero natural!"
              R()
```

Casos de prueba:

1. LR.R()
   64 - 70 ¡No es un numero natural!
   62 - 40 = 22

2. LR.R()
   64 - 82 ¡No es un numero natural!
   25 - 92 ¡No es un numero natural!
   69 - 65 = 4

3. LR.R()
   88 - 49 = 39

Con el programa anterior se ejemplifica el uso de todos los parámetros que componen la estructura *LR* incluyendo su propiedad recursiva en los casos en que la *Verificación* de *evidencias* resulta negativa.

3. Conclusiones

Es necesario especificar con mayor precisión la semántica de cada uno de los parámetros que integran la quíntupla *LR* para llegar a la definición formal y a las demostraciones necesarias que fundamentan la misma.

Un trazo de la estructura multidimensional mostrará las posibilidades de uso de los patrones que puede generar *LR*.

La estructura *LR* es independiente del lenguaje de programación que se utilice y el uso de la misma se vislumbra funcional y de relativa facilidad.

La representación para líneas de razonamiento *LR*, es una propuesta para expresar tareas o procedimientos de diversas áreas de conocimiento, en particular, tiene gran potencial en las áreas de planificación como se presenta en [4], en robótica mediante el uso de Robot Operating System (http://www.ros.org/), en la plataforma JdeRobot (www.jderobot.org/) para la programación de Drones y en aplicaciones de las tecnologías emergentes, como las móviles, aprovechando la amplia gama de sensores disponibles (http://developer.android.com/tools/device.html).

Referencias

1. Cormen, Thomas H.: Introduction to Algorithms. MIT (2009)
2. Garcés Báez, JAC.: Interrelación de algunas lógicas intermedias y Answer Set. Tesis de Maestría, FCFM-BUAP (2002)
3. Nilsson, N.J.: Inteligencia Artificial: Una nueva síntesis. McGraw-Hill (2001)
4. Russell, S., Norvig, P.: Inteligencia Artificial: Un enfoque moderno. Prentice Hall (1996)
5. Brachman, R.J., Levesque, H.J.: Knowledge representation and reasoning. Elsevier, San Francisco (2004)
6. Ensayos Científicos. 3ª. Edición de Ciencia y Desarrollo. CoNaCyT (1982)
7. Vassev, E.: Requirements and Initial Model for KnowLang – A Language for Knowledge Representation in Autonomic Service-Component Ensembles. In: C3S2E´11: Proceeding of The Fourth International C* Conference on Computer Science and Software Engineering (2011)
8. Gupta, G.: Parallel execution of prolog programs a survey. Transactions on Programming Languages and Systems (TOPLAS), Vol. 23, Issue 4 (2011)

Desarrollo de un sistema para medir similitud entre clases

R. Guzmán-Cabrera, J.C. Ruiz, M. Torres-Cisneros

University of Guanajuato, Engineering Division, Irapuato-Salamanca, Guanajuato, México

guzmanc@ugto.mx

Resumen. El incremento continuo de información en formato digital obliga a contar con nuevos métodos y técnicas para acceder, recopilar y organizar estos volúmenes de información textual. Una de las técnicas más utilizadas para organizar dicha información es la clasificación de documentos. Los sistemas de clasificación automática de textos tienen una baja eficiencia cuando las clases son muy parecidas, y en este caso es muy importante el poder identificar aquellos atributos que nos permiten separar una clase de otra. En este trabajo se presenta un sistema para generar gráficas de similitud entre documentos pertenecientes a clases de un corpus dado, tarea previa al proceso de clasificación automática. Estas gráficas son utilizadas como un método de refinamiento auxiliándose de las similitudes entre los documentos no clasificados. Con esto se busca poder anticipar el desempeño de un método de clasificación automática. Los resultados obtenidos permiten ver la viabilidad de la metodología propuesta.

Palabras clave: Clasificación de documentos, similitud.

1. Introducción

En la actualidad el almacenamiento de contenido digital se ha vuelto más abundante y menos costoso. Esto ha provocado que la cantidad de información digital generada por compañías de diferentes rubros crezca a una gran velocidad, generando de esta forma, grandes repositorios de conocimiento. Sin embargo, esto ha provocado la necesidad de crear técnicas para poder clasificar de manera automática estos volúmenes de datos [1].

