EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

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

Research in Computing Science

Vol. 152 No. 7
July 2023

Research in Computing Science

Research in Computing Science, Año 22, Volumen 152, No. 7, julio de 2023, es una publicación mensual, editada por el Instituto Politécnico Nacional, a través del Centro de Investigación en Computación. Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, Ciudad de México, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor responsable: Dr. Grigori Sidorov. Reserva de Derechos al Uso Exclusivo del Título No. 04-2019-082310242100-203. ISSN: en trámite, ambos otorgados por el Instituto Politécnico Nacional de Derecho de Autor. Responsable de la última actualización de este número: el Centro de Investigación en Computación, Dr. Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Fecha de última modificación 01 de julio de 2023.

Research in Computing Science, year 22, Volume 152, No. 7, July 2023, is published monthly by the Center for Computing Research of IPN.

Volume 152(7)

Advances in Artificial Intelligence

Iris Méndez (ed.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2023

ISSN: in process

Formerly ISSNs: 1870-4069, 1665-9899

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

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

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

ISSN 1870-4069

Table of Contents

Page

Generación de un corpus lingüístico digital en español enfocado a la depresión 5
César-Jesús Núñez-Prado, Claudia Talavera Ortega, Liliana Chanona-Hernández, Grigori Sidorov

Generación automática de descripciones taxonómicas de plantas usando la representación objeto-atributo-valor 13
Bernardo Serrano-Estrada, José Luis Villaseñor, Enrique Ortiz, Miguel Murguía-Romero

Reconocimiento y detección de señales de tráfico mexicanas de tipo preventivas, restrictivas e informativas aplicando YOLOv4 25
Daniela Bolaños-Flores, Hamurabi Gamboa-Rosales, José M. Celaya-Padilla, Tania A. Ramirez-del Real

Análisis de sentimientos en críticas de cine utilizando SVM y combinación de n-gramas 39
Cesar Alexis Estrada Palacios, José Luis Tapia Fabela

Sistemas de clasificación aplicados a la detección de paráfrasis 49
Francisco Fernando López Ponce, Gerardo Sierra, Gemma Bel Enguix

Clasificación automática de preguntas en español en el dominio de comercio electrónico usando una red neuronal convolucional 63
Melissa A. De-León-Barrón, Ana B. Rios-Alvarado, Jose L. Martinez-Rodriguez

Análisis de las características más importantes para la detección de noticias falsas en Español 77
Sergio Damián, Hiram Calvo, Alexander Gelbukh

Agente de navegación web para detección de noticias falsas usando aprendizaje profundo por refuerzo y listas de argumentos 87
Alexis Fernando Aguilera Valderrama, Iván Vladimir Meza Ruiz

Diseño de un módulo para la detección de ciberbullying en la red social twitter utilizando lenguaje natural en una aplicación móvil android implementado en un entorno universitario 101
Lissete Rosete Rosas, Luis Ángel Reyes Hernández, Beatriz Alejandra Olivares Zepahua, Ignacio López Martínez, Laura Angélica Décaro Santiago

Diferenciación de la región macular dentro de la retina utilizando operaciones morfológicas simples orientado al prediagnóstico en etapas tempranas de retinopatía diabética 115
Emanuel de-la-Cruz-Espinosa, Rita Q. Fuentes-Aguilar, Eduardo Morales-Vargas

Identificación automática de divulgadores de noticias falsas mediante el perfilado de autor 127
Cesar Macias, Miguel Soto, Hiram Calvo, José E. Valdez-Rodríguez

Modelos 3D de reconstrucción facial: Una breve revisión 139
Victor Hernández-Manrique, Miguel González-Mendoza, Carlos Vilchis, Mauricio Méndez-Ruiz, Carmina Pérez-Guerrero

Derecho comparado asistido por computadora: Procesamiento de lenguaje natural legal 149
Héctor Alejandro Vargas-Gutiérrez, Gerardo Rodriguez-Hernandez

Análisis y clasificación de tweets contextuales de desastres 163
Tania Alcántara, Omar García-Vázquez, Hiram Calvo, Marco A. Cardoso-Moreno

Análisis de sentimientos de textos de Twitter utilizando aprendizaje profundo 177
Jessica Olivares L., Abraham Sánchez L., Rogelio González V., Abraham Maldonado G.

Selección de características de representaciones de texto de BETO usando un algoritmo genético 191
Juan José Guzmán-Landa, José Clemente Hernández-Hernández, Guillermo de Jesús Hoyos-Rivera, Efrén Mezura-Montes

Elderly Mortality from COVID-19 in Mexico City: A Computational Intelligence Approach Based on Random Forests 203
Sinuhe Mazuti Osorio-Rivero, Guillermo Molero-Castillo, Everardo Bárcenas, Rocío Aldeco-Pérez

ISSN 1870-4069

Generación de un corpus lingüístico digital en español enfocado a la depresión

César-Jesús Núñez-Prado[1,2], Claudia Talavera Ortega[1], Liliana Chanona-Hernández[1], Grigori Sidorov[2]

[1] Instituto Politécnico Nacional,
Escuela Superior de Ingeniería Mecánica y Eléctrica,
México

[2] Instituto Politécnico Nacional,
Centro de Investigación en Computación,
México

{cesar.jnprado, claudiatalaveraor, lchanona}@gmail.com,
sidorov@cic.ipn.mx

Resumen. La depresión es un desorden mental que afecta a miles de personas alrededor del mundo y que de no ser detectada a tiempo puede llevar a un desenlace mortal como el suicidio. Este tipo de trastorno es considerado como un asesino silencioso debido a que no es de fácil identificación ya que los individuos con esta clase de desorden generalmente intentan ocultarlo, por lo que cualquier tipo de investigación que provea herramientas para su detección temprana siempre será de gran utilidad. Por otra parte, hoy en día, las redes sociales son una fuente ininterrumpida de información que puede ser analizada con la intención de reconocer patrones asociados a la resolución de alguna tarea de investigación, como por ejemplo el análisis de emociones. En esta investigación buscamos crear un corpus (banco de información) que contenga mensajes publicados en la red social *Twitter* y cada mensaje estará asociado con dos posibles clasificaciones: mensaje depresivo o mensaje no depresivo.

Palabras clave: Depresión, análisis de emociones, reconocimiento de patrones, corpus.

Compilation of a Digital Textual Corpus in Spanish Focused on Depression

Abstract. Depression is a mental disorder that affects thousands of people around the world and if it is not detected in time, it can lead to a fatal outcome such as suicide. This type of disorder is considered a silent killer because it is not easy to identify, since individuals with this type of disorder generally try to hide it, so any type of research that provides tools for early detection will always be very useful. On the other hand, nowadays, social networks are an uninterrupted source of information that can be analyzed with the intention of recognizing patterns associated with the resolution of some research task, such as the analysis of emotions. In this research we seek to create a corpus (information bank) that

contains messages published on the Twitter social network and each message is associated with two possible classifications: depressive message or non-depressive message.

Keywords: Depression, analysis of emotions, recognizing patterns, corpus.

1. Introducción

De acuerdo con el psicoanalista Erich Seligmann Fromm[1], en el momento en el que el ser humano se separó de la naturaleza a través de percibir su propia autoconciencia, se dio paso a la generación de todo el espectro de sentimientos conocidos, entre los que se encuentran; el amor, el odio, la tristeza, la euforia, la envidia, el enojo, la impaciencia, la satisfacción, la culpa, la preocupación, entre algunas. Tales sentimientos repercuten de manera directa en el estado de ánimo y la manera conductual de las personas, es decir; sentimientos positivos serían asociados con conductas y estados de ánimo relacionados con la felicidad, mientras que, por otro lado, los sentimientos negativos son capaces de provocar disociaciones en los individuos.

Uno de los desórdenes mentales más relacionados con los sentimientos de impacto negativo es la depresión, [1] Fromm la define como la incapacidad de sentir alegría o tristeza, es opuesta a la razón debido a que impide el desenvolvimiento humano y por tanto, irracional ya que quien la presenta rehúye a experimentar ese tipo de sentimientos necesarios para que en un futuro pueda alcanzar un crecimiento completo.

La depresión es considerada como un trastorno mental común que afecta tanto a hombres como a mujeres, sin importar si se trata de un adolescente, adulto o adulto mayor y puede desembocar en problemáticas aún más grandes como lo pueden ser las autolesiones, tendencias suicidas o finalmente, cometer suicidio. Lamentablemente, las personas que sufren de este tipo de trastorno experimentan dificultades para poder expresar libremente y de manera cómoda sus sentimientos con otras personas, especialmente con sus familias y generalmente acuden por ayuda cuando el episodio depresivo ya es muy grave.

Según la Encuesta Nacional de Epidemiología Psiquiátrica (ENEP)[2], entre los años 2001 y 2002 se reportó que un 9.2 % de la población en México había padecido algún tipo de trastorno depresivo durante su vida y desafortunadamente esta cifra continúa incrementándose ya que en el año 2022 se publicó el 2º.Diagnóstico operativo de salud mental[3] y adicciones realizado por los Servicios de Atención Psiquiátrica (SAP) de la Secretaría de Salud en México, en donde se estima que la depresión es uno de los trastornos más frecuentes en la población que cuenta con derechohabiencia con más de 3 millones de casos confirmados.

La derechohabiencia indica que las personas tienen algún servicio de salud público como el IMSS o el ISSSTE. En algunas ocasiones, las personas que presentan tendencias depresivas o suicidas, se apoyan del uso de las redes sociales, ya que encuentran en ellas un medio para poder expresarse con confianza, de una manera libre

[1] Erich Seligmann Fromm fue un psicoanalista, psicólogo social y filósofo alemán.

[2] Véase: https://www.insp.mx/avisos/sintomas-depresivos-y-atencion-a-la-depresion

[3] Véase:https://www.gob.mx/salud/prensa/008-en-mexico-3-6-millones-de-personas-adultas-padecen depresion

y con menos prejuicios, dada la posibilidad de desarrollar una identidad digital y con ello poder mantenerse en el anonimato.

Para este proyecto, nosotros utilizaremos la red social *Twitter* debido a que la longitud de los mensajes no sobrepasa los 250 caracteres y ello permite realizar el procesamiento y clasificación de los mensajes en un tiempo menor, y también porque la cuenta de desarrollador de la red social, otorga los permisos necesarios para realizar la descarga de los mensajes publicados.

Consideramos que es de vital importancia el generar recursos lingüísticos enfocados a poder determinar, si algún texto publicado en redes sociales contiene un posible discurso depresivo o suicida, con la intención de poder utilizar a favor de la sociedad, el poder de cómputo que se sigue desarrollando hasta nuestros días y con ello alertar a las personas que sufren de este tipo de trastorno con la finalidad de reducir el índice de depresión y suicidio dentro de la sociedad.

La estructura de la presente investigación mostrará los trabajos relacionados, la aplicación de la metodología empleada para la generación del corpus, los resultados obtenidos, conclusiones y trabajo a futuro.

2. Trabajos relacionados

En esta sección se describirán algunas investigaciones a fin al objetivo principal de nuestro trabajo.

En [2] se define a un corpus como un conjunto de textos producidos en condiciones naturales, representativos de una lengua, almacenado en formato electrónico y codificado con la intención de ser analizados científicamente. Refiere que la condición natural se establece, cuando existe la intención real de comunicar una idea y no ser concebidos para ilustrar algún fenómeno lingüístico. Identifica que el término representativo se da cuando se respeta un momento determinado de la historia.

Los corpuses actuales contienen miles de millones de formas y el único modo en el que se puede recuperar esta cantidad de información es de manera electrónica. Dentro de la información que proporciona un corpus se puede encontrar a los metadatos, los cuales son información interna que incluye, el nombre de la fuente de información, fecha y lugar de la publicación, editorial, entre algunos.

En [3] crearon un corpus etiquetado para el español con la finalidad de realizar análisis sobre sentimientos. Decidieron realizar la descarga de mensajes publicados en la red social *Twitter* y para considerar que mensajes se procesarían, estos debían contar con por lo menos 5 palabras; para el etiquetado de las palabras utilizaron *Freeling* y excluyeron las palabras de parada con la lista que proporciona la biblioteca *Natural Language Toolkit (NLTK).* Con dichas especificaciones, generaron un corpus con más de 20 mil mensajes que fueron clasificados de acuerdo a los *hashtags*[4] contenidos en cada mensaje. Las emociones que identificaron fueron la alegría, asco, tristeza, ira, miedo y sorpresa.

Ya combinando la creación de un corpus y el tema de la depresión en [4] desarrollaron un corpus de mensajes de ideación suicida extraídos de la *web* y la *deep web* tanto en español (33 %), como en inglés (67 %). Algunas de las categorías con las

4 Hashtag es el símbolo «#».

que realizaron la clasificación de los textos incluye: depresión, ironía, tristeza, auto-pro-suicida e indefinido y el tamaño final del corpus generado contiene más de 7 mil *tokens*.

3. Aplicación de la metodología

En la siguiente sección se mostrará la metodología empleada dentro de la presente investigación.

3.1. Fase 1

En esta fase inicial se generó una cuenta regular de usuario de *Twitter*, en donde la red social solicitó información personal tal como; nombre completo, nombre de usuario (es el nombre que aparece visible para los demás usuarios de la red social), fecha de nacimiento, cuenta de correo electrónico y número de teléfono celular. Una vez que se llenaron todos los campos requeridos, la red social envió un correo electrónico de verificación y con ello se completó con éxito la apertura de la cuenta.

3.2. Fase 2

Para poder acceder a la información disponible en la red social es indispensable contar con una cuenta de desarrollador desde la *API* de *Twitter* (interfaz de programación de aplicaciones de *Twitter*), para solicitarla se debe generar una aplicación y agregar la información de los campos solicitados entre los que se incluyen:

- Perteneces a una institución educativa o a una empresa (para instituciones educativas no hay costo, pero se limita la cantidad de información disponible).
- ¿Cuál es el uso que le darás a la información?
- ¿Qué clase de algoritmos aplicarás?
- ¿Qué resultados esperas obtener?

La apertura de la cuenta de desarrollador no es automática, el personal de *Twitter* se pone en contacto con el cliente en busca de obtener información más detallada con un intercambio de correos electrónicos, que en nuestro caso duró casi 15 días (cabe mencionar que todos los correos son enviados en inglés).

3.3. Fase 3

De manera paralela, mientras se estaba llevando a cabo la fase anterior, se realizó una búsqueda en internet de las palabras más utilizadas y asociadas a la tristeza, depresión y suicidio. Se encontraron 3 listas de las cuales se aplicó la unión de los elementos entre ellas para generar una lista con entradas sin repetir. Sobre cada elemento de la lista final se aplicó la lematización (el cual es el proceso de encontrar las palabras sin flexionar, por ejemplo; el lema de la palabra «abrumado» es «abrumar») para obtener la normalización de las entradas. En la Tabla 1 se presenta una muestra de la lista de palabras depresivas.

Tabla 1. Muestra de la lista de palabras depresivas.

Morir	Abrumar	Miseria
Tristeza	Suicidar	Odiar
Terapia	Torturar	Terminar

3.4. Fase 4

La cuenta de desarrollador proporciona 4 claves secretas, únicas y renovables para poder realizar la conexión entre diferentes lenguajes de programación y *Twitter*. Nosotros desarrollamos un código fuente en *Python*[5] y apoyándonos de la biblioteca *Tweepy*[6] utilizamos las claves para realizar la conexión y con ello poder acceder a la información disponible en la plataforma, entre la que se encuentra:

- Texto publicado (con dos posibilidades, texto completo o texto parcial).
- Nombre del usuario.
- Fecha y hora de publicación.
- Ubicación de la publicación.
- Cantidad de «me gusta».
- Cantidad de comentarios en la publicación.
- Cantidad de «*retweets*[7]».

Una ventaja de utilizar *Tweepy* es que contiene un módulo de búsqueda especializada en el que se debe estipular la palabra a buscar (*query*), el periodo de tiempo (ejemplo: mayo 2022) y el número de mensajes a recuperar. Este último punto es muy importante porque la cuenta de desarrollador es académica y únicamente permite la descarga de 800 mensajes por periodos de 30 minutos, si esta cantidad se excede, entonces *Twitter* congela las credenciales y se debe esperar de 2 a 3 días para que las activen de nuevo.

Siendo cuidadosos con la cantidad de mensajes y los periodos de tiempo, nosotros estuvimos descargando mensajes durante 30 días, utilizando como palabras de búsqueda, las palabras de la lista de la Fase 3 y con ello se realizó la descarga de un poco más de 12 mil mensajes.

3.5. Fase 5

De igual forma, se llevaron de manera casi paralela la Fase 4 y la Fase 5. Los mensajes descargados de *Twitter* se escribieron en archivos de texto plano y a todos estos mensajes se les hizo un procesamiento automático el cual consistió en lo siguiente:

- Conversión del texto a minúsculas.
- Tokenización (es el proceso mediante el cual se obtienen las unidades mínimas de las oraciones, es decir; se separan las oraciones en palabras).
- Eliminación de los «*retweets*» para procesar mensajes únicos.

[5] Se utilizó la versión 3.9.12.

[6] Se utilizó la version 4.10.0.

[7] Un «retweet» es la acción de publicar en tu propio muro la publicación de otro usuario y se identifica por las letras «RE» al inicio de cada mensaje.

- Eliminación de enlaces a internet.
- Eliminación de imágenes.
- Eliminación de emoticones.
- Eliminación de signos de puntuación.
- Eliminación de caracteres numéricos.
- Lematización.
- Etiquetado de las palabras (part of speech tagging)[8].
- Eliminación de stop-words[9].
- Eliminación de nombres propios.
- Eliminación de mensajes repetidos.

Cabe mencionar que la gran diversidad de emoticones disponibles en internet representa un esfuerzo mayúsculo para el etiquetado manual y es por ello que se decidió eliminarlos de los mensajes descargados.

Una vez que se realizó este procedimiento sobre todos los mensajes descargados de *Twitter*, se creó un banco de datos con un total de 1,623 mensajes únicos.

3.6. Fase 6

En esta última fase, se pidió el apoyo de 5 psicólogas para realizar la evaluación de los mensajes, tomando como único criterio dos posibilidades, mensaje depresivo o mensaje no depresivo.

El perfil de las sicólogas cumplió los siguientes 3 aspectos:

- Experiencia mínima de 5 años ininterrumpidos.
- Atención a pacientes con depresión o tendencia suicida.
- Uso y conocimiento de redes sociales.

Cada sicóloga debía leer mensaje por mensaje y de acuerdo a su propia experiencia de consulta, debía clasificar cada uno de los mensajes. El periodo de revisión duró 2.5 meses.

Una vez que se contó con los archivos calificados, se inició el proceso de agregar los mensajes al corpus, para ello se consideró la siguiente métrica:

- Si por lo menos 3 sicólogas calificaron como mensaje depresivo al mensaje, se asignaba la clase 1.
- Si por lo menos 3 sicólogas calificaron como mensaje no depresivo al mensaje, se asignaba la clase 0.

En la Tabla 2 se muestran algunos mensajes descargados y etiquetados por los sicólogos. En la columna de la izquierda se visualiza el mensaje y en la columna de la derecha la etiqueta final de cada mensaje.

En la Tabla 3 se visualiza el proceso de clasificación de los mensajes, cada uno de los sicólogos evalúo cada uno de los mensajes y se realizó el conteo de los votos. Cuando un mensaje contenía por lo menos 3 votos de una clase, entonces se le asignó esa clase final al mensaje.

[8] Ejemplo de etiquetado: si aparece la palabra «correr» la etiqueta asociada es «verbo».

[9] *Stop-words* es el conjunto de palabras que incluyen las palabras gramaticales como los artículos definidos, indefinidos, preposiciones, etc.

Tabla 2. Muestra de mensajes clasificados

Mensaje	Etiqueta
Sonreír siempre ha sido más fácil que explicar por qué estoy triste, es mejor morir en silencio	1
Sé que estoy sólo incluso cuando me encuentro rodeado de tantas personas.	1
La depresión ha sido mi fiel compañera desde que te marchaste y sé que me acompañará hasta el fin de mi vida	1
Me siento tan triste porque mi América perdió, pero aun así estoy alegre de gritar al mundo que soy lgbt	0

Tabla 3. Muestra de las evaluaciones de los sicólogos

Eval 1	Eval 2	Eval 3	Eval 4	Eval 5	Clase
1	1	1	1	1	1
1	1	0	1	0	1
1	1	1	1	1	1
1	0	0	0	0	0

4. Conclusiones y trabajo a futuro

Con el desarrollo de esta investigación se pudo crear un corpus lingüístico digital enfocado a la depresión que cuenta con 1,623 mensajes con dos clasificaciones; 1 si el mensaje es depresivo y 0 si el mensaje no es depresivo y se encuentra almacenado en un archivo de texto plano.

El corpus se encuentra desbalanceado, ya que contiene 985 mensajes clasificados como no depresivos y 638 mensajes clasificados como depresivos. Esta herramienta se pondrá a disposición de investigaciones académicas de manera gratuita vía correo electrónico.

Como trabajo a futuro, estamos pensando en dos posibilidades a corto plazo, por una parte, es importante balancear el corpus, por lo que es necesario seguir descargando más mensajes y solicitar a nuevos sicólogos que nos aporten con su conocimiento y experiencia en la clasificación de los mensajes y, por otra parte, es importante aplicar métodos de inteligencia artificial sobre el corpus generado para saber si es posible que identifiquen un patrón a través de los mensajes.

Referencias

1. Moreno-López, S.: La condición humana según Erich Fromm. Pensamiento, Papeles de filosofía, vol. 3, pp 151–171 (2017)
2. Rojo, G. Introducción a la lingüística de corpus en español. New York: Routledge (2021)
3. Sidorov, G., Galicia, S. N., Camacho, V. A.: Construcción de un corpus marcado con emociones para el análisis de sentimientos en Twitter en español. Revista Escritos, BUAP, vol. 1, no. 1 (2016)

4. Zafra, S., Gómez, J. M., Navarro-Colorado, B.: Diseño, compilación y anotación de un corpus para la detección de mensajes suicidas en redes sociales. Procesamiento de lenguaje natural, vol. 59, pp. 65–72 (2017)
5. Zucco, C., Calabrese, B., Cannataro, M.: Sentiment analysis and affective computing for depression monitoring. In: IEEE International Conference on Bioinformatics and Biomedicine, pp. 1988–1995 (2017) doi: 10.1109/BIBM.2017.8217966
6. Mustafa, R., Ashraf, N., Shabbir, F., Fersund, J., Shahzad B., Gelbukh, A.: A multiclass depression detection in social media based on sentiment analysis. In: 17th International Conference on Information Technology - New Generations, vol. 1134, pp. 659–662 (2020) doi: 10.1007/978-3-030-43020-7_89
7. Ameer, I., Arif, M., Sidorov, G., Gómez-Adorno, H., Gelbukh, A.: Mental illness classification on social media texts using deep learning and transfer learning. In: 8th World Conference on Soft Computing (2022) doi: 10.48550/arXiv.2207.01012
8. Bird, S., Klein, E., Loper, E.: Natural language processing with Python (2019)

ISSN 1870-4069

Generación automática de descripciones taxonómicas de plantas usando la representación objeto-atributo-valor

Bernardo Serrano-Estrada[1], José Luis Villaseñor[2], Enrique Ortiz[2], Miguel Murguía-Romero[3]

[1] SERES Sistemas Especializados, Estado de México, México

[2] Universidad Nacional Autónoma de México, Instituto de Biología, Departamento de Botánica, México

[3] Universidad Nacional Autónoma de México, Instituto de Biología, Unidad de Informática para la Biodiversidad, México

miguel.murguia@ib.unam.mx

Resumen. La descripción de especies de plantas y animales es una actividad que realizan los botánicos sistemáticos en grupos específicos de seres vivos y son una fuente de información importante para la ciencia y fundamento del conocimiento científico de la biodiversidad. Objetivo. En este trabajo se presenta una herramienta informática para la generación de descripciones taxonómicas de plantas a partir de la información que el especialista registra en una matriz de datos. Método. Las características de las especies se especifican en una base de datos mediante tripletas objeto-atributo-valor (v.g., flores – color - blanco), en los que cada atributo es asignado a un grupo u objeto (v.g., Tallo, Hojas, Flores) que es usado para guiar la construcción automática de la descripción de acuerdo con una gramática ATN predefinida. Resultados. El sistema web AbaTax construido (www.abatax.abaco2.org) permite al especialista especificar una matriz de datos de un grupo de especies que es usada con dos finalidades: a) servir como una herramienta de identificación para personas no especialistas (sistema experto) y b) la generación automática de descripciones taxonómicas con una redacción muy cercana a la necesaria para su publicación científica. Conclusiones. AbaTax es un sistema web que permite la generación de descripciones taxonómicas útiles al no experto, para que pueda conocer mejor el grupo, y al especialista para su publicación científica.

Palabras clave: lenguaje natural, representación del conocimiento, botánica, Ageratina.

Automatic Generation of Plant Taxonomic Descriptions Using the Object-Attribute-Value Representation

Abstract. The description of plant and animal species is an activity carried out by systematic botanists in specific groups of living beings and is an important source of information for science

and the foundation of scientific knowledge of biodiversity. Objective. A computer tool for the generation of taxonomic descriptions of plants from the information that the specialist records in a data matrix is presented. Method. The species characteristics are specified in a database by object-attribute-value triplets (e.g., flowers - color - white), in which each attribute is assigned to a group or object (e.g., Stem, Leaves, Flowers) that is used to guide the automatic construction of the description according to a predefined ATN grammar. Results. The built AbaTax web system (www.abatax.abaco2.org) allows the specialist to specify a data matrix of a group of species that is used for two purposes: a) serve as an identification tool for non-specialists (expert system) and b) the automatic generation of taxonomic descriptions with wording very close to that required for scientific publication. Conclusions. AbaTax is a web system that allows the generation of taxonomic descriptions useful to the non-expert, so that they can better understand the group, and to the specialist for their scientific publication.

Keywords: Natural language processing, knowledge representation, botany, *Ageratina.*

1. Introducción

Hasta la fecha se han registrado 350,980 especies de plantas vasculares en el mundo, y se estima que aún faltan por descubrir cerca de 25% más [1]. En México se ha identificado la presencia de más de 25,000 especies de plantas vasculares [2]. El proceso de anunciar el descubrimiento de una nueva especie sigue un protocolo muy formal, que incluye la publicación en una revista científica de un artículo que contiene la información que describe y justifica esa novedad taxonómica, además de bautizarla asignando un nombre en latín, por ejemplo, *Agave salomonii*.

Una descripción taxonómica es una explicación detallada y ordenada de las características de la especie que permiten distinguirla de otras. Actualmente, el número de nuevas especies de plantas vasculares descritas por año para México ronda en las 100 especies; particularmente, en el año 2022 se publicaron 113 nuevas especies de plantas vasculares [3].

Las descripciones taxonómicas son un elemento indispensable en dos tipos de actividad científica: en el descubrimiento de una nueva especie y en la documentación de las plantas de una región en una "Flora". Las "Floras" son tratados que incluyen la lista de las plantas, así como las descripciones de los ejemplares que se recolectaron en esa región, por ejemplo, la Flora de Veracruz (v. gr. [4]).

Para que una nueva especie de planta sea reconocida por la ciencia se debe generar una publicación (llamada protólogo) en una revista científica que incluya diversos elementos, como dibujos de las partes de la planta, el nombre asignado y que debe ser en latín y la descripción taxonómica, entre otros.

Las descripciones taxonómicas de plantas son documentos que los botánicos sistemáticos deben generar ya sea en un número elevado, por ejemplo, cuando se involucran en la conformación de una Flora de una región, y en menor número, pero con alta precisión, en la documentación de una nueva especie.

Recientemente se han iniciado los esfuerzos para publicar la "Flora electrónica de México", un proyecto ambicioso en el que se deberán generar las descripciones taxonómicas de las más de 25,000 especies de plantas vasculares de México [5].

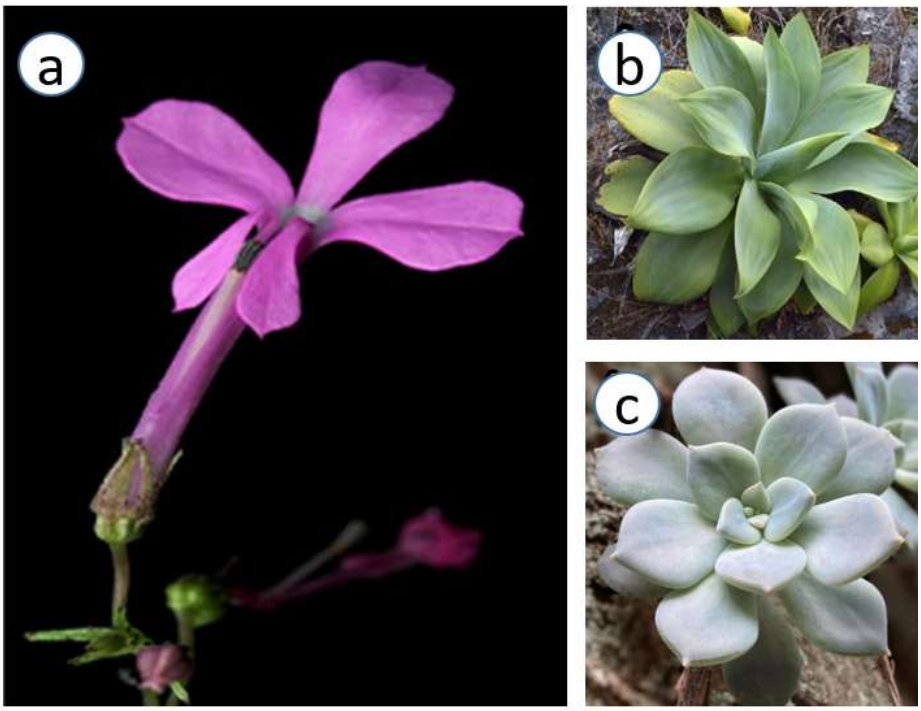


Fig. 1. Ejemplo de especies de plantas nuevas para la ciencia descritas en el año 2022. a) *Lobelia alanae*; b) *Agave rosalesii*; c) *Graptopetalum trujilloi.* Créditos: imágenes tomadas de a) Perez-Perez et al., 2022 [12]; b) y c) Sánchez Sánchez et al., 2023 [3].

1.1. Herramientas para la generación automática de descripciones taxonómicas

Los esfuerzos para generar de forma automática descripciones taxonómicas datan de finales de la década de los 70 y principios de los 80 [6, 7], con programas que procesaban un archivo de texto en un formato denominado DELTA, en el que se asocia una lista de taxones a un esquema jerárquico de los caracteres y los estados de carácter. El formato DELTA fue originalmente creado para especificar la matriz de datos para la generación de claves de identificación, ya sean dicotómicas para su impresión, o policlaves para su uso en la computadora.

Existen trabajos que describen el desarrollo de herramientas para la generación automática de descripciones taxonómicas con base en hojas de cálculo. Por ejemplo, se ha propuesto registrar los datos de cada especie en los renglones y los caracteres como nombres de las columnas, y en cada celda los estados de carácter para cada especie [8].

Esto tiene la desventaja de que en la descripción automática se incluyen todos los nombres de los atributos, resultando con frecuencia en una redundancia de palabras, lo que provoca que la descripción se aleje del lenguaje natural. Otra desventaja es que no se distinguen entre los conectores de conjunción o disyunción (y/o).

Otra alternativa para la generación de descripciones taxonómicas de plantas es el uso de herramientas genéricas. Una de las aplicaciones del método de generación de

descripciones automatizadas ha sido la venta de productos en línea, donde se utilizan palabras claves y una búsqueda en internet sobre el producto para tener más información y poder construir el texto con lenguaje natural.

Un ejemplo es neuraltext (www.neuraltext.com), cuyo sitio web utiliza una función de agrupación de palabras clave y encuentra varias sugerencias para cada término de búsqueda. Otro ejemplo es Quicktrools (tools.picsart.com) que utiliza el mismo método que neuralText, con la diferencia de que se puede elegir el "tono" de la descripción, como por ejemplo si será más profesional, amigable, relajado, persuasivo, etc. Sin embargo, estas opciones aún no generan descripciones con grado científico, con la precisión que se requiere en trabajos florísticos o taxonómicos.

El objetivo de este trabajo fue generar un sistema web que apoye al botánico en la generación automática de descripciones taxonómicas, con una redacción cercana al lenguaje natural que requieren las publicaciones de tratados florísticos y especies nuevas.

2. Representación del conocimiento mediante objeto-atributo-valor

La representación del conocimiento mediante tripletas objeto-atributo-valor (OAV) es un formalismo para la representación del conocimiento [10] y ha sido usada ampliamente desde hace varias décadas en los sistemas inteligentes (Van Melle, 1978; Waterman, 1978), principalmente para representar conocimiento declarativo. Ejemplos de tripletas OAV son:

- Hoja – margen – entero,
- Flor – color – roja,
- Fruto – tipo – cápsula.

La representación OAV puede verse como un caso particular de las redes semánticas. En la tripleta OAV, la relación objeto-atributo es del tipo pertenencia ("has a"), mientras que la de atributo-valor es del tipo presencia ("is a"). La representación OAV es una "versión" reducida de las redes semánticas, allí radica su alto poder de representación.

3. Uso de OAV en la generación de descripciones taxonómicas

3.1. OAV como representación del conocimiento taxonómico

Varios de los sistemas que permiten la generación automática de descripciones taxonómicas utilizan de forma implícita OAV para la representación de la información de los taxones o especies de plantas. En esos sistemas, la correspondencia OAV con los atributos botánicos es de tal forma que el objeto lo identifican con la especie de planta, el atributo con la característica y el valor con el estado de carácter. Ejemplos de ellos son:

- *Ageratina adenophora* – disposición de las hojas – opuestas,
- *Ageratina adenophora* – distribución – Michoacán,

- *Ageratina liebmannii* – disposición de las hojas – alternas,
- *Ageratina liebmannii* – distribución – Puebla.

En estas tríadas, el primer elemento, y que corresponde al objeto, es el nombre de la especie, el segundo (atributo) el carácter, y el tercero (valor) el estado de carácter.

Sin embargo, nosotros hemos identificado que una mejor forma de aplicar el OAV para la representación, con el objetivo de generar descripciones taxonómicas es asociar a las partes de la planta con los objetos, dejando a la especie fuera del triplete, o si se quiere ver así, creando una cuarteta:

Ageratina adenophora:
Hojas – disposición – opuestas,
Distribución – estado – Michoacán.
Ageratina liebmannii:
Hojas - disposición – alternas,
Distribución – estado – Puebla.

La correspondencia OAV con la matriz de datos informativos es entonces:

objeto ↔ grupo de caracteres
atributo ↔ carácter
valor ↔ estado de carácter

El objeto (o grupo de caracteres) se identifica generalmente con alguna parte de la planta, como "Hojas" o "Flor", pero también puede ser de otra naturaleza, como "Distribución".

3.2. Generación de descripciones taxonómicas

Con base en la representación OAV, se identificó una forma de generar descripciones taxonómicas en lenguaje natural. La adaptación del OAV para la generación de descripciones taxonómicas consistió en:

1. Identificar y clasificar los diferentes atributos asignándolos a un objeto. Por ejemplo, todas las características de la hoja pueden asignarse al objeto "Hojas".
2. Escribir el nombre del objeto como se desea que se incluya en la redacción de la descripción. Por ejemplo, "Hojas" y no "Hoja".
3. Escribir los nombres de los valores de los atributos en concordancia gramatical con el nombre del objeto. Por ejemplo, "opuestas" y no "opuesta" ni "opuesto", para que concuerde con "Hojas".
4. Los nombres de los atributos están implícitos en la descripción, por lo que sus textos funcionan para documentar la tripleta OAV, pero no se incluyen en la descripción taxonómica. Por ejemplo, "Hojas opuestas", pero no "Hojas con disposición opuesta".
5. El usuario establece de forma explícita el orden de los objetos, atributos y valores en la descripción, mediante un número.

Por ejemplo, si se tiene la siguiente representación OAV para una especie:

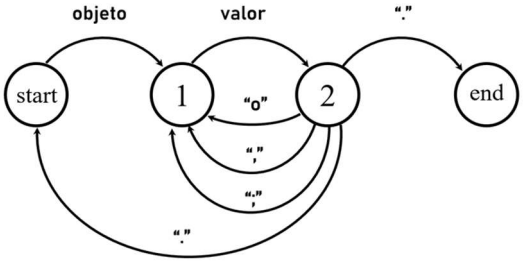


Fig. 2. Red de transición aumentada (ATN) de la gramática definida para la generación automática de las descripciones taxonómicas.

Ageratina adenophora
Hojas – disposición – opuestas
Hojas – venación – palmado-nervadas,
Hojas – forma – lanceoladas,
Hojas – forma – ovadas,
Distribución – estado – Ciudad de México,
Distribución – estado – México,
Distribución – estado – Michoacán,
Distribución – estado – Morelos,
Distribución – estado – Puebla.

Quizá convenga adaptar el nombre del objeto "Distribución" por "Se distribuye en". Además, es necesario especificar el orden de los valores que se desea en la descripción. Por lo que la lista de OAV podría transformarse en:

10. Hojas – disposición – opuestas,
20. Hojas – venación – palmado-nervadas,
30. Hojas – forma – lanceoladas,
40. Hojas – forma – ovadas,
50. Se distribuye en – estado – Ciudad de México,
60. Se distribuye en – estado – México,
70. Se distribuye en – estado – Michoacán,
80. Se distribuye en – estado – Morelos,
90. Se distribuye en – estado – Puebla.

En esta lista se ha cambiado el nombre del objeto "Distribución" por "Se distribuye en" y el número que inicia el renglón es el orden deseado en la descripción. Así, la descripción generada por el sistema podría ser:

Ageratina adenophora.- Hojas opuestas; palmado-nervadas; lanceoladas u ovadas. Se distribuye en Ciudad de México, México, Michoacán, Morelos o Puebla.

La concatenación de los valores de un mismo atributo se considera una disyunción, por lo que se concatenan mediante "o" (o "u" cuando el siguiente valor comienza con "o"). Por ejemplo, "Hojas lanceoladas u ovadas".

La concatenación de los valores de diferentes atributos para el mismo objeto se considera una conjunción, por lo que se concatenan mediante "y" o ";". Por ejemplo, "Hojas opuestas; palmado-nervadas; lanceoladas u ovadas."

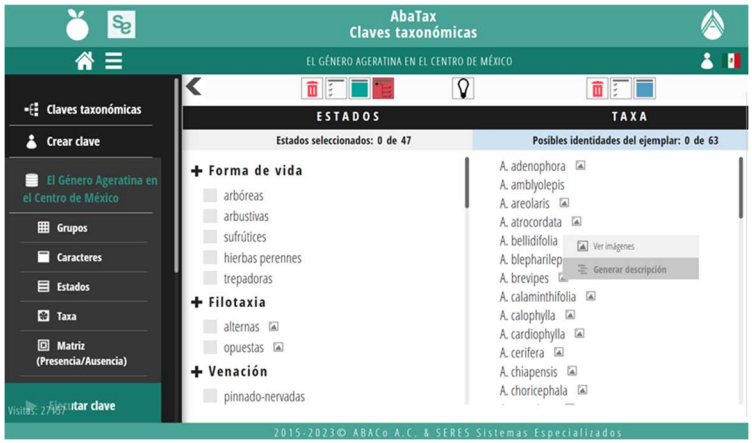


Fig. 3. Interfaz del usuario en AbaTax. En la ventana izquierda se muestran los atributos valores y en la derecha la lista de especies. Se observa el menú pop-up para la generación de descripciones que aparece dando clic sobre el nombre de una especie.

El separador de diferentes objetos es el "punto y seguido". Por ejemplo, en la descripción: "Hojas opuestas; palmado-nervadas; lanceoladas u ovadas. Se distribuye en Ciudad de México, México, Michoacán, Morelos o Puebla." se incluyen atributos y valores de dos objetos: "Hojas" y "Se distribuye en", por lo que antes del segundo se inserta un punto.

La red de transición aumentada (ATN) que especifica la gramática definida (Figura 2) incluye solo dos categorías de palabras (objeto y valor) y tres símbolos constantes (punto, punto y coma y la letra "o"). Los rasgos se han definido como valores booleanos resultado de la comparación entre los objetos y los atributos de pares de nodos conectados. El arco [2, start, "."] establece una concordancia objeto(OAV, previo) <> objeto(OAV, actual), mientras que para el arco [2, 1, "o"] la concordancia es objeto(OAV, previo) = objeto(OAV, actual) AND atributo(OAV, previo) = atributo(OAV, actual), y para el arco [2, 1, ";"] la concordancia es objeto(OAV, previo) = objeto(OAV, actual) AND atributo(OAV, previo) <> atributo(OAV, actual). Así, solo los componentes objeto y valor de la tripleta OAV son usados como categorías de palabras, mientras que el componente atributo es usado como rasgo.

4. Resultados

4.1. El sistema Web AbaTax

Con base en las consideraciones sobre la representación mediante OAV de la sección anterior y la gramática ATN definida, se desarrolló el sistema web AbaTax (www.abatax.abaco2.org; Figura 3). Este sistema permite la representación de las características de las especies de plantas mediante OAV y que genera las descripciones taxonómicas de cada especie en un lenguaje cercano al natural, como se requieren en las publicaciones científicas. La representación OAV se diseñó en dos niveles: en la base de datos relacional y en la interfaz del usuario.

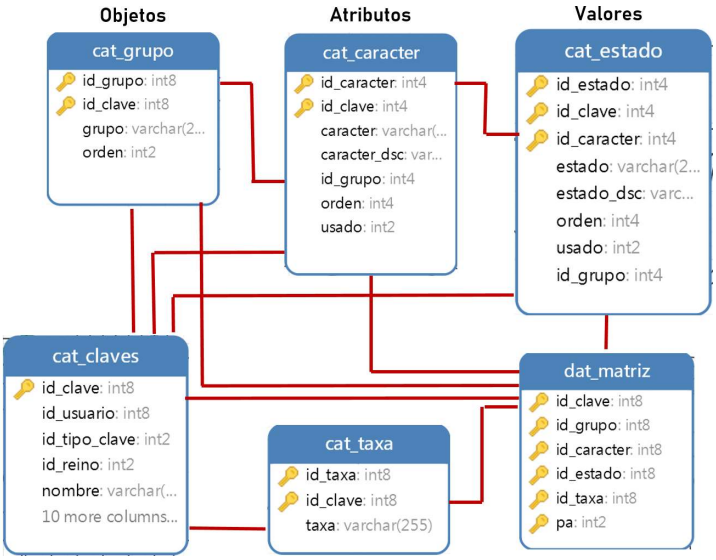


Fig. 4. Diagrama de relaciones de las tablas en las que se representan las tripletas OAV en AbaTax.