En el caso de clasificación documentos de texto, los documentos son convertidos de su contenido original, a arreglos de información los cuales representan el contenido de esos documentos. Una de las técnicas más utilizadas para la representación de los documentos es la de usar la característica de frecuencia de aparición de una palabra o frase en el documento [1]. Además de la necesidad de ejemplos para el entrenamiento [2], un problema recurrente al que se enfrenta un clasificador, es el de la similitud entre los documentos de diferentes clases, este problema consiste en qué tan parecido es un documento de una clase a con respecto de una clase b. La similitud, por lo general está representada por una escala numérica entre 0 y 1, donde 0 representa que no existe similitud alguna y 1 representa que el documento es el mismo. Sean di y dj documentos

representados de la forma $d_j = (w_{1j}, w_{2j}, ..., w_{mj})$, entonces algunas medidas de similitud o distancia se definen a continuación:

El coeficiente de similitud de Jaccard mide la similitud entre dos conjuntos de muestras. Aunque originalmente fue utilizado para comparar tipos de flores en un ecosistema [3], ha tenido buena aceptación en el campo del análisis de documentos [4].

$$\boldsymbol{jaccard}(\boldsymbol{d_i}, \boldsymbol{d_j}) = \frac{\sum(\boldsymbol{w_{ik}} \times \boldsymbol{w_{jk}})}{(\sum \boldsymbol{w_{ik}^2} + \sum \boldsymbol{w_{jk}^2}) - (\sum \boldsymbol{w_{ik}} + \sum \boldsymbol{w_{jk}})}$$

El coeficiente de Dice determina la similitud entre dos documentos pesa- dos dando importancia a los atributos de la intersección [5].

$$\boldsymbol{Dice}(\boldsymbol{d_i}, \boldsymbol{d_j}) = \frac{\boldsymbol{2}\sum(\boldsymbol{w_{ki}} \times \boldsymbol{w_{kj}})}{\sum \boldsymbol{w_{ki}} + \sum \boldsymbol{w_{kj}}}$$

La medida coseno es una de las más populares para determinar la similitud de los documentos. El objetivo es determinar el ángulo entre dos vectores, en este caso los vectores de representación de los documentos [6].

$$\boldsymbol{Coseno}(\boldsymbol{d_i}, \boldsymbol{d_j}) = \frac{\sum \boldsymbol{w_{ki}} \times \boldsymbol{w_{kj}}}{\sqrt{\sum \boldsymbol{w_{ki}^2}} \times \sqrt{\sum \boldsymbol{w_{kj}^2}}}$$

2. Desarrollo

En la figura 1 se muestra el proceso que realizará el programa en un esquema general.

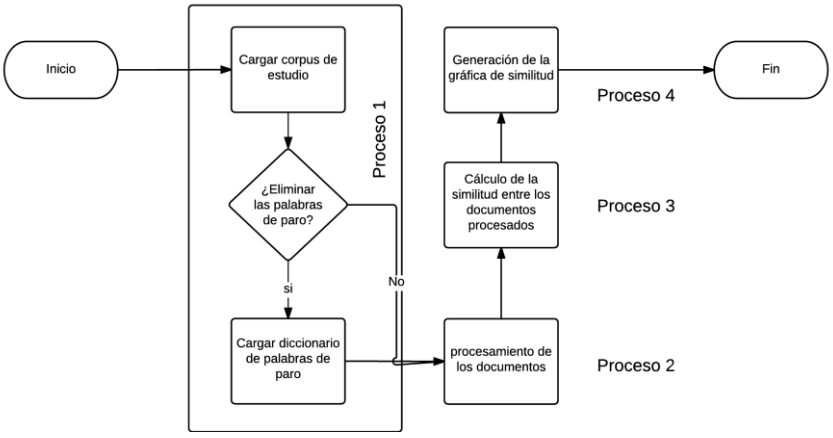


Fig. 1. Diagrama de flujo del proceso del programa.

En la figura 2, se muestra la ventana principal de la aplicación.