Las tripletas OAV también son usadas por el sistema para proveer una interfaz de identificación taxonómica (sistema experto) en la que el usuario indica las tripletas OAV presentes en el ejemplar bajo determinación. Una vez seleccionadas, el sistema filtra las especies congruentes con dichas selecciones [11].

4.2. Representación OAV en la base de datos

La base de datos del sistema AbaTax se implementó en PostgresSQL y está compuesta por más de 60 tablas, de las que seis están directamente involucradas en la representación OAV, y las restantes almacenan información administrativa de los usuarios y del control del despliegue de la interfaz del usuario.

El catálogo de OAV se almacena en tres diferentes tablas: cat_grupo, cat_caracter y cat_estado, que corresponden a los objetos, atributos y valores, respectivamente. En concordancia con lo expuesto en la sección anterior, los objetos corresponden a los grupos de caracteres y no a las especies. En la tabla cat_taxa se almacena la lista de especies, y en la tabla dat_matriz se especifica la relación de cuáles OAV se asocian a cada especie. En tabla cat_claves se almacena el catálogo de grupos de especies (Figura 4).

4.3. Representación OAV en la interfaz del usuario

A nivel de la interfaz del usuario, el OAV se implementó en tres secciones del sistema web: 1) en la interfaz de identificación taxonómica, en la que el usuario puede interactuar seleccionado las OAV presentes en el ejemplar bajo identificación (Figura 3); 2) en la matriz de datos OAV × especies, en la que el usuario puede indicar la

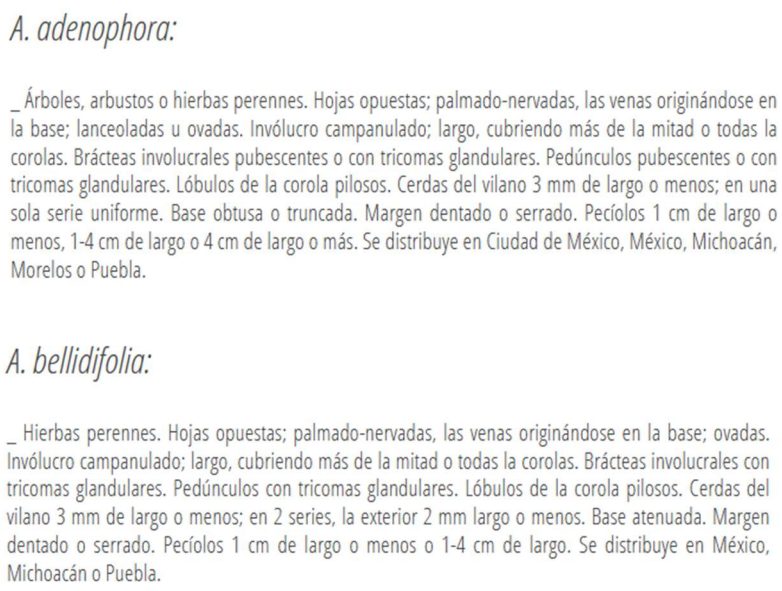


Fig. 6. Ejemplo de dos descripciones taxonómicas generadas en AbaTax de la policlave "El género *Ageratina* en el centro de México". Al hacer clic sobre el nombre del taxón en la interfaz se le da la instrucción a AbaTax para que genere su descripción (ventana derecha de la Figura 3).

presencia o ausencia de cada OAV en cada especie (Figura 5a); y 3) en el catálogo de OAV, en la que el usuario puede agregar, modificar o eliminar las tripletas OAV (Figura 5b).

4.4. Generación de descripciones taxonómicas en AbaTax

El sistema web AbaTax permite especificar los valores OAV de grupos de especies a cualquier usuario interesado en generar descripciones taxonómicas. Algunas matrices de datos son públicas estando disponibles a cualquier usuario de la web. Otras son privadas, disponibles solo para el usuario que las generó y a los usuarios para los que él designe permisos. El usuario debe elegir el grupo de especies del que desea generar descripciones taxonómicas, por ejemplo "El género *Ageratina* en el centro de México".

Después de leer la matriz de datos del grupo de especies seleccionado, el sistema mostrará la lista de las tripletas OAV en la ventana izquierda y la lista de especies en la ventana derecha (Figura 1). Al seleccionar alguna de las especies el sistema generará la descripción taxonómica correspondiente (Figura 6).

4.5. Fortalezas, debilidades y perspectivas

La herramienta construida, incluida el sistema web y la gramática definida, permiten al botánico especificar de una forma rápida las tuplas OAV necesarias para la generación automática de descripciones taxonómicas. Una de las virtudes es que las

a)

ESTADOS/TAXA	A. adenophora	A. amblyolepis	A. areolaris	A. atrocordata	A. bellidifolia
arbóreas	☑	☐	☑	☐	☐
arbustivas	☑	☑	☑	☑	☐
sufrútices	☐	☑	☐	☑	☐
hierbas perennes	☑	☑	☑	☑	☑
trepadoras	☐	☐	☐	☐	☐
alternas	☐	☐	☐	☐	☐
opuestas	☑	☑	☑	☑	☑

b)

Objetos	Atributos	Valores
Grupos	Caracteres	Estados
Plantas	Forma de vida	arbóreas
Plantas	Forma de vida	arbustivas
Plantas	Forma de vida	sufrútices
Plantas	Forma de vida	hierbas perennes
Plantas	Forma de vida	trepadoras
Hojas	Filotaxia	alternas
Hojas	Filotaxia	opuestas

Fig. 5. Representación de OAV en AbaTax. a) Matriz de datos OAV (renglones) × especie (columnas) editable por el usuario; b) Lista de catálogos de tripletas OAV como se muestran al usuario en el sistema.

descripciones generadas tienen un formato homogéneo. Sin embargo, se deja al usuario la responsabilidad de la redacción adecuada (concordancia) de cada uno de sus componentes, principalmente los objetos y los valores, pues los textos de los atributos no se incluyen en el texto de las descripciones.

Uno de los componentes importantes en las descripciones taxonómicas, que la herramienta no considera, son los adjetivos de frecuencia, por ejemplo, “siempre”, “a veces”, “casi nunca”. Este componente se podrá incluir en futuras versiones de la herramienta de forma sencilla asociándolo a cada tupla OAV, sin necesidad de modificar la red ATN.

El desarrollo de técnicas y herramientas para la generación automática de descripciones taxonómicas como los aquí presentados proveen el fundamento para la generación masiva de descripciones que son necesarias para documentar la flora de regiones con alta riqueza, como lo es la de México.

Las técnicas para el procesamiento del lenguaje natural podrán precisarse y especializarse para cubrir necesidades adicionales como lo es el análisis de textos botánicos para extraer información de forma automática y estructurarla en bases de datos.

5. Conclusiones

La generación de descripciones taxonómicas es una actividad que los botánicos sistemáticos realizan en su quehacer cotidiano como científicos. Mediante la representación del conocimiento objeto-atributo-valor (OAV), implementada en un modelo relacional de base de datos y realizando una correspondencia adecuada con los elementos taxonómicos, el OAV adquiere un alto poder de representación.

Todo esto facilita la generación automática de descripciones taxonómicas en lenguaje natural, cercanas a las requeridas en las publicaciones científicas. Además, la misma base de datos es útil en la generación de herramientas para la identificación taxonómica.

Con ello el sistema aporta dos importantes quehaceres rutinarios del botánico sistemático, la generación tanto de descripciones taxonómicas como la de claves para la identificación de los taxones bajo estudio.

Agradecimientos. Guadalupe Segura ha usado la herramienta AbaTax participando en la construcción de varias matrices de datos OAV, incluida la que se ejemplifica aquí, proponiendo mejoras y detectando errores en el sistema que lo han mejorado sustancialmente.

Referencias

1. Murguía-Romero, M., Ortiz, E., Serrano-Estrada, B., Villaseñor, J. L.: The Kew's "World checklist of vascular plants" and its relevance to the knowledge of the flora of Mexico. Botanical Sciences, vol. 101, No. 2, pp. 632–653 (2023) doi: 10.17129/botsci.3223
2. Villaseñor, J. L., Meave, J. A.: Floristics in Mexico today: Insights into a better understanding of biodiversity in a megadiverse country. Botanical Sciences, pp. 14–33 (2022) doi: 10.17129/botsci.3050
3. Sánchez-Sánchez, C. D., Velázquez-Ríos, P., Alvarado-Cárdenas, L. O.: 2022, un año récord en descubrimientos botánicos para México. Sociedad botánica de México, Macpalxóchitl, vol. 3, no. 2, pp. 34–42 (2023)
4. Nash, D. L., Nee, M.: Verbenaceae. Flora de Veracruz, fascículo 41, Instituto Nacional sobre Recursos Bióticos, Xalapa, Veracruz, México, pp. 154 (1984)
5. Sosa, V., Alvarado-Cárdenas, L. O., Duno de Stefano, R., González-Gallegos, J. G., Hernández-Sandoval, L., Jiménez-Rosenberg, R., Ochoterena, H., Rodríguez, A., Vibrans, H., Angulo, D. F.: The online flora of Mexico: eFloraMEX. Botanical Sciences, vol. 101, no. 2, pp. 324–340 (2023) doi: 10.17129/botsci.3123
6. Pankhurst, R. J.: The printing of taxonomic descriptions by computer. Taxon, vol. 27, no. 1, pp. 35–38 (1978) doi: 10.2307/1220476
7. Dallwitz, M. J.: A general system for coding taxonomic descriptions. Taxon, vol. 29, pp. 41–46 (1980) doi: 10.2307/1219595
8. Magalhaes, I. L.: Spreadsheets to expedite taxonomic publications by automatic generation of morphological descriptions and specimen lists. Zootaxa, vol. 4624, no. 1, pp. 147–150 (2019)
9. Orłowska, E., Pawlak, Z.: Expressive power of knowledge representation systems. International Journal of Man-Machine Studies, vol. 20, no. 5, pp. 485–500 (1984) doi: 10.1016/S0020-7373(84)80023-1

10. Van Melle, W.: MYCIN: A knowledge-based consultation program for infectious disease diagnosis. International journal of man-machine studies, vol. 10, no. 3, pp. 313–322 (1978) doi: 10.1016/S0020-7373(78)80049-2
11. Murguía-Romero, M., Serrano-Estrada, B., Ortiz, E., Villaseñor, J. L.: Taxonomic identification keys on the web: tools for better knowledge of biodiversity. Revista Mexicana de Biodiversidad, vol. 92, no. e923592 (2021) doi: 10.22201/ib.20078706e.2021.92.3592
12. Perez-Perez, M. A., Ayers, T. J., Amith, J. D.: A new species of Lobelia (Campanulaceae: Lobelioideae) from the Sierra Madre Oriental, Mexico. Phytotaxa, vol. 568, np. 1, pp. 1–7 (2022)
13. Waterman, D. A.: A rule-based approach to knowledge acquisition for man-machine interface programs. International Journal of Man-Machine Studies, vol. 10, no. 6, pp. 693–711 (1978) doi: 10.1016/S0020-7373(78)80028-5

ISSN 1870-4069

Reconocimiento y detección de señales de tráfico mexicanas de tipo preventivas, restrictivas e informativas aplicando YOLOv4

Daniela Bolaños-Flores[1], Hamurabi Gamboa-Rosales[1], José M. Celaya-Padilla[1], Tania A. Ramirez-del Real[2,3]

[1] Universidad Autónoma de Zacatecas, México

[2] Consejo Nacional de Ciencia y Tecnología, México

[3] Centro de Investigación en Ciencias de Información Geoespacial, México

tramirez@centrogeo.edu.mx, bolanosfloresdaniela@gmail.com

Resumen. Una variedad de factores a lo largo del camino pueden poner en peligro la seguridad de los conductores o peatones y dar lugar a accidentes de alto impacto durante la conducción, por lo que las señales de tránsito son elementos esenciales que brindan información sobre el estado del camino durante el viaje. Técnicas novedosas permiten el desarrollo de herramientas que ayudan a reconocer y clasificar objetos de interés. En este trabajo en particular, se emplean para diseñar un modelo de reconocimiento y detección de señales de tránsito mexicanas, utilizando técnicas de aprendizaje computacional. Se recolectó un conjunto de datos enfocado en señales de tránsito en 5 ciudades diferentes de la república mexicana dentro de las principales vías urbanas. La base de datos contiene un total de 2,160 elementos de carretera divididos en 14 clases diferentes para el entrenamiento y validación de algoritmos. Además, la recopilación de datos tuvo lugar en diversas condiciones climáticas.La detección automática se realiza mediante una red neuronal convolucional denominada You Only Look Once (YOLOv4), que presenta un porcentaje mean Average Precision (mAP) mayor al 95 %.

Palabras clave: Reconocimiento y detección, señales de tráfico, aprendizaje computacional.

Recognition and Detection of Preventive, Restrictive and Informative Mexican Traffic Signs Applying YOLOv4 Architecture

Abstract. A variety of factors along the road can jeopardize the safety of drivers or pedestrians and lead to high-impact accidents while driving. Therefore, traffic signs are essential elements, providing information about the state of road

during travel. Novel techniques allow for the development of tools that help to recognize and classify objects of interest. This work uses computational learning techniques to design a recognition and detection model for Mexican traffic signs. A dataset focused on traffic signs was collected in 5 different cities of the Mexican Republic within the main urban roads. The database contains 2,160 road elements divided into 14 classes for algorithm training and validation—furthermore, data collection takes place in various weather conditions. The automatic detection uses a convolutional neural network called You Only Look Once (YOLOv4), yielding a mean Average Precision (mAP) percentage more significant than 95%.

Keywords: Recognition and detection, traffic signs, machine learning.

1. Introducción

En la actualidad, el desarrollo de la tecnología ha aumentado de manera considerable, por lo que para la sociedad y para diversas áreas de la ingeniería ha sido de gran aportación, como la interpretación de información visual en máquinas, haciendo uso de técnicas computacionales para el reconocimiento en imágenes, obteniendo resultados prometedores en diferentes campos de aplicación.

En el área automotriz se han implementado arquitecturas basadas en inteligencia artificial, donde se proponen y desarrollan modelos inteligentes de reconocimiento y detección de elementos viales, posteriormente se pueden utilizar en autos de conducción inteligente, permitiendo al piloto considerar señales o elementos que se encuentran en el transcurso de carreteras o zonas urbanas con la finalidad de advertir señales que no haya atendido. Estos vehículos incluyen funciones de piloto automático que permiten al usuario identificar señales de alto y semáforos a cierta distancia, siempre y cuando esté bajo una supervisión de manera activa [24].

Por otra parte, adquirir una de estas unidades suele ser costoso, además, las señales que identifica y reconoce son enfocadas al estándar estadounidense, asiático y europeo, generando una limitación en el área donde se implementa. Con frecuencia la falta de seguridad vial dentro del territorio mexicano se debe a ciertos factores, como la ausencia de elementos de señalización en zonas urbanas o carreteras trayendo consigo consecuencias como accidentes catastróficos [23].

También, en algunas zonas, las señales sufren modificaciones por diversas causas climáticas o vandalizaciones, por lo que la conducción es más complicada para el individuo. La literatura técnico ciéntifica reporta modelos inteligentes asociados al análisis de señales viales o de tráfico dentro de algún país en particular, lo cual presenta una brecha, ya que los modelos propuestos no alcanzan la generalización, debido a la implementación con conjuntos de datos específicos a la localidad donde se formularon [17].

Lo anterior, ha permitido la propuesta de diversos trabajos de investigación enfocados al reconocimiento y detección de señales de tráfico basados en técnicas de aprendizaje automático [1, 8, 9, 12, 27], obteniendo resultados con buen desempeño al evaluarlos, el área de oportunidad puede establecerse por la existencia del sesgo de la información, debido a la región de adquisición de datos, que generalmente pertenecen a

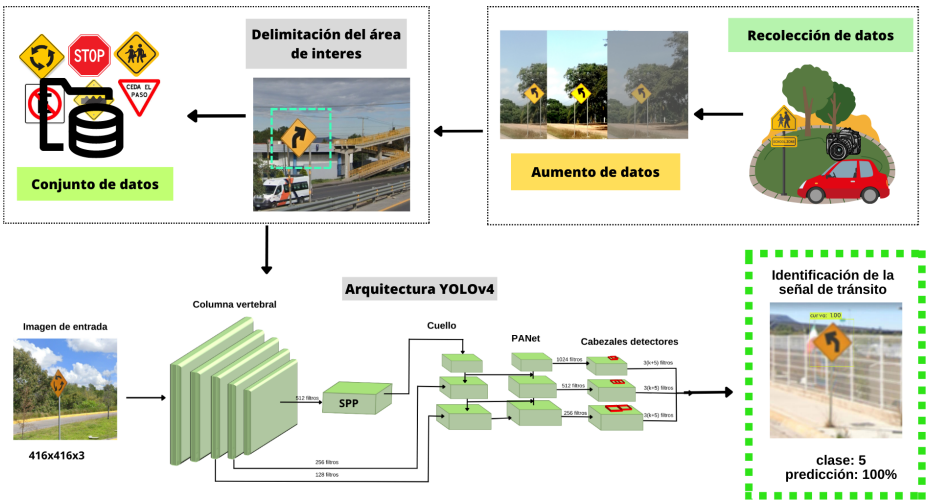


Fig. 1. Diagrama general para el desarrollo del modelo.

señales extranjeras, por lo que complica la implementación para el territorio mexicano, ya que existen diferencias entre las señales mexicanas y las extranjeras, como lo son el diseño, el color, la simbología, entre otras.

Considerando las problemáticas previamente expuestas, surge la necesidad de proponer una base de datos que contenga señales de tránsito en las principales zonas urbanas de diversas ciudades de la región. Este conjunto de datos se utilizará posteriormente para implementar un modelo capaz de identificar y reconocer las señales preventivas y restrictivas en el territorio mexicano.

El presente trabajo está organizado de la siguiente forma, en la Sección 2 se encuentran los principales trabajos relacionados, en la Sección 3 se explican los materiales y métodos enfocados al conjunto y procesamiento de datos, además la arquitectura de YOLOv4 y el proceso para el desarrollo del modelo. A continuación, en la Sección 4 se localiza la experimentación y resultados presentados, además de la evaluación y desempeño general del algoritmo. Finalmente, en la Sección 5, se concluye y proponen los trabajos futuros.

2. Trabajos relacionados

Dentro de la literatura existe una extensa cantidad de investigaciones acerca de Reconocimiento de Señales de Tráfico (TSR, por sus siglas en inglés) y Detección de Señales de Tráfico (TSD, por sus siglas en inglés); donde la mayoría de los autores utilizan un conjunto de datos públicos con una variedad de clases y un extenso número de imágenes, principalmente se pueden mencionar los siguientes:

– High Resolution Remote Sensing Detection(HRRSD) [28]: Este grupo contiene 13 categorías donde solo el 11 % pertenece a señales de tráfico asiáticas.

Tabla 1. Tipos de señales de tránsito.

Número de secuencia	Tipo	Total
	Señales Restrictivas	
1	Ceda el paso	69
2	Límite de velocidad	422
3	Prohibido estacionarse	408
4	Alto	173
	Señales Preventivas	
5	Cruce peatonal	355
6	Curva	135
7	Glorieta	68
8	Incorporación al tránsito	156
9	Salida	140
10	Reductor de velocidad	144
	Señales Informativas	
11	Parada de autobus	20
12	Zona de discapacitados	35
13	Zona de taxis	15
14	Gasolinera	20
Total		**2160**

– Tsinghua-Tencent 100K (TT100K) [29]:Este conjunto contiene 128 clases donde solo 30,000 están enfocados a señales de tránsito asiáticas.

– German Traffic Sign Recognition Benchmark (GTSRB) [10] : Este conjunto cuenta con 43 categorías y una cantidad aproximada de 51,839 imágenes con elementos viales alemanes.

Derivado de los conjuntos de datos anteriores, surgen diversos trabajos con propuestas para mejorar el desempeño en la detección de señales viales, a continuación se describen algunos de los planteamientos y resultados sobresalientes más actuales.

Liu et al. [16] desarrollaron un modelo inteligente de TSR y TSD. Dentro de la fase de prueba del sistema, se utilizó un conjunto de datos denominado HRRSD donde contiene aproximadamente 21,761 imágenes con categorías como vehículos, avión, puentes, entre otras, es importante resaltar que los datos no están enfocados a señales de tránsito, sino, es utilizada para evaluar la capacidad del algoritmo propuesto, obteniendo un porcentaje de 85.5 % de mean Average Precision (mAP).

En los trabajos de Liu et al. y Li et al. [14, 13] utilizaron la base de datos asiática denominada TT100K para la creación del modelo de reconocimiento o detección. El principal enfoque en el trabajo de Liu et al. [14] es el reconocimiento de señales de tráfico TSR, de la misma forma, los autores utilizaron 10,267 imágenes del total del conjunto de datos TT100K, para desarrollar un modelo que permita reconocer señales de proporción limitada, borrosas y complejas dentro de un ámbito natural, por medio de una arquitectura modificada obteniendo resultados por arriba del 86 % en mAP, una precisión de 87.45 % y una sensibilidad de 79.65 %.

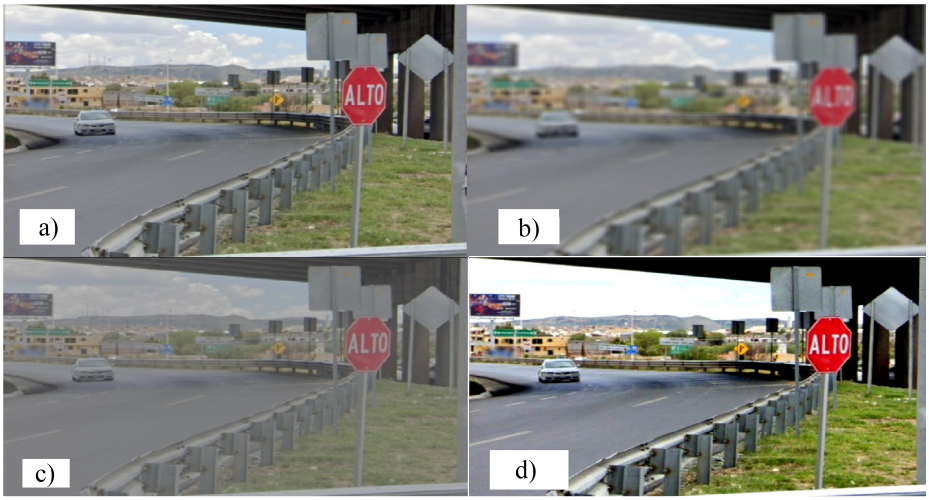


Fig. 2. Demostración de aumento de datos para la clase alto. a) imagen original, b) imagen en desenfoque, c) imagen con baja saturación y d) imagen con alta saturación.

Li et al. [13] realizaron reconocimiento y detección de señales de tráfico (TSR y TSD), utilizando 5,857 imágenes del total del conjunto de TT100K para la fase de entrenamiento y prueba, desarrollando un modelo que reconoce y clasifica señales con dígitos numéricos, dentro de un recorrido en automóvil, utilizando una red neuronal basada en aprendizaje profundo, modificada y propuesta por los creadores, presentando en un desempeño del 72 % medido en mAP.

Posteriormente, los trabajos de Cao et al., Vennelakanti et al., Barodi et al. y Yao et al. [6, 25, 2, 26], hacen uso del conjunto de datos alemanes denominados GTSRB para evaluar el desempeño del modelo propuesto. Los estudios mencionados también utilizan imágenes de diversos países como lo son de China, Nueva Zelanda, India, México y Marruecos.

Particularmente, Cao et al. [6] realizaron reconocimiento y detección de señales de tráfico, por medio de la arquitectura LeNet-5 CNN y otra herramienta modificada para mejorar la seguridad del individuo al momento de conducir, permitiendo tener un asistente en la conducción de vehículos, su desempeño logró un mAP de 99.75 %.

Vennelakanti et al. [25] utilizaron 21,383 imágenes del total del conjunto de datos German Traffic Sign Recognition Benchmark (GTSRB), donde propusieron un modelo con una arquitectura denominada CNN-Ensemble, teniendo la finalidad de reconocer y detectar señales circulares y triangulares pertenecientes a Bélgica y Alemania, obteniendo una tasa de reconocimiento del 99.75 %.

Posteriormente, Ramkumar et al. [20] proponen una arquitectura de elaboración propia contando con 500 capas ocultas, la cual permite realizar reconocimiento y clasificación, utilizando el total de datos del conjunto de datos alemán para el entrenamiento del modelo, presentando resultados con una exactitud de 98.64 %. De igual manera, para la investigación de Barodi et al. [2] proponen una de arquitectura de redes neuronales convolucionales, donde utilizan distintas técnicas de visión e

Fig. 3. Anotación de imágenes por medio de LabelImg.

implementación de inteligencia artificial aplicado a tráfico en tiempo real, presentando resultados del 98 % de precisión, una sensibilidad del 98 % y un valor F1 del 98 %. El estudio de Yao et al. [26] propone un modelo inteligente para reconocer señales asiáticas, su evaluación se realiza en diversos conjuntos de datos.

Para el primer modelo se utiliza la base de datos de GTSDB, donde se utiliza la cantidad de 15,734 imágenes, la arquitectura empleada es una combinación de YOLOv4 Tiny [11], de igual manera un método de fusión de características basado en una Red de Pirámide de Características Adaptativas (AFPN, por sus siglas en inglés) [11] y un Bloque de Campo Receptivo (RFB, por sus siglas en inglés), el cual consiste en poner escalas de campos receptivos mejorando la capacidad de extracción de características para la red neuronal convolucional [15], logrando el 93.5 % y 82.4 % de precisión y sensibilidad, asimismo, presentan porcentajes de valor F1 de 87.6 % y mAP de 86.8 %. De lo anterior es importante destacar la falta de datos con señales en el territorio Mexicano.

3. Materiales y métodos

En la Figura 1 se presenta la metodología usada para el desarrollo del modelo de reconocimiento y detección de señales de tráfico utilizando la arquitectura de una red neuronal convolucional designada como YOLOv4.

De entrada, se realizó una recolección de datos de dos maneras, en la primera se utilizó una cámara de video dentro de un vehículo haciendo una recorrido por la ciudad, para la segunda forma se utilizaron herramientas de Google que permiten la visualización de rutas en calles de zonas urbanas y carreteras.

Una vez terminada la recopilación, se procedió a aplicar la técnica de aumento de datos para transformar las imágenes con vista a diferentes cambios climáticos, tales como días muy soleados simulando ciudades que se encuentran cerca del mar, neblinosos suponiendo a los lugares donde existe más humedad en el ambiente y por último, días donde la temperatura es constante.

Tabla 2. Especificación del etiquetado de imagen.

Clase	Coordenada en x	Coordenada en y	Ancho	Alto
Ceda el paso	0.0.153802	0.282593	0.113688	0.192693
Prohibido estacionarse	0.545817	0.315544	0.122814	0.267192

Una vez completa la tarea anterior, se procede a delimitar la zona de interés, es decir, por medio de herramientas de etiquetado se procede a delimitar donde se encuentran las señales de tránsito, con el fin de proponer el conjunto de datos para entrenamiento, validación y prueba.

El siguiente paso es el entrenamiento del modelo computacional, donde se utiliza una red pre-entrenada con una arquitectura utilizada para predicciones, por medio de supresión máxima, asimismo, la primera fase de la red neuronal convolucional es conocida como CSPDarknet53 y el resto de la estructura contiene diversas capas de extracción de características [22].

El bloque de Agrupación de Pirámide Espacial (SPP, por sus siglas en inglés), está conectado desde las capas principales hasta las capas de extracción de características de igual manera que la Red de agregación de rutas (PANet, por sus siglas en inglés), la cual extrae características en diferentes etapas del procesamiento, por último, se encuentra la capa de detección donde realiza una combinación de capas en varios niveles ayudando a detectar objetos de diferentes tamaños. Cabe mencionar que la función de activación utilizada es la Mish [22].

3.1. Recolección de datos

En la primera etapa se hizo la recolección de señales de tráfico verticales, las cuales se dividen en preventivas, restrictivas e informativas. Este trabajo se enfoca en señales restrictivas y preventivas, debido a que las informativas varían dependiendo el área de la región. Las imágenes recolectadas fueron adquiridas de las ciudades de Puebla de Zaragoza, Veracruz, Zacatecas, Monterrey y Xalapa, dentro de las avenidas principales, parques, áreas verdes, mercados y áreas urbanas.

Por otra parte, para la captura de las imágenes utilizadas en el entrenamiento del modelo computacional de reconocimiento y detección fue de dos maneras diferentes, la primera opción fue por medio de una herramienta de Google Maps denominada Google Street View, la cual permite la visualización de calles y avenidas de manera virtual.

Mientras, que la segunda opción fue dentro las calles principales de la capital de Zacatecas, con la ayuda de una cámara de video (GoPro Session 5). Obteniendo así las siguientes categorías representadas en la Tabla 1.

El total de las imágenes recolectadas fueron 1,966 y el 28.5 % de estas representan señales restrictivas, 43 % pertenecen a señales preventivas y el 28.5 % son señales informativas. Además, como las señales de tránsito recolectadas se encuentran en un entorno no controlado se presentan aspectos fotográficos como cambio de iluminación, diferentes ángulos y enfoque variable.

Tabla 3. Conjunto de datos de entrenamiento, validación y prueba.

	Clase	Entrenamiento	Validación	Prueba	Total
1	Alto	145	20	8	173
2	Prohibido estacionarse	350	41	17	408
3	Ceda el paso	57	9	3	69
4	Limite de velocidad	351	44	27	422
5	Cruce peatonal	281	57	17	355
6	Curva	112	16	7	135
7	Glorieta	54	12	2	68
8	Incorporación al transito	87	59	10	156
9	Salida	109	26	5	140
10	Reductor de velocidad	124	16	4	144
11	Parada de autobus	12	6	2	20
12	Zona de discapacitados	23	10	2	35
13	Gasolinera	12	6	2	20
14	Zona de taxis	11	2	2	15
Total		**1728**	**324**	**108**	**2160**

3.2. Aumento de datos

El aumento de datos consiste en un conjunto de técnicas que permite ampliar artificialmente la cantidad de información a partir de datos [7], realizando modificaciones a la imagen. Por lo que, para que exista variedad dentro del conjunto de datos recolectado se procederá a modificar la saturación en un rango de menor a mayor y la visibilidad realizando difuminado, con el fin de simular diferentes condiciones climáticas como días con lluvia o neblina, y condiciones de iluminación como nublado y soleado. Teniendo como ventaja de que el conjunto se adapte más en línea con escenas reales dentro de la vialidad.

3.3. Cuadro delimitador en la región de interés

Para realizar la detección del objeto de interés en el interior de la imagen se procede a realizar el etiquetado por medio de un cuadro delimitador conocido como BoundingBox, la delimitación se realiza mediante la herramienta, indicando donde se encuentra la señal de tránsito dentro de la ilustración, con el fin de comparar la etiqueta a predecir contra la del modelo.

La Figura 3 representa la anotación que se realizó por medio de la interfaz de etiquetado LabelImg con una imagen del conjunto de datos propuesto, donde se puede observar dos cuadros delimitadores, uno perteneciente a la clase de ceda el paso y otro a la señal de prohibido estacionarse. Una vez concluido, se procede a guardar en formato tipo YOLO, este formato guarda las coordenadas de la región etiquetada, es decir, las coordenadas x y y demuestran el centro del cuadro delimitador en el que se encuentra el objeto, ademas del "ancho" y "alto" del cuadro que se debe detectar.

De igual modo, en la Tabla 2 se observa un ejemplo de las coordenadas almacenadas mencionadas anteriormente y correspondientes a las clases de ceda el paso y prohibido estacionarse.

Tabla 4. Parámetros del entorno de experimentación.

Sistema operativo	Ubuntu 9.4.0
Tipo de GPU	NVIDIA Tesla T4
RAM	25.5
Lenguaje de programación	Python 3.8.10
Versión de CUDA	11.6

3.4. Preparación de imágenes de entrenamiento y prueba

Para la preparación de imágenes de entrenamiento y prueba se realizó una división del grupo total de imágenes en una proporción de 80:15:5, correspondiente a un 80 % para entrenamiento, 15 % para validación y un 5 % de prueba, como lo menciona [21] el set de prueba nos permite visualizar si el modelo generado realiza reconocimiento y detección con un desempeño aceptable.

En este trabajo la métrica para medir el rendimiento del modelo es mean Average Precision (mAP). Esta métrica permite evaluar la precisión con la cual se detecta el área que ocupa el objeto, para el cálculo de esta primero se mide la precisión promedio para cada clase, posteriormente se calcula la media de las precisiones obtenidas [19].

De igual modo, la distribución en la que se encuentran las imágenes de entrenamiento validación y prueba se pueden observar en la Tabla 3 el conjunto de datos está conformado por 14 clases, donde están incluidas los tres tipos de señalamientos verticales, de igual manera, se muestra la cantidad equivalente de imágenes para el proceso de entrenamiento, validación y prueba.

4. Experimentación y resultados

4.1. Ambiente de experimentación

Para llevar a cabo el experimento de reconocimiento y detección de señales de tránsito implementando la arquitectura de YOLOv4 y basándose en el conjunto de datos propuesto, se utilizó el entorno de codificación de Google Colaboratory, debido a que contiene un soporte de GPU y dependencias necesarias [3]. En la Tabla 4 se muestra los parámetros que se utilizaron para el entrenamiento del modelo de reconocimiento y detección.

4.2. Configuración de la red y generación del modelo

Una vez separado y preparado el conjunto de datos, se deberán establecer dentro de Google Drive, debido a que el entorno experimental de Google Colab requiere acceso a los datos de esa plataforma. Posterior a esto, se procede a adaptar los parámetros establecidos mostrados en la Tabla 4, debido a que se requiere la GPU ya que se necesitan dependencias que posibiliten utilizar el procesamiento gráfico como el instrumento de CUDA [18], teniendo una mejor optimización de costo computacional para el desarrollo del modelo.

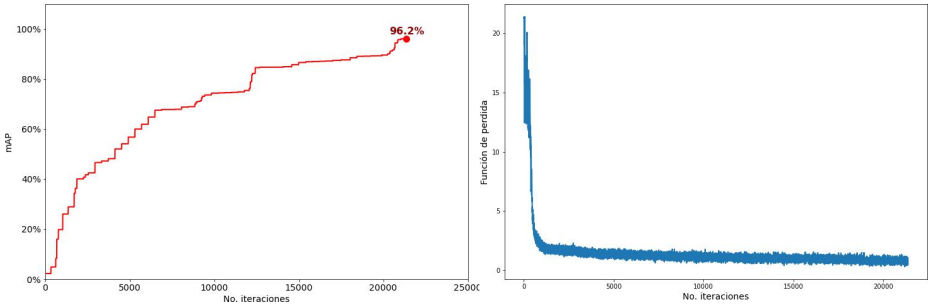


Fig. 4. Rendimiento promedio de la red YOLOv4 en el reconocimiento de señales de tránsito.

Además, del cambio de entorno de ejecución se requieren herramientas que contribuyan a la visión artificial, como OpenCV[5]. Posteriormente, se realiza la implementación de la arquitectura a utilizar desde el repositorio de los autores [4] y se procede a descargar los pesos pre-entrenados para adaptarlos dentro del entrenamiento.

4.3. Resultados del entrenamiento

Para llevar a cabo el entrenamiento del modelo, se concretó por medio de 1,728 imágenes donde están incluidas las 14 clases existentes de señales de tránsito de tipo restrictivas, preventivas e informativas y sus variantes. En la Figura 4, se observa el rendimiento promedio del modelo, donde alcanzó un porcentaje mayor del 95 % de mAP en la detección del objeto de estudio durante 21,371 iteraciones en un aproximado de 10 horas.

Una vez terminada la ejecución del entrenamiento, se procede a evaluar la predicción del modelo desarrollado por medio del algoritmo YOLOv4, para este proceso se utilizó la partición de 108 imágenes pertenecientes al conjunto de prueba.

Asimismo, como demostración de este procedimiento algunas de las pruebas de experimentación se muestran en la Figura 5, donde se puede observar que las 7 imágenes presentadas contienen una o dos señales de tránsito diferentes tales como ceda el paso, prohibido estacionarse, curva, glorieta, cruce de peatones, alto, incorporación y límite de velocidad.

Además de mostrar la delimitación de la detección realizada por el modelo, representada en color morado y por otra parte, en color amarrilo, el resultado esperado ideal o mejor conocido como ground truth, el cual se generó mediante la interfaz de etiquetado. Por último, se observa la diferencia que existe entre ambos cuadros delimitadores.

Es relevante destacar que el algoritmo implementado en este estudio recorre la imagen de entrada hasta el punto de identificar el objeto de interés y, posteriormente delimitarlo de acuerdo a su respectiva clase. Los resultados presentados demuestran una alta eficiencia en el reconocimiento de diferentes tipos de señales de tránsito, lo que se traduce a un excelente desempeño del modelo.

Fig. 5. Demostración de la posición real del objeto y la detección del algoritmo propuesto.

5. Conclusiones y trabajos futuros

En el presente trabajo, se hizo uso de una red neuronal convolucional denominada You Only Look Once (YOLOv4), la cual permite realizar un modelo para reconocer y detectar elementos viales dentro de las zonas urbanas y carreteras de tipo restrictivas, preventivas e informativas.

Asimismo, se han aplicado diferentes técnicas de aumento de datos para alimentar el conjunto de datos propuesto para este estudio y tener variabilidad y robustez al momento de desarrollar el modelo. Este enfoque se utiliza comúnmente dentro del aprendizaje computacional mejorando significativamente la capacidad del modelo al tener nuevos escenarios.

La Figura 4 muestra el comportamiento del modelo en términos de la métrica establecida (mAP), las cuales presentan un excelente desempeño con un porcentaje mayor al 95 %. Además, el ejemplo de la Figura 5 ilustra el reconocimiento exitoso y preciso en imágenes con distintas señales de tránsito, donde se puede observar al delimitar la posición exacta dentro de la imagen. Estos resultados resaltan la capacidad del modelo para abordar eficientemente la tarea de reconocimiento y detección de señales de tráfico en diferentes entornos complejos.

Finalmente, se propone como trabajo futuro el incremento del conjunto de datos utilizado en este estudio, con el propósito de abarcar una mayor diversidad de señalética urbana presente en el territorio mexicano. Además, se plantea la implementación de nuevas arquitecturas para poder comparar y evaluar el desempeño en relación al modelo propuesto.

Agradecimientos. Los autores agradecen al Programa Nacional de Posgrados de Calidad (PNPC) del Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico a través de la convocatoria Becas Nacional (Tradicional) 2021-2.

Referencias

1. Almutairy, F., Alshaabi, T., Nelson, J., Wshah, S.: ARTS: Automotive repository of traffic signs for the United States. In: IEEE Transactions on Intelligent Transportation Systems, vol. 22, pp. 457–465 (2021) doi: 10.1109/tits.2019.2958486
2. Barodi, A., Bajit, A., Zemmouri, A., Benbrahim, M., Tamtaoui, A.: Improved deep learning performance for real-time traffic sign detection and recognition applicable to intelligent transportation systems. International Journal of Advanced Computer Science and Applications, vol. 13, no. 5 (2022)
3. Bisong, E.: Building machine learning and deep learning models on google cloud platform, Apress (2019) doi: 10.1007/978-1-4842-4470-8
4. Bochkovskiy, A., Wang, C. Y., Mark-Liao, H. Y.: Yolov4: Optimal speed and accuracy of object detection (2020) doi: 10.48550/arXiv.2004.10934
5. Bradski, G.: The OpenCV library. Dr Dobb's Journal of Software Tools, vol. 25, no. 11, pp. 120–125 (2000)
6. Cao, J., Song, C., Peng, S., Xiao, F., Song, S.: Improved traffic sign detection and recognition algorithm for intelligent vehicles. Sensors, vol. 19, no. 18, pp. 4021 (2019) doi: 10.3390/s19184021
7. Dilmegani, C.: What is data augmentation? techniques y examples in 2023 (2022) research.aimultiple.com/data-augmentation/
8. Dubey, U., Chaurasiya, R. K.: Efficient traffic sign recognition using CLAHE-based image enhancement and ResNet CNN architectures. International Journal of Cognitive Informatics and Natural Intelligence, vol. 15, no. 4, pp. 1–19 (2021) doi: 10.4018/IJCINI.295811
9. Gao, X., Chen, L., Wang, K., Xiong, X., Wang, H., Li, Y.: Improved traffic sign detection algorithm based on Faster R-CNN. Applied Sciences, vol. 12, no. 18, pp. 8948 (2022) doi: 10.3390/app12188948
10. Houben, S., Stallkamp, J., Salmen, J., Schlipsing, M., Igel, C.: Detection of traffic signs in real-world images: The german traffic sign detection benchmark. In: International Joint Conference on Neural Networks, pp. 1–8 (2013) doi: 10.1109/ijcnn.2013.6706807
11. Khokhlov, I., Davydenko, E., Osokin, I., Ryakin, I., Babaev, A., Litvinenko, V., Gorbachev, R.: Tiny-YOLO object detection supplemented with geometrical data. In: IEEE 91st Vehicular Technology Conference, pp. 1–5 (2020) doi: 10.1109/vtc2020-spring48590.2020.9128749
12. Lahmyed, R., Ansari, M. E., Kerkaou, Z.: Automatic road sign detection and recognition based on neural network. Soft Computing, vol. 26, no. 4, pp. 1743–1764 (2022) doi: 10.1007/s00500-021-06726-w
13. Li, Z., Chen, M., He, Y., Xie, L., Su, H.: An efficient framework for detection and recognition of numerical traffic signs. In: IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 2235–2239 (2022) doi: 10.1109/icassp43922.2022.9747406
14. Liu, S., Cai, T., Tang, X., Zhang, Y., Wang, C.: Visual recognition of traffic signs in natural scenes based on improved RetinaNet. Entropy, vol. 24, no. 1, pp. 112 (2022) doi: 10.3390/e24010112
15. Liu, S., Huang, D., Wang, Y.: Receptive field block net for accurate and fast object detection (2017) doi: 10.48550/ARXIV.1711.07767
16. Liu, Y., Shi, G., Li, Y., Zhao, Z.: M-YOLO: Traffic sign detection algorithm applicable to complex scenarios. Symmetry, vol. 14, no. 5, pp. 952 (2022) doi: 10.3390/sym14050952
17. Narejo, S., Talpur, S., Memon, M., Rahoo, A.: An automated system for traffic sign recognition using convolutional neural network. 3c Tecnología: Glosas de Innovación Aplicadas a la PYME, vol. 9, no. 1, pp. 119–135 (2020)