Una función adicional de la ventana principal, es el desplegar el histograma de frecuencias referente a cada clase o a cada documento, este módulo se muestra ante el evento de dar doble clic en alguna de las clases o alguno de los documentos. En la figura 3 se muestra un histograma de palabras a manera de ejemplo. El módulo de

visualización de histogramas de frecuencia, tiene como finalidad entregar una asistencia visual al usuario para el análisis de las tablas de frecuencias entregadas por el programa. Cuenta con la operación de acercar/alejar y funciona tanto para mostrar los histogramas de las clases, como de los documentos únicos.

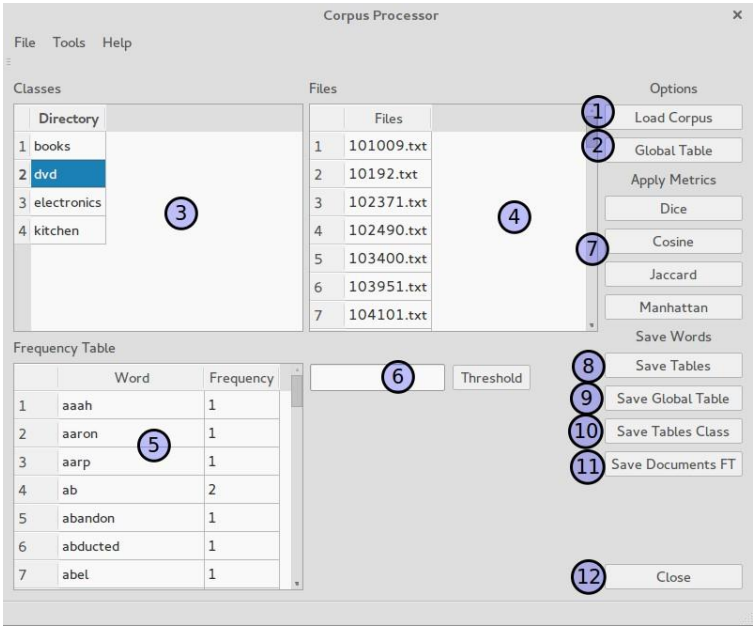


Fig. 2. Ventana principal de la aplicación.

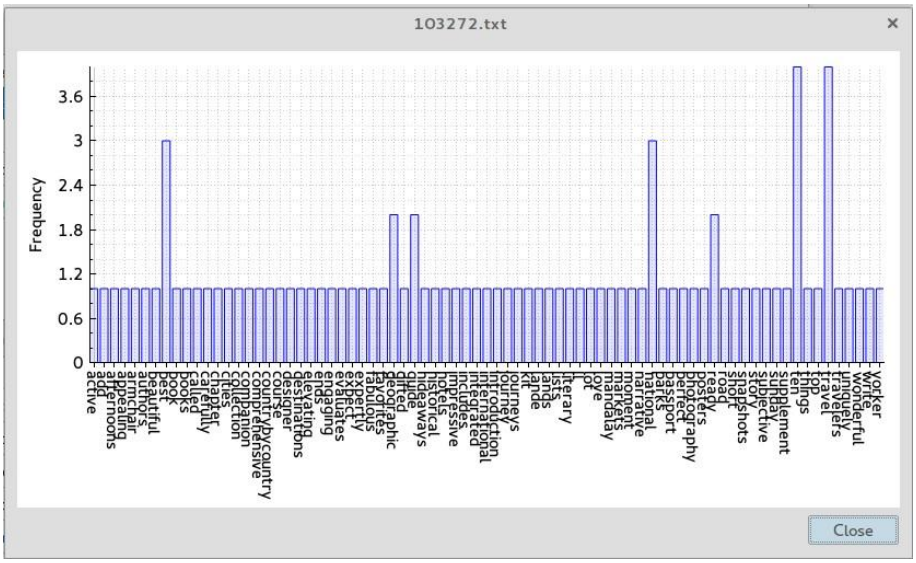


Fig. 3. Módulo de visualización de histogramas.