18. NVIDIA, Vingelmann, P., Fitzek, F. H.: Cuda, release: 10.2.89 (2020) developer.nvidia.com/cuda-toolkit
19. Padilla, R., Passos, W. L., Dias, T., Netto, S. L., da Silva, E. A. B.: A comparative analysis of object detection metrics with a companion open-source toolkit. Electronics, vol. 10, no. 3, pp. 279 (2021) doi: 10.3390/electronics10030279
20. Ramkumar, S., Ganapathy, R., Sridhar, K.: Traffic sign detection and recognition using CNN. ECS Transactions, vol. 107, no. 1, pp. 17447–17455 (2022) doi: 10.1149/10701.17447ecst
21. Recuero-de-los Santos, P.: Datos de entrenamiento vs datos de test (2022) empresas. blogthinkbig.com/datos-entrenamiento-vs-datos-de-test/
22. Roszyk, K., Nowicki, M. R., Skrzypczyński, P.: Adopting the YOLOv4 architecture for low-latency multispectral pedestrian detection in autonomous driving. Sensors, vol. 22, no. 3, pp. 1082 (2022) doi: 10.3390/s22031082
23. Ruiz-Rivero, M. S., Astorga-Bustillos, F. R., Villa-Herrera, J. E.: La importancia de las señales de tránsito en las vías terrestres. FINGUACH Revista de Investigación Científica de la Facultad de Ingeniería de la Universidad Autónoma de Chihuahua, vol. 5, no. 17, pp. 10–11 (2018)
24. Support, T.: Piloto automático y capacidad de conducción autónoma total (2020) www.tesla.com/es_ES/support/autopilot-and-full-self-driving-capability
25. Vennelakanti, A., Shreya, S., Rajendran, R., Sarkar, D., Muddegowda, D., Hanagal, P.: Traffic sign detection and recognition using a CNN ensemble. In: IEEE International Conference on Consumer Electronics (ICCE) (2019) doi: 10.1109/icce.2019.8662019
26. Yao, Y., Han, L., Du, C., Xu, X., Jiang, X.: Traffic sign detection algorithm based on improved YOLOv4-tiny. Signal Processing: Image Communication, vol. 107, pp. 116783 (2022) doi: 10.1016/j.image.2022.116783
27. Zhang, H., Zhao, J.: Traffic sign detection and recognition based ondeep learning. Engineering Letters, vol. 30, no. 2 (2022)
28. Zhang, Y., Yuan, Y., Feng, Y., Lu, X.: Hierarchical and robust convolutional neural network for very high-resolution remote sensing object detection. In: IEEE Transactions on Geoscience and Remote Sensing, vol. 57, pp. 5535–5548 (2019) doi: 10.1109/TGRS.2019.2900302
29. Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., Hu, S.: Traffic-sign detection and classification in the wild. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2110–2118 (2016) doi: 10.1109/CVPR.2016.232

ISSN 1870-4069

Análisis de sentimientos en críticas de cine utilizando SVM y combinación de n-gramas

Cesar Alexis Estrada Palacios, José Luis Tapia Fabela

Universidad Autónoma del Estado de México,
Estado de México,
México

{kirdrazler, joseluis.fabela}@gmail.com

Resumen. El análisis de sentimientos en críticas de cine es una tarea importante en la industria cinematográfica y en la investigación académica. En este trabajo se propone un nuevo método para el análisis de sentimientos en críticas de cine utilizando *SVM* y la combinación de n-gramas. El método fue evaluado en el corpus Muchocine y se comparó con los métodos del estado del arte. Los resultados muestran que el método propuesto logra un mejor rendimiento en términos de *precision*, *recall* y *f-measure* en comparación con los métodos del estado del arte, demostrando que la combinación de n-gramas como modelo de representación mejora el rendimiento del clasificador. Las aportaciones de este trabajo incluyen el nuevo método propuesto, la demostración de la eficacia de la combinación de n-gramas y la comparación con los métodos del estado del arte.

Palabras clave: Análisis de sentimientos, críticas de cine, SVM, n-gramas.

Sentiment Analysis in Movie Reviews Using SVM and Combination of N-grams

Abstract. Sentiment analysis in film reviews is an important task in the film industry and in academic research. In this paper, a new method for the analysis of sentiments in film reviews is proposed using SVM and the combination of n-grams. The method was evaluated in the Muchocine corpus and compared with state-of-the-art methods. The results show that the proposed method achieves a better performance in terms of precision, recall and f-measure compared to the state of the art methods, demonstrating that the combination of n-grams as a representation model improves the performance of the classifier. The contributions of this work include the new proposed method, the demonstration of the efficiency of the combination of n-grams and the comparison with the methods of the state of the art.

Keywords: Sentiment analysis, movie reviews, SVM, n-grams.

1. Introducción

La creciente cantidad de información disponible en línea ha llevado a una mayor necesidad de herramientas y técnicas que permitan el análisis y la clasificación de

grandes volúmenes de datos textuales. La clasificación de sentimientos es un tema relevante en el análisis de opiniones y evaluaciones de productos y servicios. Un caso en particular es el análisis de sentimientos en críticas de cine el cual es de gran interés en la industria del entretenimiento.

A lo largo de los años, se han propuesto varios métodos para abordar este problema, incluyendo enfoques basados en reglas, aprendizaje automático y redes neuronales. A pesar de los avances, todavía existen desafíos en la identificación precisa de la polaridad de los sentimientos.

En este contexto, el presente estudio propone un nuevo método basado en aprendizaje automático para la clasificación de sentimientos en críticas de cine en español. El método se evaluó en el corpus Muchocine y se comparó con los métodos del estado del arte. Se utilizó una combinación de características basadas en n-gramas y la técnica de clasificación *SVM primal*. En línea con estudios previos, se reconoce la importancia de la selección de características y del algoritmo de clasificación en el rendimiento del modelo [1, 2]. Por otra parte, se investigó el efecto del *stemming* en la *precision* de la clasificación y se encontró que empeora los resultados.

Los resultados del presente estudio indican que el método propuesto logra una *precision*, *recall* y *f-measure* superiores a los resultados expuestos en el estado del arte [2, 3]. Además, se encontró que la combinación de características basadas en n-gramas mejora el rendimiento del clasificador. Estos hallazgos sugieren que el método propuesto podría ser útil en la industria del entretenimiento para la evaluación de críticas de cine.

En cuanto a trabajos futuros, se sugiere la exploración de modelos basados en redes neuronales recurrentes y la utilización de modelos pre-entrenados como Bert para la clasificación de sentimientos en críticas de cine en español. También se propone la optimización de *SVM* mediante algoritmos genéticos y la evaluación del modelo utilizando *word2vec*. Asimismo, se sugiere la aplicación y evaluación del método propuesto en otros corpus de críticas de cine en español.

En resumen, el presente estudio aborda el problema de la clasificación de sentimientos en críticas de cine en español y propone un nuevo método basado en aprendizaje automático. Los resultados indican que el método propuesto supera los métodos del estado del arte y que la combinación de características basadas en n-gramas mejora el rendimiento del clasificador. Estos hallazgos podrían ser útiles en la industria del entretenimiento para la evaluación de críticas de cine en español.

2. Materiales y método

2.1. Materiales

Los materiales utilizados para los experimentos fueron:

- Corpus de críticas de cine: El corpus utilizado en este estudio es Muchocine, que consta de 5500 críticas de cine en español, previamente etiquetadas como positivas, negativas o neutrales. Estas críticas fueron recopiladas de diferentes fuentes en línea, como blogs y sitios web de reseñas de películas. Este corpus ha sido ampliamente utilizado en la literatura para evaluar el rendimiento de diferentes métodos de análisis de sentimientos en español [2,

3]. La importancia de este corpus radica en el hecho de que es el único corpus en español que se ha utilizado específicamente para el ámbito de críticas de cine. Además, se utilizaron algunas herramientas de procesamiento de lenguaje natural como la biblioteca Natural *Language Toolkit* (NLTK) [4] y la herramienta *Porter Stemmer* [5] para el preprocesamiento de los textos.

- Procesamiento de texto: Se utilizó la biblioteca NLTK para preprocesar el corpus de críticas de cine. Este preprocesamiento incluyó la eliminación de puntuación, la conversión a minúsculas, la eliminación de stopwords y la lematización.
- Características de texto: Para representar las críticas de cine, se utilizó la técnica de representación de n-gramas, que ha demostrado ser efectiva en la detección de polaridad en textos [6], pero en este trabajo se propone la combinación de n-gramas de palabras (unigramas, bigramas y trigramas) con la finalidad de conocer si el combinar n-gramas mejora los resultados obtenidos, tal como se ha visto en el trabajo de [7] para la tarea de atribución de autoría donde se logran grandes resultados al combar n-gramas.
- Clasificadores de texto: Se evaluaron diferentes implementaciones de SVM, como SVM lineal y SVM primal, para conocer si se pueden mejorar aún más los resultados obtenidos [8].
- Evaluación del modelo: Para evaluar el rendimiento del modelo, se utilizaron las métricas estándar de precision, recall y f-measure [2]. También se utilizó la matriz de confusión para evaluar la capacidad del modelo para distinguir entre las críticas positivas y negativas.

2.2. Método

En esta sección se describe detalladamente el proceso seguido para desarrollar el método propuesto en este trabajo de investigación. El objetivo principal de este método es el análisis de sentimientos en críticas de cine en español, utilizando como corpus de evaluación el conjunto de datos Muchocine.

Se utilizó una metodología basada en el proceso de minería de texto, que se divide en las siguientes etapas:

1. Preprocesamiento de datos: En esta etapa, se aplicaron diversas técnicas para limpiar y normalizar los datos. Se eliminaron signos de puntuación, caracteres especiales y se convirtieron todas las letras a minúsculas tal como se ha hecho en otros trabajos de análisis de sentimientos en español [9, 10].
2. Extracción de características: En esta etapa, se construyeron los modelos de representación de las críticas, a partir de las características que se consideran relevantes para la clasificación. Para ello se utilizaron los n-gramas, que han demostrado ser una técnica efectiva en el análisis de sentimientos en español [6]. Además, se utilizó la técnica de representación de bolsa de palabras (*bag of words*) para la construcción de las características [11]. Se utilizaron tres tipos de características: palabras, bigramas y trigramas. La combinación de estos n-gramas se convierte en el modelo de representación de cada crítica.

3. Entrenamiento del clasificador: En esta etapa, se entrenó el clasificador para que pudiera clasificar las críticas en positivas o negativas. Para el entrenamiento del clasificador se utilizó *SVM*. Este clasificador ha sido ampliamente utilizado en el análisis de sentimientos en español. Además, se utilizó la técnica de validación cruzada para evaluar el rendimiento de cada clasificador, tal como se ha hecho en otros trabajos de análisis de sentimientos en español.
4. Evaluación del rendimiento del clasificador: Se utilizaron diversas métricas, como *precision*, el *recall* y el *f-measure* [12]. Estas métricas han sido ampliamente utilizadas en la literatura para la evaluación del rendimiento de clasificadores en el análisis de sentimientos en español. Además, se realizaron pruebas de significación estadística para comparar los resultados obtenidos con los métodos del estado del arte, tal como se ha hecho en otros trabajos de análisis de sentimientos en español.
5. Análisis de resultados: En esta etapa, se analizaron los resultados obtenidos para determinar cuál es el mejor método de clasificación de sentimientos en críticas de cine en español en base a los resultados obtenidos.

3. Estado del arte

El análisis de sentimiento en críticas de películas ha sido un área de investigación popular en los últimos años, y se ha explorado ampliamente el uso de técnicas de aprendizaje automático para realizar esta tarea. En un estudio reciente realizado por [13] propone el uso de análisis de sentimiento para predecir el éxito o fracaso de una película a través de reseñas en línea utilizando minería de texto y la técnica de aprendizaje automático de *SVM*.

La principal dificultad en el análisis de sentimiento es la selección de palabras que indican sentimiento y la forma en que las personas expresan su opinión. El artículo también señala que los algoritmos de análisis de sentimiento pueden ser limitados debido a factores culturales, matices lingüísticos y contextos diferentes. Los resultados de la investigación anterior muestran que los métodos de aprendizaje automático, como *SVM* y *Naive Bayes*, han mejorado significativamente la precisión en la clasificación de las revisiones de películas en positivas o negativas.

El artículo "Análisis de sentimientos en Twitter para español" presenta un enfoque de *SVM* para el análisis de sentimientos en tweets en español. Los autores participaron en el taller TASS2013, que se centró en el idioma español, y utilizaron un corpus de tweets en español anotados con polaridad.

El artículo destaca los desafíos del análisis de sentimientos en Twitter, incluyendo el uso de frases no gramaticales, emoticonos, abreviaturas y jerga. En general, proporciona información sobre los desafíos del análisis de sentimientos en Twitter en español, y presenta un enfoque de *SVM* que logró los mejores resultados para las tareas de análisis de sentimientos del taller TASS2013 [14].

El artículo [15] presenta un enfoque de análisis de sentimientos de revisiones de películas utilizando *SVM* con el método de Ganancia de Información para la selección de características. El objetivo del estudio es mejorar la precisión de la clasificación de opiniones positivas y negativas en revisiones de películas. Los resultados muestran que *SVM* basado en Ganancia de Información logró una mayor precisión que el *SVM*

convencional en dos conjuntos de datos de revisiones de películas, alcanzando un aumento de precisión del 2.6% en el conjunto de datos de Cornell y un aumento de precisión del 0.166% en el conjunto de datos de Stanford.

El artículo también discute la importancia de la selección de características en el rendimiento de la clasificación de texto, y cómo la Ganancia de Información ha demostrado ser el mejor método de selección de características en comparación con otros algoritmos. Los autores creen que el estudio puede ayudar a los usuarios a tomar decisiones informadas sobre la calidad de las películas, y contribuir al desarrollo de la teoría relacionada con el análisis de sentimientos y la clasificación de texto.

En el campo de análisis de sentimientos en críticas de cine en español, el trabajo de Cruz [3] fue pionero. Para superar la falta de un corpus adecuado, los autores propusieron el corpus Muchocine, que consistía en 3878 críticas de cine extraídas del sitio web. Sin embargo, debido a las limitaciones en tiempo y hardware disponible en ese momento, el corpus no fue utilizado en su totalidad. El autor utilizó 400 críticas en total para experimentar con diferentes métodos, dividiéndolas en 200 críticas positivas y 200 críticas negativas, y dejando a un lado las críticas neutrales.

Utilizó bigramas de palabras, pero no realizó ninguna combinación como proponemos nosotros. Además, no utilizó un clasificador como SVM o redes neuronales, ya que no tenía los recursos para procesar todas las críticas. El autor logró obtener un resultado del 77%, lo que proporciona una base para comparar los resultados con el método propuesto. Aunque el autor solo experimentó con un conjunto reducido de críticas, su trabajo sigue siendo valioso para entender los desafíos y oportunidades del análisis de sentimientos en críticas de cine en español.

Este artículo [2] se enfoca en el análisis de sentimientos en textos en español, que es una tarea desafiante relacionada con la minería de texto y el procesamiento del lenguaje natural. Aunque hay trabajos actuales, la mayoría se centran en textos en inglés. La presencia del español en la web está aumentando, por lo que se han llevado a cabo algunos experimentos sobre un corpus de reseñas de cine en español.

En este trabajo se presentan varios experimentos utilizando cinco algoritmos de clasificación (*SVM, Naive Bayes, BBR, KNN, C4.5*). Los resultados obtenidos son muy prometedores y alentadores para continuar investigando en esta línea. Este artículo describe los experimentos realizados sobre un corpus de reseñas de cine en español, y se comparan los resultados con otros trabajos que han utilizado el enfoque semántico.

4. Experimentos y resultados

En esta sección comparamos los mejores resultados obtenidos por el método propuesto contra los del estado del arte, posteriormente mostramos una serie de experimentos que nos condujeron a lograr a estos resultados.

4.1. Comparación con los métodos del estado del arte

Se realizó una comparación entre los mejores resultados obtenidos por el método propuesto y los métodos del estado del arte previamente aplicados al corpus Muchocine, con el objetivo de evaluar su rendimiento. Los resultados se presentan en

Tabla 1. Comparación de los resultados del método propuesto (Estrada) y los métodos de Martinez [2] y Cruz [3].

Autor	Clasificador	Precision	Recall	F-measure
Estrada	SVM primal	91.94	89.06	90.48
Martinez	SVM Lineal	87.73	87.69	87.71
Martinez	Naive Bayes	84.08	84.01	84.04
Cruz	$SOv1$ - S	77.50	N/A	N/A
Cruz	$SOv2$ - NS	69.05	N/A	N/A

*N/A No se cuenta con la información, ya que no se evalúo con esta métrica.

la tabla 1. Se puede observar que el método propuesto logra una *precision*, *recall* y *f-measure* superior a los métodos del estado del arte [2, 3], demostrando que el método propuesto es una mejor opción para el análisis de sentimientos en críticas de cine.

Específicamente, el método propuesto logra una *precision* del 91.94%, un *recall* del 89.06% y una *f-measure* del 90.48%, lo que representa una mejora significativa en comparación con los resultados obtenidos por otros autores que utilizaron clasificadores *SVM lineal* y *Naive Bayes*.

Además, el método propuesto también supera a los resultados obtenidos por Cruz utilizando los clasificadores $SOv1$ - S y $SOv2$ - NS. Se concluye que la combinación de n-gramas es la clave para lograr un mejor rendimiento en la clasificación de textos, y que la elección del clasificador se vuelve secundaria cuando se utiliza una combinación de características en el modelo de representación.

A pesar de los resultados prometedores, todavía hay margen de mejora para alcanzar la meta de una precisión del 99.9%. Por lo tanto, se espera que este trabajo sirva como punto de partida para futuros estudios que busquen optimizar la precisión y aplicarlo en la vida real.

Es importante mencionar que, para llegar a estos resultados, se testearon diferentes variantes del método, como fue probar distintos *SVM*, tamaños de n-gramas, combinación de n-gramas y el uso de *stemming*. A continuación, se presentan estos experimentos y se comparan los resultados con las distintas variantes evaluadas.

4.2. Experimento 1: Comparación entre SVM primal, lineal y polinomial

En este experimento, se evaluaron las variantes de *SVM lineal*, *primal* y *polinomial* como clasificadores en el método propuesto, utilizando la combinación de unigramas + bigramas + trigramas como base de comparación. Los parámetros c=1 y épsilon=0.001 para cada variante. Los resultados obtenidos se presentan en la tabla 2.

Los resultados mostrados en la tabla 2 indican que el clasificador *SVM Primal* tuvo el mejor rendimiento, con una *precision* del 90.87% y un *f-measure* del 90.48%. El *SVM lineal* también obtuvo buenos resultados, con una *precision* del 89.73% y un *f-measure* del 89.24%. Por otro lado, el *SVM polinomial* tuvo un rendimiento inferior, con una *precision* del 84.79% y un *f-measure* del 83.74%. Estos resultados sugieren que el clasificador *SVM Primal* es la mejor opción para la clasificación de textos en este corpus específico.

Tabla 2 Comparación del rendimiento de las variantes de SVM.

Clasificador	Accuracy	Precision	Recall	F-measure
SVM Primal	90.87	91.94	89.06	90.48
SVM lineal	89.73	91.06	87.50	89.24
SVM polinomial	84.79	87.29	80.47	83.74

Tabla 3 Comparación entre diferentes tamaños de n-gramas de palabra.

Característica	Accuracy	Precision	Recall	F-measure
Unigramas	88.21	88.19	87.5	87.84
Bigramas	80.61	79.84	80.47	80.16
Trigramas	79.09	78.29	78.91	78.6

Tabla 4 Comparación de la combinación de las bolsas n-gramas.

Característica	Accuracy	Precision	Recall	F-measure
Unigramas+bigramas	88.97	88.98	88.28	88.63
Unigramas+trigramas	88.97	85.6	83.59	84.58
Bigramas+trigramas	85.17	79.23	80.47	79.84
Unigramas+bigramas +trigramas	90.87	91.94	89.06	90.48

4.3. Experimento 2: Comparación de rendimiento entre diferentes tamaños de n-gramas de palabra

En este experimento, se evaluó el rendimiento de tres modelos diferentes de n-gramas: unigramas, bigramas y trigramas, para identificar con cual se obtienen mejores resultados. La table 3 muestra los resultados en términos de las métricas *accuracy*, *precision*, *recall* y *f-measure*.

De acuerdo con los resultados, unigramas presenta una mayor *accuracy* con un 88.21%, seguido de bigramas con 80.61% y trigramas con 79.09%. En cuanto a la métrica *precision*, unigramas también presenta el mayor valor con 88.19%, seguido de bigramas con 79.84% y trigramas con 78.29%.

En cuanto al *recall*, unigramas presenta un valor superior a bigramas y trigramas con 87.5% en comparación con 80.47% y 78.91% respectivamente. Respecto a la métrica *f-measure*, unigramas nuevamente presenta un valor superior con 87.84%, seguido de bigramas con 80.16% y trigramas con 78.6%. En conclusión, los resultados indican que en general, el tamaño de unigramas es el que presenta un mejor rendimiento en las diferentes métricas evaluadas.