3. Pruebas y resultados

El corpus de estudio utilizado para las pruebas es el "Multi-Domain Sentiment Dataset (version 2.0)" [7]. El corpus está compuesto de opiniones de 4 diferentes categorías de productos tomados de la base de datos de Amazon [8]. Las cuatro categorías seleccionadas son:

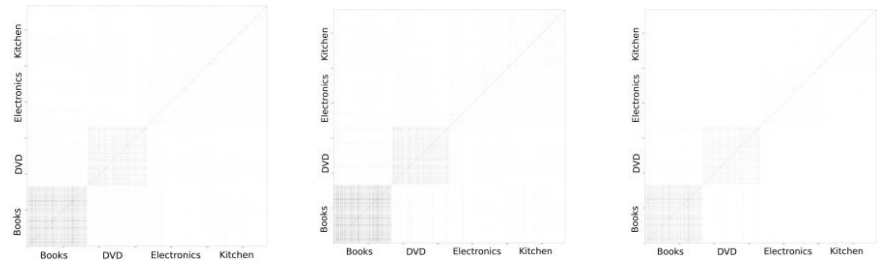
- Libros: 1463 opiniones positivas y 1039 opiniones negativas.
- DVD'S: 1391 opiniones positivas y 1396 opiniones negativas.
- Artículos de cocina: 832 opiniones positivas y 922 opiniones negativas.
- Electrónicos: 948 opiniones positivas y 1014 opiniones negativas.

Para la realización de las pruebas se creó un conjunto de documentos compuesto de 1600 archivos (400 archivos en cada clase). Este conjuntos a su vez, se subdividió en un conjunto de entrenamiento (70 % del conjunto original) y un conjunto de prueba (30 % del conjunto original) los cuales se utilizaron para realizar el proceso clasificación automática de documentos. Los resultados de similitud obtenidos para este conjunto se muestran en la figura 4.

	books	dvd	electronics	kitchen
kitchen	0.0681353	0.142873	0.420928	1
electronics	0.0500725	0.153189	1	0.420928
dvd	0.110637	1	0.153189	0.142873
books	1	0.110637	0.0500725	0.068135

	books	dvd	electronics	kitchen
kitchen	0.139786	0.176653	0.430826	1
electronics	0.124711	0.218162	1	0.430826
dvd	0.13653	1	0.218162	0.176653
books	1	0.13653	0.124711	0.139786

	books	dvd	electronics	kitchen
kitchen	0.0352692	0.0769325	0.266567	1
electronics	0.0256792	0.0829481	1	0.266567
dvd	0.0585581	1	0.0829481	0.076932
books	1	0.0585581	0.0256792	0.035269



(a) Dice (b) Coseno (c) Jaccard

Fig. 4. Similitud entre clases resultante de cada métrica utilizando un umbral de frecuencia de 2.

Entre mayor es el valor del umbral de la frecuencia, mayor es la eliminación de palabras en los documentos, llegando a un punto en el cual comienza a existir pérdida de información; por lo que se espera que el clasificador actúe mejor cuando cuenta con una cantidad de información relevante sin caer en la pérdida de ésta.

Para las pruebas de clasificación se utilizó el software WEKA de la Universidad de Waikato [9] el cual provee de diferentes técnicas de clasificación. Los métodos de clasificación seleccionados para las pruebas fueron, Naive Bayes y Máquinas de Vectores de Soporte.

En la tabla 1 se presentan los resultados obtenidos por los clasificadores bajo diferentes umbrales de frecuencia para la prueba con 1600 documentos. Como se puede apreciar, la precisión del clasificador disminuye, conforme aumenta la cantidad de palabras descartadas por el umbral de frecuencia (TH).

Tabla 1. Resultados clasificación de 1,600 documentos.