4.4. Experimento 3: Combinación de n-gramas como modelo de representación

En este experimento se combinan diferentes tamaños de n-gramas para crear una sola bolsa de n-gramas. En la tabla 4 se muestran los resultados de las diferentes

Tabla 5 Aplicación de *stemming* en la etapa de preprocesamiento.

Característica	Accuracy	Precision	Recall	F-measure
Sin stemming	90.87	91.94	89.06	90.48
Con stemming	87.45	88.62	85.16	86.85

combinaciones posibles entre unigramas, bigramas y trigramas. Con base en los resultados obtenidos, se puede afirmar que la combinación de unigramas + bigramas + trigramas es la que proporciona el mejor rendimiento, con una accuracy del 90.87%, una *precision* del 91.94%, un *recall* del 89.06% y un *f-measure* del 90.48%. Estos resultados demuestran que la utilización de tres tamaños de n-gramas es más efectiva que la combinación de solo dos tamaños.

En segundo lugar, se encuentra la combinación de unigramas + bigramas, con una *accuracy* del 88.97%, una *precision* del 88.98%, un *recall* del 88.28% y un *f-measure* del 88.63%. Aunque estos resultados son buenos, no superan a los obtenidos por la combinación de tres tamaños de n-gramas.

Las combinaciones de unigramas y trigramas, y bigramas y trigramas, presentaron resultados similares, pero aún están por debajo de la mejor combinación de n-gramas. Esto indica que la combinación de tres tamaños de n-gramas es la que proporciona el mejor rendimiento en este corpus.

En conclusión, la combinación de tres tamaños de n-gramas puede ser de gran utilidad para mejorar la *precision* de modelos de lenguaje en el futuro. Los resultados obtenidos en este estudio proporcionan una guía útil para la elección de la combinación de n-gramas en la construcción de modelos de clasificación de texto.

4.5. Experimento 4: Impacto de la técnica de stemming

En este experimento se evaluó el impacto de la técnica de *stemming* en el rendimiento del método propuesto. Para llevar a cabo la comparación, se utilizó el modelo de n-gramas combinado que previamente había obtenido los mejores resultados, es decir, la combinación de unigramas, bigramas y trigramas de palabra. Los resultados obtenidos utilizando *stemming* en la etapa de preprocesamiento se compararon con los resultados obtenidos sin utilizar esta técnica.

Los resultados obtenidos se presentan en la tabla 5. Se puede observar que la técnica de *stemming* no mejora los resultados del clasificador, al contrario, se obtuvieron peores resultados en todas las métricas. Además, la técnica de *stemming* conlleva más tiempo de ejecución en la etapa de preprocesamiento. Es importante destacar que esta comparación se limitó únicamente al modelo de n-gramas combinado que previamente había obtenido los mejores resultados.

5. Conclusiones y trabajo a futuro

La discusión de los resultados obtenidos en este estudio revela varias conclusiones importantes. En primer lugar, se pudo demostrar que el método propuesto en este trabajo logró un desempeño superior en términos de *accuracy*, *precision*, *recall* y f-

measure en comparación con los métodos del estado del arte evaluados en el corpus Muchocine.

Esto indica que la combinación de características basadas en n-gramas puede mejorar significativamente la *precision* del análisis de sentimientos en críticas de cine. Además, se observó que el rendimiento del clasificador *SVM Primal* fue ligeramente superior al del clasificador *SVM lineal* en todas las métricas evaluadas.

Otra conclusión importante que se puede extraer de los resultados obtenidos es que el uso de *stemming* en la etapa de preprocesamiento no mejoró los resultados del clasificador, sino que en realidad empeoró los resultados. Este hallazgo es consistente con los estudios del estado del arte y sugiere que el *stemming* no es una técnica efectiva para mejorar el análisis de sentimientos en críticas de cine.

Además, se observó que las diferencias en los resultados obtenidos para cada métrica de evaluación (*precision*, *recall* y *f-measure*) fueron relativamente pequeñas. Esto se debe en parte a la cantidad similar de críticas positivas y negativas evaluadas. Sin embargo, estos resultados también sugieren que las métricas de evaluación pueden ser insuficientes para medir la efectividad del análisis de sentimientos en críticas de cine, y que pueden ser necesarias medidas adicionales de desempeño.

En cuanto a las aportaciones de este estudio, se propone un nuevo método para el análisis de sentimientos en críticas de cine, que logra mejores resultados que los métodos del estado del arte evaluados en el corpus Muchocine. Además, se demuestra que la combinación de n-gramas es mejor que no combinar n-gramas, lo que puede abrir el camino a nuevas investigaciones donde se puedan combinar distintos tipos de características.

Finalmente, es importante destacar que este estudio tiene algunas limitaciones. Por ejemplo, se utilizó un único corpus de datos Muchocine y se limitó el análisis a las críticas de cine en español. Por lo tanto, es posible que los resultados no sean generalizables a otros idiomas o géneros de películas.

Además, se utilizó un conjunto de características relativamente limitado, lo que puede limitar la *precision* del análisis de sentimientos. Se necesitan investigaciones adicionales para evaluar la generalización de los resultados obtenidos en este estudio y para explorar la utilidad de otros conjuntos de características para el análisis de sentimientos en críticas de cine.

Referencias

1. Turney, P.: Thumbs Up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In: Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pp. 417–424 (2002) doi.org/10.3115/1073083.1073153
2. Martínez-Cámara, E., Martín-Valdivia, M. T., Ureña-López, L. A.: Opinion classification techniques applied to a Spanish corpus. In: Muñoz, R., Montoyo, A., Métais, E. (eds.) Natural Language Processing and Information Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 169–176 (2011) doi: 10.1007/978-3-642-22327-3_17
3. Cruz, F. L., Troyano, J. A., Enriquez, F., Ortega, J.: Clasificación de documentos basada en la opinión: experimentos con un corpus de críticas de cine en español. Procesamiento de Lenguaje Natural, no. 41, pp. 73–80 (2008)
4. Bird, S., Klein, E., Loper, E.: Natural language processing with Python: Analyzing text with the natural language toolkit. O'Reilly Media, Inc. (2009)

5. Porter, M. F.: An algorithm for suffix stripping. Program: Electronic Library and Information Systems, vol. 14, no. 3, pp. 130–137 (1980) doi: 10.1108/eb046814
6. Gamon, M., Aue, A., Corston-Oliver, S., Ringger, E.: Pulse: Mining customer opinions from free text. In: Famili, A. F., Kok, J. N., Peña, J. M., Siebes, A., Feelders, A. (eds.) Advances in Intelligent Data Analysis VI, Springer Berlin Heidelberg, pp. 121–132 (2005) doi: 10.1007/11552253_12
7. Sari, Y., Stevenson, M., Vlachos, A.: Topic or Style? Exploring the most useful features for authorship attribution. In: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, pp. 343–353. (2018) doi: 10.17863/CAM.78746
8. Machova, K., Mach, M., Vasilko, M.: Comparison of machine learning and sentiment analysis in detection of suspicious online reviewers on different type of data. Sensors, vol. 22, no. 1, pp. 155 (2021) doi: 10.3390/s22010155
9. Pérez-Rosas, V., Banea, C., Mihalcea, R.: Learning sentiment lexicons in Spanish. In: Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12), vol. 12, pp. 73 (2012)
10. Tellez, E. S., Miranda-Jiménez, S., Graff, M., Moctezuma, D., Siordia, O. S., Villaseñor, E. A.: A case study of Spanish text transformations for twitter sentiment analysis. Expert Systems with Applications, vol. 81, pp. 457–471 (2017) doi: 10.1016/j.eswa.2017.03.071
11. Qader, W. A., Ameen, M. M., Ahmed, B. I.: An overview of bag of words: Importance, implementation, applications, and challenges. In: 2019 International Engineering Conference (IEC), pp. 200–2004 (2019) doi: 10.1109/IEC47844.2019.8950616
12. Sebastiani, F.: Machine learning in automated text categorization. ACM Computing Surveys, vol. 34, no. 1, pp. 1–47 (2002) doi: 10.1145/505282.505283
13. Priya, B.: Sentiment analysis for online movie reviews using SVM. International Journal of Scientific research and Review, vol. 7, no. 5, pp. 804–811(2019)
14. Pla, F., Hurtado, L. F.: Sentiment analysis in twitter for Spanish. In: Métais, E., Roche, M., and Teisseire, M. (eds.) Natural Language Processing and Information Systems, Springer International Publishing, Cham, pp. 208–213 (2014) doi: 10.1007/978-3-319-07983-7_27
15. Maulana, R., Rahayuningsih, P. A., Irmayani, W., Saputra, D., Jayanti, W. E.: Improved accuracy of sentiment analysis movie review using support vector machine based information gain. Journal of Physics: Conference Series, vol. 1641 (2020). doi: 10.1088/1742-6596/1641/1/012060

ISSN 1870-4069

Sistemas de clasificación aplicados a la detección de paráfrasis

Francisco Fernando López Ponce,
Gerardo Sierra, Gemma Bel Enguix

Universidad Nacional Autónoma de México,
Instituto de Ingeniería, Grupo de Ingeniería Lingüística,
México

francisco.lopez.ponce@ciencias.unam.mx,
{gsierram, gbele}@iingen.unam.mx

Resumen. En este artículo analizaremos una de las múltiples tareas del PLN que es la clasificación de textos, en particular la clasificación binaria aplicada a la detección de paráfrasis. Implementamos una variedad de modelos de clasificación basados en el aprendizaje automático, desde modelos tradicionales como la regresión logística o Naive Bayes, hasta el estado del arte que son modelos de redes neuronales usando la arquitectura Transformer. Hacemos uso de distintas representaciones matemáticas del texto para realizar el preprocesamiento adecuado previo a la implementación de los modelos. Para el proceso de entrenamiento y evaluación usamos los datasets MRPC, donde los modelos se terminan por evaluar usando las métricas estándar para problemas de clasificación binaria como son Accuracy, Precision, Recall y F1, obteniendo una F1 máxima de 0.89.

Palabras clave: PLN, paráfrasis, redes neuronales, transformers, clasificación binaria.

Classification Systems Applied to Paraphrase Detection

Abstract. In this paper we analyze the NLP task of text classification, in particular binary classification applied to paraphrase detection. We implemented a myriad of classification models based on machine learning, from traditional models like logistic regression or Naive Bayes to state-of-the-art neural models based on the Transformer architecture. For feature extraction we use multiple mathematical representations of text as well as adequate preprocessing. When it comes to training and testing we used the MRPC as well as standard classification metrics such as Accuracy, Precision, Recall, and F1, obtaining a maximum F1 score of 0.89.

Keywords: NLP, paraphrase, neural networks, transformers, binary classification.

1. Introducción

La paráfrasis es el fenómeno que describe la similitud entre distintos textos que han sido modificados mediante sustituciones léxicas o reformulaciones estructurales [22], decimos que un texto es paráfrasis del otro si después de estas modificaciones se preserva el contenido dentro de los textos. Podemos observar ejemplos de este fenómeno en los pares 1.1, 1.2, y 2.1, 2.2.

1.1 El gato come feliz.

1.2 El minino come alegre.

1.3 El niño camina velozmente.

1.4 Rápidamente anda el infante.

Este fenómeno puede analizarse de manera profunda desde el ámbito lingüístico al definir distintos tipos de modificaciones que se hacen a un texto para llegar a una nueva instancia parafraseada. Si bien estas modificaciones, generalmente clasificadas como omisión, sustitución, modificación morfológica, y modificación en el orden, son herramientas que permiten analizar el contenido lingüístico de la paráfrasis, no presentan las mejores herramientas a la hora de implementar modelos computacionales que busquen analizar este fenómeno.

A nivel computacional este problema se trabaja como uno de clasificación binaria en donde un sistema computacional debería de ser capaz de determinar la existencia (o falta) de este fenómeno al comparar pares de textos. Generalmente este problema se simplifica trabajando con pares de oraciones para facilitar la creación e implementación de modelos entrenados con este tipo de datos.

La detección automática de paráfrasis es un tema de estudio de alta relevancia debido a sus aplicaciones como la detección de estilo o detección de plagio. En este artículo presentaremos una implementación de múltiples modelos computacionales que hacen uso del PLN y el aprendizaje automático para enfrentarse a este problema. Estos modelos nos terminan por generar un sistema de clasificación basado en diversas representaciones matemáticas del texto y algoritmos de aprendizaje automático.

Implementaremos modelos clásicos del aprendizaje, como los clasificadores a partir de regresión logística y Naive Bayes, modelos más avanzados como redes neuronales recurrentes con diferentes arquitecturas y capas, hasta el estado del arte que hacen uso de modelos pre-entrenados a partir de Transformers. En el capítulo 2 hablaremos de trabajos relacionados con la detección de paráfrasis, en el 3 describiremos a nivel matemático los modelos implementados, mientras que en el 4 desarrollaremos la implementación puntual de dichos modelos; concluimos en el 5 presentando los resultados, comparándolos entre ellos, y planteamos posibles trabajos a futuro.

2. Trabajos relacionados

Autores como Meshram [13] recopilan de manera superficial diversos métodos y artículos correspondientes a sistemas de detección automática. De una manera similar Mohamed [16] recopila y presenta modelos de aprendizaje automático divididos entre

los sistemas clásicos y los de aprendizaje profundo donde nos muestra los resultados de algunos modelos particulares. Por otra parte existen una plétora de artículos donde se presentan distintos modelos con arquitecturas y representaciones matemáticas del texto evaluados bajo el corpus de paráfrasis de Microsoft (MRPC). Al inicio de la década pasada Ji [9] presenta un modelo que logra optimizar el uso de sistemas estadísticos usando variaciones de tf-idf, evaluándose con el corpus MRPC y obteniendo Accuracy de 80 % y una F1 de 85 %.

Mientras que trabajos como el de Vrbanec [23] comparan modelos de embeddings como Word2Vec, GLoVE, o ELMO, obteniendo una F1 máxima de 81 % en el mismo corpus. Por otra parte el inicio de ésta década presenta modelos como Arase [1], un fine-tuning que permite la convergencia rápida de un modelo pre-entrenado con BERT [4].

Debido a la creación de distintos modelos pre-entrenados tenemos artículos que implementan y evalúan modelos como RoBERTa [12], XLM-R [2] y ALBERT [11], como lo hace Corbeil [3] haciendo ajustes al MRPC por medio del aumento de datos y obteniendo un F1 tan alto como 91 %.

La cercanía aparente entre los modelos estadísticos y los modelos neuronales, al igual que la constante creación de modelos pre-entrenados especializados, nos motiva a presentar implementaciones de distintos modelos al igual que distintas caracterizaciones matemáticas del texto, para ser entrenados y evaluados usando el MRPC.

3. Modelos

Al hablar de implementaciones de modelos de aprendizaje automático dentro del contexto del PLN tenemos que hacer una diferencia importante entre los algoritmos de clasificación y las representaciones matemáticas del texto. El aprendizaje automático se enfoca en encontrar tendencias numéricas a partir de múltiples algoritmos de enseñanza que hacen uso de optimizadores y funciones de pérdida para eficientizar su rendimiento, como contraste en el caso del PLN no se trabaja, inicialmente, con datos numéricos por lo que es necesario encontrar métodos de representación matemática del texto tales que las propiedades léxicas y morfológicas se preserven, así permitiendo que los resultados de estos modelos de aprendizaje se presten a una interpretación adecuada a nivel lingüístico. Presentaremos primero las representaciones matemáticas del texto.

3.1. Representación matemática del texto

Las representaciones que veremos obtienen desde valores numéricos hasta espacios vectoriales a partir de un texto, podemos notar que algunas representaciones sencillas necesitan solo un par de textos para implementarse mientras que algunas más complejas requieren un conjunto de textos (corpus) completo. Estos objetos matemáticos obtenidos son las características que vamos a usar a la hora de entrenar nuestro modelo.

Es importante notar que no todas las representaciones son compatibles con todos los distintos modelos de clasificación, como veremos, existen casos de representaciones que se crean a partir de un modelo particular y cuya implementación va de la mano con el modelo base.

Coeficientes y métricas. Los coeficientes de Dice y Jaccard, desarrollados originalmente en el contexto de la estadística, se prestan a una fácil aplicación al contexto del PLN, mientas que las distancias de edición (edit distances) son una familia de métricas sobre hilos que nos permiten obtener un valor de similitud a partir de un par de oraciones.

Para los primeros coeficientes consideremos A y B dos oraciones expresadas como conjuntos donde cada palabra de cada oración corresponde a un elemento del conjunto (existen implementaciones que consideran las letras y signos de puntuación en vez de las palabras como los elementos de los conjuntos, pero la definición es análoga), entonces:

– Definimos el **Coeficiente de Dice** como:

$$D(A,B) = \frac{2|A \cap B|}{|A| + |B|}. \tag{1}$$

– Definimos el **Coeficiente de Jaccard:** como:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}. \tag{2}$$

El concepto de distancia de edición ya es propio del PLN, busca determinar la similitud de dos hilos u oraciones en función de la cantidad de modificaciones que se deben de aplicar sobre el primero para obtener el segundo, estos cambios son inserción, eliminación y substitución.

A cada cambio se le puede otorgar un peso dependiendo de la complejidad lingüística, aunque normalmente se maneja con el mismo peso para todos los cambios. Si bien existen distintas variaciones que dependen de las modificaciones permitidas, normalmente se usa la distancia de Levenshtein [10] (mínima distanca de edición) al ser la más permisiva con pesos iguales para cada modificación.

Frecuencia de términos. Los siguientes modelos presentan una complejidad mayor debido a la influencia de la hipótesis distribucional del lenguaje, resumida elegantemente por la frase del lingüísta J. R. Firth: “conocerás una palabra por la compañía que tenga” [5]. Estos modelos generan un espacio vectorial en función de la frecuencia de las palabras dentro de un corpus para describir similitudes entre palabras y textos.

La representación vectorial base corresponde a la frecuencia de términos (tf). Dado un corpus C y un conjunto de palabras P, correspondiente al vocabulario del corpus, creamos una matriz $A \in M_{|P|x|C|}$ donde la entrada $x_{p_i c_j}$ corresponde a la cantidad de veces que se usa la palabra p_i en el texto c_j. Esta matriz nos permite crear dos vectores, considerando columnas obtenemos un vector característico del texto c_j en función del vocabulario P, mientras que considerando filas obtenemos un vector característico de la palabra p_i en función de los textos C [10].

Un problema con este modelo radica en que la frecuencia de las palabras es un valor sesgado que no encapsula el significado completo de una palabra. Además, debido a que los idiomas siguen la distribución de Zipf [24] con respecto al uso de palabras, existen casos con una frecuencia extremadamente alta como son los artículos, mientras otras que pueden aparecer una o dos veces en todo el corpus, esto genera representaciones vectoriales sesgadas a favor de aquellas palabras con alta frecuencia.

Para arreglar este problema se recalculan las entradas de cada vector en función de la cantidad de documentos en los que aparece dicha palabra con la finalidad de darle mayor peso a palabras exóticas y minimizar el impacto de palabras demasiado comunes, a este valor se le conoce como frecuencia inversa de documento o idf por sus siglas en inglés.

A este proceso se le llama peso tf-idf (tf-idf weighing) [19]. Una vez que obtenemos una representación vectorial de nuestro vocabulario calculamos similitudes entre palabras a partir de técnicas matemáticas, generalmente usando la distancia coseno entre vectores para determinar similitudes.

Formalmente consideremos una palabra p, un documento d, N la cantidad de documentos en el corpus, y df_t la cantidad de documentos en donde se encuentra la palabra t, entonces definimos la frecuencia de términos tf, y la frecuencia inversa de documentos idf como:

$$\mathrm{tf}_{p,d} = \log_{10}(\text{cantidad}(p, d) + 1), \tag{3}$$

$$\mathrm{idf}_t = \log_{10}\left(\frac{N}{df_t}\right). \tag{4}$$

Por lo que la entrada de cada vector en un modelo tf-idf está dada por:

$$c = tf_{p,d} \cdot idf_t. \tag{5}$$

Naturalmente, solo podemos comparar vectores del mismo tipo (texto con texto, y palabra con palabra) ya que no hay garantía de que las matrices $M_{|P|x|C|}$ sean cuadradas. Estas representaciones poseen ventajas sobre su contraparte estadística ya que permiten contextualizar las palabras y minimizan sesgos presentes en la variada frecuencia del uso de palabras, además con implementaciones computacionales podemos analizar textos de mayor longitud y complejidad.

No obstante estos modelos tienen limitaciones ya que un vocabulario puede generar vectores cuyas dimensiones sean demasiado altas, por ejemplo el corpus Brown [6] tiene aproximadamente un millón de palabras. Esto genera problemas computacionales y vectores con escasa información, una palabra que aparece una única vez va a ser representada por un vector de un millón de dimensiones que tiene una sola entrada no cero.

Word Embeddings. La idea subyacente de esta representación es similar a la de los modelos previos: crear un espacio vectorial que capture las similitudes lingüísticas de las palabras. Las diferencias radican en los algoritmos usados para crear los espacios, las dimensiones, y la eficiencia de los mismos. El primer modelo de word embeddings fue Word2Vec (w2v) [15] que en vez de usar frecuencia y pesos para ajustar los vectores, considera una palabra y una ventana de contexto sobre la palabra donde aplica

regresión logística, entrenándose con contexto real y contexto falso, este último creado de manera aleatoria. Esto hace que w2v no analice la cercanía de las palabras si no las probabilidades de que dada una palabra base a y una palabra objetivo b, b sí pertenezca al contexto de a.

Consideremos la oración .ª mí me gusta el pan con café y frutas", sea p = pan la palabra base, con ventana de contexto 2, y c_j las palabras dentro de la ventana del contexto de p. Obtenemos 4 ejemplos positivos de pares de palabras, mientras que creamos 4 ejemplos negativos, n_j, de manera aleatoria:

- $c_1 = (\text{pan}, \text{gusta})$,
- $c_2 = (\text{pan}, \text{el})$,
- $c_3 = (\text{pan}, \text{con})$,
- $c_4 = (\text{pan}, \text{cafe})$,
- $n_1 = (\text{pan}, \text{automovil})$,
- $n_2 = (\text{pan}, \text{musica})$,
- $n_3 = (\text{pan}, \text{ellos})$,
- $n_4 = (\text{pan}, \text{goloso})$.

A partir de conjuntos similares w2v logra maximizar la similitud de la palabra base y objetivo (p, c_j), y minimizar la similitud entre la palabra base y los ejemplos negativos (p, n_j). Los word embeddings logran encapsular el contexto de uso de las palabras, además de agruparlas en función de su similitud individual y de contexto global, por ejemplo palabras como pan, café, o desayuno, se encuentran en una vecindad cercana aunque su significado individual difiera.

Esto nos permite encontrar analogías de manera matemática, es decir operaciones como $'\text{Paris}' - '\text{Francia}' + '\text{Italia}' = '\text{Roma}'$, o $'\text{Rey}' - '\text{Hombre}' + '\text{Mujer}' = '\text{Reina}'$ tienen una consistencia y veracidad alta [14]. Existen múltiples modelos de embeddings que optimizan y ajustan distintos parámetros y que sobrepasan a w2v; sin embargo, están sujetos a los problemas inherentes del modelo.

Para empezar los vectores son estáticos es decir cada palabra está asignada a un valor único, generando un conflicto con la polisemia, así mismo la arquitectura los embeddings genera conflictos al analizar palabras fuera del vocabulario del corpus de entrenamiento, volviéndolo dependiente de las propiedades del corpus.

Además los embeddings son muy propensos a aprender sesgos dentro del corpus, sesgos sexistas o racistas se pueden observar en la inconsistencia de operaciones entre palabras al cambiar el género, o las cercanías que llegan a tener grupos sociales particulares con palabras negativas [25].

3.2. Modelos de aprendizaje automático

Hablaremos brevemente sobre los modelos de aprendizaje automático. Describiremos la forma en la que los modelos estadísticos analizan un valor de entrada (input) y lo clasifican, explicaremos distintas capas de redes neuronales recurrentes y acabaremos analizando el modelo Transformer.

La descripción que damos de estos modelos hace énfasis en las formas de clasificar un valor, por lo que evitamos hablar del proceso de entrenamiento y ajuste de pesos ya que se desvía del tema central del artículo y, si bien es sumamente importante para el aprendizaje automático, no es vital para el análisis de la paráfrasis.

Naive Bayes. Este es un clasificador probabilístico entonces dada una entrada i y un conjunto de clases objetivo C, Naive Bayes (NB) calculará la probabilidad de que dicho input pertenezca a cada una de las clases posibles y regresará la clase con mayor probabilidad. En términos matemáticos la clase $c \in C$ que va a regresar NB es:

$$c = \underset{c_j \in C}{\text{máx}}\ P(c_j|i), \tag{6}$$

Lo interesante de NB es el manejo que hace de la probabilidad (4), y una supusición bastante fuerte durante el manejo algebráico de la misma, primero expresa $P(c_j|i)$ en términos de la regla de Bayes para probabilidades condicionales, luego como i es fijo y los valores son positivos podemos ignorar el denominador, y considerando $i = \alpha_1, \alpha_2, ..., \alpha_k$ en función de sus características llegamos a la expresión:

$$P(c_j|i) = \frac{P(i|c_j)P(c_j)}{P(i)} = P(i|c_j)P(c_j) = P(\alpha_1, \alpha_2, ..., \alpha_k|c_j)P(c_j), \tag{7}$$

NB obtiene su nombre a partir de la suposición ingenua (naive) de que las probabilidades condicionales $P(\alpha_l|c_j)$ son todas independientes. Por último, para evitar problemas computacionales con valores numéricos muy grandes se ajusta aplicando el logaritmo:

$$c = \underset{c_j \in C}{\text{máx}}\ P(c_j) \prod_{\alpha_l \in \alpha} P(\alpha_l|c_j) = \underset{c_j \in C}{\text{máx}}\ \log P(c_j) \sum_{\alpha_l \in \alpha} \log P(\alpha_l|c_j), \tag{8}$$

Regresión logística. Si bien este modelo sigue siendo un clasificador probabiístico en contraste con NB la regresión logística (RL) no busca caracterizar las clases en función de las probabilidades de pertenencia, más bien busca encontrar elementos que diferencien a las clases sin necesariamente aprender las propiedades particulares de cada clase.

La RL no trabaja en un inicio con probabilidades, al principio del entrenamiento genera un vector de pesos a partir del conjunto de aprendizaje, el peso se asocia al input y determina la relevancia dentro del problema para cada característica. Se genera una combinación lineal del vector de pesos ($\mathbf{p}$), el input ($\mathbf{i}$) y el factor de sesgo (s) para posteriormente calcular la probabilidad de clasificación dado un input:

$$z = (\mathbf{p} \cdot \mathbf{i}) + s, \tag{9}$$

Como la RL es probabilística necesitamos asegurarnos que $z \in (0, 1)$, para esto se aplica la función logística sobre z y, dependiendo de la cantidad de clases sobre las que queramos clasificar, ajustamos haciendo softmax:

$$P(y = 1|i) = \sigma(z) = \frac{1}{1 + e^{(-z)}} = \frac{1}{1 + e^{(-((\mathbf{p} \cdot \mathbf{i}) + s)}}, \tag{10}$$

En el caso de la clasificación binaria no es necesario ajustar con softmax ya que podemos encontrar las dos expresiones necesarias para las dos clases recordando que $1 - \sigma(x) = \sigma(-x)$ y que $P(y = 1) + P(y = 0) = 1$, por lo que:

$$P(y = 0) = 1 - \sigma(z) = 1 - \frac{1}{1 + e^{-z}} = \frac{e^{(-((\mathbf{p}\cdot\mathbf{i})+s)}}{1 + e^{(-((\mathbf{p}\cdot\mathbf{i})+s)}}, \tag{11}$$

Ya que tenemos las probabilidades determinamos un umbral de clasificación y clasificamos en función de esto, si bien el umbral estándar es 0.5 se puede modificar si se logra mejorar el rendimiento del modelo.

Redes neuronales. Se plantearon a la par que NB y RL, pero no fue hasta la última década, con la inovación de los Transformers, que las redes neuronales han logrado superar a los modelos estadísticos. Una red neuronal está compuesta por unidades neuronales sencillas (neural units) conectadas donde, dado un valor de entrada, se realizan operaciones, se produce un valor de salida, y se conecta con otra unidad para ser modificada hasta obtener un valor final tras recorrer toda la red.

El modelo más sencillo es una red prealimentada (feedfoward network), es decir una red donde las capas no generan ciclos entre ellas, y donde todas las unidades entre capas adyacentes están conectadas. Sin embargo nosotros usaremos modelos más complejos como son los recurrentes que usan ciclos entre capas, permitiendo múltiplies ajustes de un valor en una misma capa y a la preservación de información a lo largo del procesamiento.

Otro modelo que usaremos surge a partir del concepto de atención, un mecanismo que logra optimizar el proceso recurrente en función de las características más relevantes de las entradas, este modelo es el Transformer y los modelos preentrenados a partir de esta arquitectura.

Las unidades sencillas que crean estas redes se comportan de una manera similar a la RL, cada unidad posee un vector de pesos y un sesgo, en combinación con el input se hace una combinación lineal como en (7), posteriormente se aplica una función no lineal al valor obtenido, como en (8), esta función no necesariamente tiene que ser la logística, ejemplos de dos funciones de activación distintas son:

$$\tanh(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}, \tag{12}$$

$$\mathrm{ReLU}(z) = \mathrm{máx}(z, 0). \tag{13}$$

La agrupación de estas unidades en capas permite un mejor rendimiento, una red neuronal está formada por 3 tipos de capas, el primero es la capa de entrada donde se presentan los valores iniciales, el segundo son las capas ocultas donde se realiza la mayoría del procesamiento (una red puede tener múltiples capas ocultas), y finalmente la capa de salida donde se obtiene el resultado final que dependerá del problema en cuestión, para clasificación binaria obtendremos un valor probabilístico de pertenencia en una clase u otra. Las redes recurrente permiten revisitar una capa durante el procesamiento agregando un factor temporal a la forma de analizar un input, esto permite almacenar memoria sobre estados previos, es decir información previa afecta los cálculos inmediatos.

Estas redes son capaces de apilar múltiples capas para mejorar su rendimiento, si bien puede aumentar el tiempo de entrenamiento, este análisis tiende a entender mejor el input en distintos niveles de profundidad.

Además las redes recurrentes optimizan la apilación de capas gracias a la bidireccionalidad, es decir dado un input apilamos dos capas recurrentes donde la segunda capa procesa el input en sentido inverso, el valor de salida está dado por una operación sobre los outputs de cada entrada de la capa bidireccional.

[18] Aunque las redes recurrentes permiten un manejo creativo de la información, son susceptibles al problema del desvanecimiento del gradiente ya que las operaciones dentro del entrenamiento pueden llevar a que el gradiente tienda a cero, como ajuste se usan capas llamadas Long Short-Term Memory (LSTM) [8]. Una capa LSTM es capaz de discernir entre información que no será relevante en un futuro (y descartar la misma), además de preservar la información que sí se considera relevante al igual que adecuarla para usos futuros.

Se logra esto mediante el uso de compuertas que permiten el flujo de información entre capas, llamadas compuertas de olvido, suma, y salida (forget gate, add gate, output gate), estas compuertas clasifican la información en función de operaciones secuenciales y funciones de activación como $\tanh$, generando como salida un vector que se usará en futuras capas al igual que un vector de contexto (VC). LSTM no solo presenta beneficios conceptuales si no que logra resolver el problema del gradiente, por lo que estas capas normalmente se usan sobre la arquitectura base de una red recurrente.

Como buen modelo matemático las redes recurrentes presentan fallas, en particular a la hora de implementarse en modelos codificador-decodificador (encoder-decoder) [20], ya que el vector de contexto obtenido por el codificador no resume de manera óptima la información de todo un input, además la conexión entre estas dos secciones se ve ralentizada por la unicidad de este vector.

La solución a estos problemas son los mecanismos de atención que permiten más conexiones entre el encoder y decoder generando un acceso continuo al VC de cada capa oculta, y evitando el sesgo proveniente del VC final ya que en cada paso se logra priorizar el contexto más relevante para el problema.

Modelos Preentrenados. A partir de la atención se crea el modelo Transformer [21] y como consecuencia los modelos preentrenados. Los mecanismos de atención permiten comparar elementos y determinar sus niveles de relevancia dado un contexto, los Transformers son capaces de replicar esto sobre un solo input al tener acceso a esta relevancia de cada valor del input desde el inicio hasta el punto en curso, a esta variación se le conoce como auto-atención (self-attention).

Es importante notar que un Transformer no es una red recurrente con auto-atención, estos modelos están compuestos por bloques Transformer (Transformer Blocks) que hacen uso de capas con auto-atención, capas prealimentadas, y capas de ajuste para optimizar el entrenamiento.

Además al trabajar con múltiples bloques con distintos parámetros es posible que un modelo logre aprender diferentes relaciones sin la necesidad de aumentar las dimensiones de una red. Lo increíble de los Transformers radica en que entre más información esté disponible para su entrenamiento, mejores van a ser las implementaciones del modelo.

Tabla 1. Parámetros de regresión logística.

Métrica	C	Penalty	Solver
Dice	1	None	saga
Jaccard	1	None	LBFGS
Levenshtein	1	None	saga
tf	1	None	saga
tf-idf	100	None	saga
Multi	10	L1	saga

Esto introduce la idea de los modelos preentrenados, que radica en entrenar un Transformer con una alta cantidad de información, en el caso del PLN textual, y una vez obtenido el modelo final hacer un segundo entrenamiento con un corpus más específico al problema en cuestión.

Recordando los modelos codificador-decodificador existen distintas formas de preentrenar un modelo de PLN dependiendo del objetivo que se busque, si se busca dar una representación del texto, dígase un word embedding, se usa un modelo encoder, en caso de que se busque generar texto se usa un modelo decodificador, y en caso que se quiera hacer las dos tareas podemos usar los modelos codificador-decodificador. BERT es un codificador, GPT (en sus múltiples versiones) es un decodificador, y T5 es un modelo mixto.

Las ventajas del preentrenamiento es que se necesita entrenar una sola vez para obtener un modelo base que rinde de manera eficiente en una gran variedad de problemas de PLN, estos modelos son ajustados dependiendo del problema que se esté tratando de resolver. Nosotros haremos este ajuste para la detección de paráfrasis.

4. Implementación

Primero describiremos las especificaciones técnicas de la computadora en donde se realizaron los experimentos, mencionaremos las paqueterías usadas, el preprocesamiento aplicado al corpus, y finalmente describiremos todos los modelos con sus parámetros correspondientes dividiéndolos en 3 secciones.

Nuestros experimentos se realizaron usando Python 3.9 desde un entorno de Anaconda, en una computadora con Windows 10 Pro de 64 bits, 32GBs de RAM, procesador i7-12700K, y una tarjeta NVIDIA RTX 3060.

La paqutería usada para Regresión Logística y Naive Bayes fue SKLearn, mientras que para las redes neuronales se usó TensorFlow, los modelos preentrenados se obtuvieron desde HuggingFace. Para el preprocesamiento se eliminaron signos de puntuación, todas las letras mayúsculas se convirtieron en minúsculas, y se preservaron los números. Las representaciones usadas varían dependiendo del modelo en cuestión, veamos los detalles.

Naive Bayes y Regresión. Entrenamos estos dos modelos usando las métricas de Dice, Jaccard, vectores tf y vectores tf-idf, primero analizando cada característica de manera individual y por último combinando las 4 características en un solo modelo. Para obtener los parámetros adecuados de la regresión se realizó un GridSearch sobre el conjunto de entrenamiento, no hay parámetros de NB ya que el modelo GaussianNB

de SKLearn se implementa de manera directa. En la tabla siguiente se pueden observar los parámetros usados en cada modelo de regresión logística, dependiendo de las cada métrica usada.

Redes Neuronales. Implementamos 4 redes siamesas con diferentes arquitecturas base. Una red siamesa está compuesta por dos redes idéndticas que procesan las oraciones de manera independiente, posteriormente realizan una concatenación de los valores respectivos, usan una capa densa y clasifican en función esta última capa. Las arquitecturas bases que usaremos son:

– Red 1: LSTM ,
– Red 2: biLSTM,
– Red 3: Doble LSTM,
– Red 4: Doble biLSTM.

Como parte del preprocesamiento hicimos embeddings locales de dimensión $n = 25$, las funciones de activación de cada capa oculta fueron ReLU, mientras que para el output se consideró $\tanh$. Recordando que tenemos un problema de clasificación binaria usamos como función de pérdida Binary Cross-Entropy (BCE), mientras que el optimizador fue Adam para las 4.

Modelos pre-entrenados. Usamos 3 modelos preentrenados obtenidos de HuggingFace: BERT [4], DistilBERT [7], all-Mini [17]. A partir de estos hicimos un fine-tuning usando el MRPC, al ser modelos neuronales ajustamos nuevamente algunos hiperparámetros para igualar los modelos neuronales previos, es decir la pérdida fue BCE mientras que el optimizador fue Adam.

5. Resultados

Al trabajar con problemas de clasificación binaria se tienen métricas definidas que resumen el comportamiento de un modelo dedicado a este problema. Las 4 métricas que usaremos serán Accuracy, Precision, Recall, y F1 que se calculan en función de 4 clasificaciones de las predicciones de un modelo.

Recordemos que como nuestro problema es uno de aprendizaje supervisado, dado un input sabemos el valor real de dicho input; en el contexto de pares de oraciones significa que dadas dos oraciones sabemos a priori si las oraciones son o no son paráfrasis; diremos que un par de oraciones pertenecen a la clase positiva si sí son paráfrasis, en caso contrario diremos que pertenecen a la clase negativa. En función de esto definimos 4 categorías de predicciones de un modelo.

– **Verdadero Positivo (TP)**: Una predicción verdadera de un par de oraciones pertenecientes a la clase positiva.
– **Verdadero Negativo (TN)**: Una predicción falsa de un par de oraciones pertenecientes a la clase negativa.
– **Falso Positivo (FP)**: Una predicción verdadera de un par de oraciones pertenecientes a la clase negativa.
– **Falso Positivo (FN)**: Una predicción falsa de un par de oraciones pertenecientes a la clase positiva.

Tabla 2. Regresión logística.

Métrica	Acc	Recall	Precision	F1
Dice	0.69	0.69	0.67	0.62
Jaccard	0.69	0.69	0.68	0.65
Levenshtein	0.67	0.68	0.68	0.65
tf	0.72	0.72	0.7	0.69
tf-idf	0.72	0.72	0.71	0.7
Multi	**0.73**	**0.73**	**0.72**	**0.72**

Tabla 3. Naive Bayes.

Métrica	Acc	Recall	Precision	F1
Dice	0.67	0.67	0.67	0.55
Jaccard	0.69	0.69	0.67	0.63
Levenshtein	0.67	0.67	0.68	0.65
tf	**0.72**	**0.72**	0.7	0.69
tf-idf	**0.72**	**0.72**	0.71	0.7
Multi	0.71	0.71	**0.72**	**0.72**

Tabla 4. Redes neuronales.

Modelo	Acc	Recall	Precision	F1
LSTM	0.73	**0.88**	0.71	**0.79**
biLSTM	**0.74**	0.75	0.72	0.74
2xLSTM	**0.74**	0.74	0.73	0.73
2xbiLSTM	0.73	0.71	**0.74**	0.72

Tabla 5. Preentrados.

Modelo	Acc	Recall	Precision	F1
BERT	0.86	0.85	**0.94**	**0.89**
DistilBERT	**0.87**	**0.88**	0.9	**0.89**
all-Mini	0.77	0.82	0.81	0.81

Definimos:

$$\textbf{Accuracy} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}, \tag{14}$$

$$\textbf{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}, \tag{15}$$

$$\textbf{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \tag{16}$$

$$\textbf{F1} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}. \tag{17}$$

Ahora presentamos los resultados completos con las métricas de evaluación previamente mencionadas. Observamos que los modelos clásicos de clasificación funcionan mejor entre más características puedan analizar, aunque modelos como $\mathrm{tf} - \mathrm{idf}$ son suficientemente buenos de manera aislada. Sin embargo estos modelos se quedan considerablemente detrás de las implementaciones neuronales, incluso la peor red funciona a la par de los modelos clásicos.

Notamos que una red LSTM sencilla supera modelos más complejos y que una arquitectura demasiado rebuscada como la doble bidireccional puede ser contraproducente para problemas sencillos. Por otra parte los modelos preentrenados superan sustancialmente al resto, BERT y DistilBERT son 2 de los 10 modelos más usados en todo HuggingFace por lo que el alto rendimiento es esperado, aunque incluso un modelo no tan notorio como es all-Mini al estar diseñado para el análisis de oraciones supera a las redes neuronales tradicionales.

6. Conclusiones

Confirmamos que los modelos preentrenados son el estado del arte para la detección de paráfrasis automatizada, si bien los modelos tradicionales presentan resultados adecuados se ven opacados por modelos preentrenados. Por otra parte este estudio presenta posibilidades de crecimiento, futuros trabajos podrían buscar la implementación de estos modelos en corpus más especializados o no tan conocidos como lo es el MRPC. A niveles técnicos las implementaciones fueron simples, para expandir estos experimentos se pueden hacer búsquedas más extensas sobre parámetros particulares y distintas arquitecturas planteadas.

Agradecimientos. Este artículo fue financiado por los proyectos Detección automática de paráfrasis mediante métodos basados en la distribución de texto, A1-S-27780, y Grafos conceptuales para la construcción de diccionarios inversos en áreas de especialidad, CF-2023-G-64, del Consejo Nacional de Ciencia y Tecnología (CONACYT), al igual que el proyecto Desarrollo del sistema de gestión de corpus GECO, IT100822 del Programa de Apoyo a Proyectos de Investigación e Innovación Tecnológica (PAPIIT).