Clasificador	Entrenamiento	Prueba	Precisión	Recuerdo	F-Measure
Naive Bayes Original	280	120	0.834	0.827	0.826
SVM Original	280	120	0.822	0.802	0.806
Naive Bayes TH2	280	120	0.725	0.684	0.691
SVM TH2	280	120	0.735	0.697	0.705
Naive Bayes TH3	280	120	0.739	0.602	0.619
SVM TH3	280	120	0.747	0.623	0.642
Naive Bayes TH4	280	120	0.734	0.604	0.603
SVM TH4	280	120	0.76	0.619	0.618

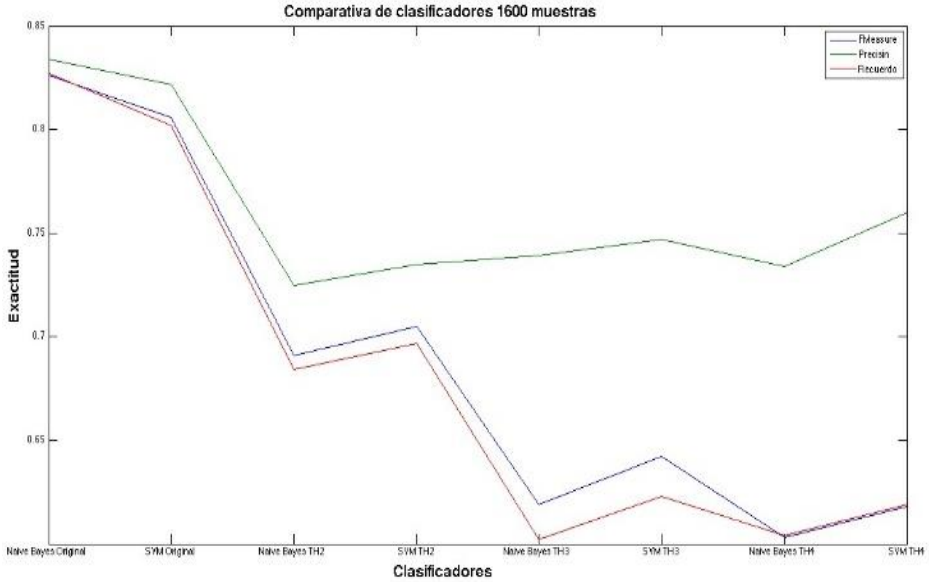


Fig. 5. Exactitud de los clasificadores para 1,600 documentos.

5. Conclusiones y trabajo futuro

Con el desarrollo de este proyecto se demostró que el resultado presentado por las gráficas de similitud, nos permite saber de una manera previa, cómo se comportará el clasificador. A su vez se pudo apreciar cómo el filtrado de información mediante un

umbral de frecuencia puede contribuir a un mejor desempeño, siempre y cuando no se llegue a un grado donde exista una pérdida de información relevante.

Para las métricas de Cosenos, Dice y Jaccard se puede apreciar que la métrica con mejores resultados fue la de Cosenos al mostrar una mejor nitidez ante las otras bajo el mismo conjunto de información.

Referencias

1. Chen, Z., Ni, C., Murphey, Y.L.: Neural network approaches for text document categorization. In: International Joint Conference on Neural Networks (IJCNN'06), pp. 1054–1060 (2006)
2. Nigam, K., McCallum, A.K., Thrun, S., Mitchell, T.: Text classification from labeled and unlabeled documents using EM. Mach. Learn. 39(2-3), pp. 103–134 (2000)
3. Jaccard, P.: Étude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin del la Société Vaudoise des Sciences Naturelles 37, pp. 547–579 (1901)
4. Niwattanakul, S., Singthongchai, J., Wanapu, S.: Using of Jaccard coefficient for keywords similarity. In: Proceedings of the International MultiConference of Engineers and Computer Scientists, Vol. 1, IMECS, March 13-15, Hong Kong (2013)
5. Salton, G., McGill, M.: Introduction to Modern Information Retrieval. McGraw Hill (1983)
6. Frakes, W.B., Baeza-Yates, R.: Information Retrieval: Data structures and Algorithms. Englewod Cliffs, Prentice Hall (1992)
7. Multi-domain sentiment dataset (version 2.0). http://www.cs.jhu.edu/mdredze/datasets/ sentiment/
8. Amazon site. http://www.amazon.com
9. Weka 3: Data mining software in java. http://www.cs.waikato.ac.nz/ml/weka/

Impreso en los Talleres Gráficos
de la Dirección de Publicaciones
del Instituto Politécnico Nacional
Tresguerras 27, Centro Histórico, México, D.F.
octubre de 2015
Printing 500 / Edición 500 ejemplares