Referencias

1. Arase, Y., Tsujii, J.: Transfer fine-tuning of BERT with phrasal paraphrases. Computer Speech and Language, vol. 66, pp. 101164 (2021) doi: 10.1016/j.csl.2020.101164
2. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale (2019) doi: 10.48550/arXiv.1911.02116
3. Corbeil, J. P., Abdi Ghavidel, H.: BET: A backtranslation approach for easy data augmentation in transformer-based paraphrase identification context (2020) doi: 10.48550/arXiv.2009.12452
4. Devlin, J., Chang, M. W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, vol. 1, pp. 4171–4186 (2019) doi: 10.18653/v1/N19-1423
5. Firth, J. R.: A synopsis of linguistic theory 1930-1955. Studies in Linguistic Analysis, Special Volume of the Philological Society, vol. 1952-59, pp. 1–31 (1957)
6. Francis, W. N., Kucera, H.: Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US (1979) icame.uib.no/brown/bcm.html
7. HF Canonical Model Maintainers: Distilbert-base-uncased-finetuned-sst-2-english (2022) doi: 10.57967/HF/0181
8. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Computation, vol. 9, no. 8, pp. 1735–1780 (1997) doi: 10.1162/neco.1997.9.8.1735
9. Ji, Y., Eisenstein, J.: Discriminative improvements to distributional sentence similarity. pp. 891–896 (2013)
10. Jurafsky, D., Martin, J. H.: Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition. Prentice Hall, Pearson Education International (2009)

11. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A Lite BERT for self-supervised learning of language representations (2019) doi: 10.48550/ARXIV.1909.11942
12. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach (2019) doi: 10.48550/ARXIV.1907.11692
13. Meshram, S.: Review on NLP paraphrase detection approaches. International Journal of Innovative Science and Research Technology, vol. 4, no. 2, pp. 351–354 (2019)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. International Conference on Learning Representations (2013)
15. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, vol. 26 (2013) doi: 10.48550/ARXIV.1310.4546
16. Mohamed, I., Wael, H.: Exploring the recent trends of paraphrase detection. International Journal of Computer Applications, vol. 182, no. 46, pp. 1–5 (2019) doi: 10.5120/ijca2019918317
17. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using siamese BERT-networks (2019) doi: 10.48550/ARXIV.1908.10084
18. Schuster, M., Paliwal, K.: Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, vol. 45, no. 11, pp. 2673–2681 (1997) doi: 10.1109/78.650093
19. Sparck-Jones, K.: A statistical interpretation of term specificity and its application in retrieval. Document Retrieval Systems, vol. 28, no. 1, pp. 11–21 (1972) doi: 10.1108/eb026526
20. Sutskever, I., Vinyals, O., Le, Q. V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014) doi: 10.48550/arXiv.1409.3215
21. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I.: Attention is all you need (2017) doi: 10.48550/ARXIV.1706.03762
22. Vila, M., Martí, M. A., Rodríguez, H.: Paraphrase concept and typology. A linguistically based and computationally oriented approach. Sociedad Española para el Procesamiento del Lenguaje Natural, vol. 46 (2011)
23. Vrbanec, T., Meštrović, A.: Corpus-based paraphrase detection experiments and review. Information, vol. 11, no. 5, pp. 241 (2020) doi: 10.3390/info11050241
24. Yu, S., Xu, C., Liu, H.: Zipf's law in 50 languages: Its structural pattern, linguistic interpretation, and cognitive motivation (2018) doi: 10.48550/ARXIV.1807.01855
25. Zhao, J., Wang, T., Yatskar, M., Cotterell, R., Ordonez, V., Chang, K.-W.: Gender bias in contextualized word embeddings. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Association for Computational Linguistics, vol. 1, pp. 629–634 (2019) doi: 10.18653/v1/N19-1064

ISSN 1870-4069

Clasificación automática de preguntas en español en el dominio de comercio electrónico usando una red neuronal convolucional

Melissa A. De-León-Barrón[1], Ana B. Rios-Alvarado[1], Tania Y. Guerrero-Melendez[1], Heidy Marisol Marín-Castro[2], Jose L. Martinez-Rodriguez[3]

[1] Universidad Autónoma de Tamaulipas, Facultad de Ingeniería y Ciencias, México

[2] Universidad Autónoma de Tamaulipas, Consejo Nacional de Ciencia y Tecnología, México

[3] Universidad Autónoma de Tamaulipas, Unidad Académica Multidiciplinaria Reynosa, México

{melissa.barron, arios, tyguerre}@docentes.uat.edu.mx, {hmarisol, lazaro.martinez}@uat.edu.mx

Resumen. El comercio electrónico representa el proceso de comprar y vender productos y servicios por Internet. Uno de los principales retos que enfrenta el comercio electrónico es mejorar la experiencia del cliente, principalmente en el proceso de atención al cliente al momento de resolver preguntas de forma automática sobre los productos en venta o el proceso de compra. En este artículo se propone un modelo para la representación y clasificación de preguntas en español en el dominio de comercio electrónico. Se describe el diseño y desarrollo de un modelo de Aprendizaje Profundo basado en Redes Neuronales Convolucionales (CNN). La estrategia se centra en recuperar y asociar preguntas a una de las dos posibles clases: técnica (relacionados al producto) u operativa (relacionadas al proceso de compra). El conjunto de datos se representa mediante word embeddings utilizando las arquitecturas de Word2vec. Los resultados obtenidos demuestran que mediante la implementación de una red CNN se pueden desarrollar modelos que obtengan un buen desempeño para la tarea clasificación de preguntas en español.

Palabras clave: Minería de texto, aprendizaje profundo, CNN, word embedding, procesamiento del lenguaje natural.

Melissa A. De-León-Barrón, Ana B. Rios-Alvarado, Jose L. Martinez-Rodriguez

Automatic Classification of Spanish Questions in the E-Commerce Domain Using a Convolutional Neural Network

Abstract. E-commerce represents the process of buying and selling products and services over the Internet. One of the main challenges faced by e-commerce is to improve the customer experience, mainly in the customer service process when resolving questions automatically about the products for sale or the purchase process. This paper proposes a model for representing and classifying Spanish queries in the e-commerce domain. We describe the design and development of a Deep Learning model based on Convolutional Neural Networks (CNN). The strategy focuses on retrieving and associating questions to one of two possible classes: technical (related to the product) or operational (related to the purchasing process). The dataset is represented by word embeddings using Word2vec architectures. The results demonstrate that implementing a CNN network makes it possible to develop models that perform competently for the Spanish question classification task.

Keywords: Text mining, deep learning, CNN, word embedding, natural language processing.

1. Introducción

El surgimiento del Internet ha contribuido a hacer negocios y a mejorar el estilo de vida de las personas. Además, es uno de los principales requisitos para la existencia del comercio electrónico. El comercio electrónico es un concepto emergente que describe el proceso de comprar y vender productos, servicios e información vía Internet [3]. De acuerdo con un estudio realizado por Search Logistics [16], 2.140 billones de personas en todo el mundo compraron productos o servicios en línea en el 2021, eso es aproximadamente el 27 % de la población mundial.

Esto representa un aumento de 1.66 billones con respecto al 2016. Específicamente, en México durante el 2022 más de 63 millones de mexicanos adquirieron productos o servicios en Internet, lo que significa que nueve de cada 10 usuarios mayores de 18 años compró en línea [2]. En el transcurso del año 2020, una gran cantidad de países en todo el mundo atravesó por diversos tipos de medidas de restricción debido a la pandemia por el COVID-19, lo cual provocó que los consumidores y los negocios cambiaran drásticamente su estrategia de negocio.

Por esto, el Internet se convirtió en el principal medio de compra - venta, y de acuerdo a un estudio realizado por Think with Google reportó que la tendencia por realizar consultas sobre como realizar compras en línea creció en un 200 % en todo el mundo [1].

En la actualidad, existen una gran cantidad de plataformas de comercio electrónico operando todos los días alrededor del mundo, en donde una forma de comunicación entre los consumidores y los vendedores es a través de preguntas que los clientes potenciales realizan sobre los productos o servicios que ofrece dicha empresa.

Uno de los retos más importantes que se presentan en el comercio electrónico es la capacidad de los sistemas para responder automáticamente dichas preguntas.

Dentro de los sistemas de generación de respuestas automáticas existe un módulo de clasificación de preguntas. Es deseable que ante las preguntas en idioma español se pueda contar con modelos de representación y clasificación de textos cortos en ese idioma para apoyar la generación de respuestas automáticas.

Distintos proyectos han propuesto soluciones para la clasificación de textos cortos en varios contextos e idioma inglés, por ejemplo existen modelos para la clasificación de tuits [15], minería de opinión [17], sistemas de pregunta-respuesta [9], chatbots [4], entre otros. Sin embargo, son todavía pocos los modelos aplicados al español, debido a los retos que implica este lenguaje.

Por lo anterior, en este artículo se propone una estrategia para el diseño y desarrollo de un modelo de clasificación automática de preguntas en español para un dominio de comercio electrónico. El modelo esta basado en una arquitectura de Red Neuronal Convolucional (CNN) que utiliza las dos arquitecturas de Word2vec: Continuous Bag of words (CBOW) y Skip-gram para la representación del conjunto de datos. Para este estudio se evalúan diferentes configuraciones de la red CNN con el objetivo de encontrar la configuración óptima para alcanzar una mayor exactitud en la tarea de clasificación.

2. Trabajos relacionados

Con el crecimiento de las redes sociales, el comercio electrónico y la comunicación en línea, la clasificación de textos cortos se ha convertido en un tema de tendencia en años recientes [13], un texto corto se puede encontrar en diferentes contextos o idiomas como por ejemplo tuits [10], mensajes de chat [**?**], reseñas de productos [18] o preguntas de clientes [11]. Una de las principales características de los textos cortos es su longitud, los mensajes de texto tienen alrededor de 70 caracteres, los títulos de noticias contienen 30 caracteres y los tuits no cuentan con más de 280 caracteres [19].

Por otro lado, la tarea de responder preguntas (en inglés, Question Answering) es una tarea importante en el Procesamiento del Lenguaje Natural que consiste en brindar respuestas en lenguaje natural a preguntas. Por esto, un elemento esencial de los sistemas de preguntas-respuesta es la clasificación de preguntas, la cual es una tarea que consiste en asignar una clase a una pregunta.

Por ejemplo, si un sistema de pregunta-respuesta conoce que el nombre de una persona es un tipo de respuesta a la pregunta "¿Quién es el presidente de México?" la respuesta se restringirá a una clase de tipo nombre propio. La clasificación de preguntas está basada en la clasificación de texto debido a que son similares en algunos aspectos pero difieren claramente en que una pregunta es usualmente más corta [20].

Kulkarni et al. [9], presentan un estudio para la respuesta de preguntas de clientes en páginas de productos de un dominio de comercio electrónico. Uno de los módulos de este sistema es la tarea de clasificación de preguntas. Ellos propone implementar una red CNN (Convolutional Neural Network) y una representación de n-gramas. El sistema propuesto obtiene un resultado 66 % de precisión. En Kim [7] se presenta una serie de experimentos con redes convolucionales (CNN) y vectores pre entrenados de Word2vec por Mikolov et al. [12] con 100 billones de palabras de Google News.

Fig. 1. Diseño del modelo de clasificación de preguntas.

En este trabajo se muestra la arquitectura de la red CNN para el desarrollo de un modelo de clasificación de texto, dicha red neuronal consta de una arquitectura simple, ya que cuenta con una sola capa convolucional. El modelo reporta resultados de un 45 % de exactitud. Finalmente, en Kumar et al. [10] presentan un modelo de textos cortos basado en un enfoque de aprendizaje profundo. El conjunto de datos con el que se entrena dicho modelo está compuesto por reseñas de películas.

El modelo resultante reporta resultados de 99.07 % de exactitud para el entrenamiento y un 82.19 % de exactitud en la predicción. Con base en estos trabajos se observa que el aprendizaje profundo es efectivo para las tareas de clasificación de texto aunque para alcanzar dichos valores de exactitud se propone utilizar modelos pre entrenados de Word2vec.

3. Método propuesto

En esta sección se describe la metodología propuesta para la construcción de un modelo de clasificación de preguntas en español de un dominio de comercio electrónico. En la Figura 1 se muestra un diagrama con la metodología propuesta que consta de cuatro principales etapas: el preprocesamiento y representación de texto, el entrenamiento y configuración del modelo de clasificación, la evaluación del modelo se considera en el método propuesto.

3.1. Conjunto de preguntas

En esta tarea se tiene el objetivo de recolectar y construir el conjunto de preguntas en español. Las preguntas se extrajeron del sitio de comercio electrónico: Mercado Libre México[4], la cual fue seleccionada debido a que alberga una de las más relevantes plataformas de comercio en línea en todo América Latina y su crecimiento en el 2020 ha sido exponencial debido a la pandemia por COVID-19.

Por lo tanto, se pueden encontrar muchas preguntas realizadas por los clientes sobre las características de los productos, métodos de pago, devoluciones y reembolsos, servicio de envío, etc. El conjunto de preguntas se llevó a cabo mediante dos tareas:

– Recolección: Para descargar las preguntas de Mercado Libre México, se desarrolló un script de Python para realizar la extracción del contenido de la URL (Uniform Resource Locator) de diferentes productos contenidos en esta plataforma. Las preguntas extraídas fueron almacenadas en un archivo de texto.

[4] www.mercadolibre.com.mx/

Tabla 1. Ejemplo de estructura del conjunto de preguntas.

Pregunta	Clase
¿En cuántos días hacen el envío a Acapulco?	1
¿Qué colores tienes disponibles?	2
¿Hay disponible en XS?	1
¿Compatible con Huawei Mate 20	2
...	...

– Etiquetado: El conjunto de datos consta de 500 preguntas divididas en dos clases (250 de cada una): operativas y técnicas. Las preguntas operativas son sobre métodos de pago, envíos, devoluciones y reembolsos. Por otro lado, las preguntas técnicas se refieren a las características de los productos, por ejemplo, tamaño o color. El conjunto de preguntas contiene dos atributos: la pregunta y la etiqueta. Es importante mencionar que un experto en lingüística etiquetó manualmente el conjunto de datos. La Tabla 1 muestra un ejemplo de la estructura del conjunto de preguntas.

3.2. Preprocesamiento de texto

El preprocesamiento es una tarea crucial que permite la limpieza y preparación del texto para posteriormente llevar a cabo su clasificación. Los textos en línea generalmente contienen mucho ruido por lo que contar con datos debidamente procesados permite reducir el ruido en el texto y mejorar el rendimiento del clasificador [5].

Las preguntas del conjunto de datos poseen una longitud de hasta 485 caracteres, aunque se pueden encontrar preguntas con una longitud mínima de 15 caracteres. Estas preguntas pueden llegar a estar conformadas hasta por 90 términos, que correspondería a las preguntas más extensas.

Por otro lado, algunas preguntas del conjunto cuentan con solo dos términos, debido a las características de las preguntas, este estudio se aborda como una tarea de clasificación de textos cortos. Sin embargo, para las preguntas en español, se realizó una estrategia de preprocesamiento específica compuesta por los siguientes pasos:

1. Normalización. Esta tarea es necesaria para procesar texto, específicamente, textos informales. Esto debido a que el lenguaje utilizado en las plataformas de comercio electrónico difiere de los textos escritos formalmente, por lo que se puede encontrar un uso del idioma diferente. En esta tarea se aplicaron algunas técnicas de limpieza de texto:

– Convertir el texto a minúsculas.
– Segmentar el texto en palabras (tokens).
– Eliminar acentos.
– Eliminar signos de puntuación.
– Eliminar caracteres numéricos.
– Eliminar caracteres especiales. En el español también se consideran como caracteres especiales la ”ñ”, y ”ü”, estas se reemplazan por las letras ”n” y ”u” respectivamente.

– Eliminar caracteres repetidos. Los usuarios usualmente repiten un carácter en una palabra para enfatizar y exagerar al momento de describir algo. Solamente se exceptúan los casos "ll" y "rr", que son caracteres consecutivos válidos en el idioma español.
– Normalizar URL's: una de las características que más se encuentran en las preguntas del conjunto de datos son las direcciones web, ya que los usuarios utilizan estos recursos para realizar una pregunta aún más específica. Una URL se considera una característica y se representa mediante la palabra "url".

2. Eliminación de palabras vacías. Esta tarea permite eliminar palabras que aparecen con mucha frecuencia en las preguntas pero no son relevantes. En muchos idiomas, como el español, existe un conjunto de las palabras más comunes, como por ejemplo, los determinantes o las conjunciones (p.ej., el o y).

Este proceso permite reducir las palabras y mejorar el rendimiento del procesamiento del conjunto de datos. Para esta tarea se utilizo una lista de palabras vacías en español incluidas en el paquete de NLTK de Python[5].

3. Stemming. El objetivo principal de esta técnica es eliminar cualquier sufijo y prefijo de las palabras con el fin de obtener la palabra raíz. El "stem" se obtiene después de aplicar un conjunto de reglas sin preocuparse por la parte del discurso (POS) o el contexto de ocurrencia de la palabra [6].

En este estudio se utilizó el Snowball Stemmer para el idioma español implementado en el paquete de NLTK de Python. El texto preprocesado se guarda en un archivo JSON (Java Script Object Notation) que se utiliza para la representación de texto de las preguntas.

3.3. Representación de texto

En la etapa de representación de texto se transforma el texto preprocesado de la pregunta a una representación matemáticamente computable, esta representación en la mayoría de los casos es un vector ponderado. Para llevar a cabo la representación de las preguntas para su posterior clasificación con una red CNN se utilizó el modelo Word2vec.

Word2vec es un método que permite obtener word embeddings para una mejor representación de las palabras. Su principal objetivo es transformar un espacio de características de gran dimensionalidad a vectores de características de baja dimensión al preservar la similitud contextual del conjunto de datos. A continuación se describen los pasos para la construcción de un modelo de word embeddings utilizando las arquitecturas de Word2vec.

– Construcción de Modelo Word2vec. Como primer paso se debe construir un modelo Word2vec, para esto se utilizó la biblioteca Gensim[6] de Python, ya que provee de una clase para trabajar con modelos Word2vec permitiendo utilizar las dos arquitecturas de Word2vec: Continuous Bag Of Words (CBOW) y Skip-gram. Para que Word2vec genere los vectores se necesita que se proporcione como entrada los textos de las preguntas.

[5] pypi.org/project/nltk/
[6] pypi.org/project/gensim/

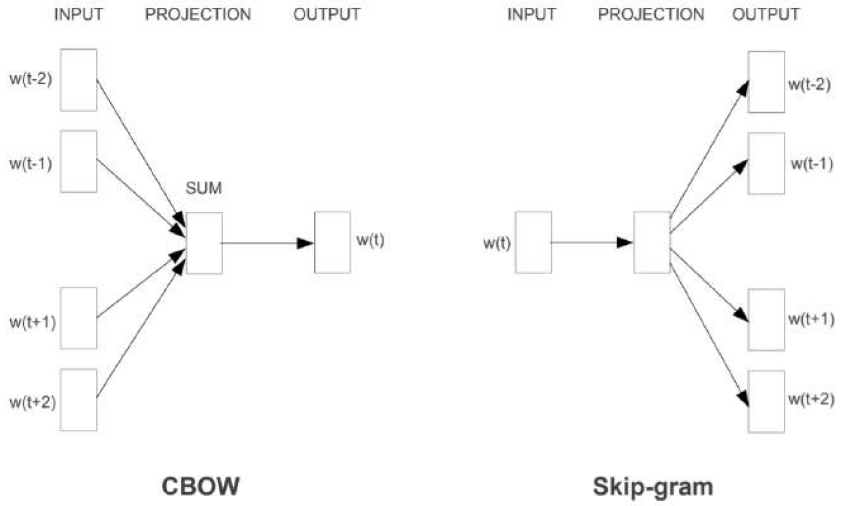


Fig. 2. Arquitecturas de Word2vec: CBOW predice la palabra actual basado en el contexto. Skip-gram predice las palabras contexto dada una palabra (basada en Mikolov et.al [12]).

– Representación de las preguntas. Para representar las preguntas se realiza la transformación del texto a una secuencia de números, utilizando la clase Tokenizer de Keras[7], mediante la implementación de un método se vectoriza el conjunto de preguntas, convirtiendo cada uno de los textos en una secuencia de enteros. Un punto importante es que los vectores de las preguntas codificadas se deben normalizar a una longitud máxima, esta con respecto a la longitud máxima de los textos que conforman el conjunto de entrenamiento y se rellenan con ceros las dimensiones sobrantes.
– Construcción de la matriz de embeddings. La matriz de embeddings tiene el objetivo de alimentar la capa de embedding de una red neuronal de aprendizaje profundo. Esta capa de la red neuronal se utiliza principalmente en aplicaciones relacionadas con el procesamiento del lenguaje natural, como el modelado del lenguaje.

  Una matriz de word embeddings es una lista de todas las palabras del vocabulario y sus respectivos embeddings o pesos. La dimensión que se definió para la construcción del modelo Word2vec para este estudio fue de 300 dimensiones.

3.4. Configuración del modelo word2vec

Para la implementación del modelo de Word2vec se utilizan las dos arquitecturas propuestas por el modelo, las cuáles utilizan una red neuronal de tres capas (1 capa de entrada, 1 capa oculta, 1 capa de salida): CBOW y Skip-gram, así como se muestra en la Figura 2. Para que Word2vec genere los vectores se necesita que se proporcione como entrada los textos de las preguntas, a continuación se especifican los parámetros que se configuraron para construir dicho modelo.

[7] keras.io/

– Sentences: son los datos de entrada, en este caso son los textos de las preguntas. Como entrada se tiene un total de 2,000 preguntas, esto debido a que se pretende que el modelo se entrene con un mayor número de textos para que modele un vocabulario más amplio.
– Workers: 2. Número de hilos que se corren en paralelo.
– Size: 300. Normalmente, el tamaño que se utiliza es de 300 dimensiones, ya que se obtienen vectores más densos y más ricos con respecto a la relevancia de cada palabra en cada dimensión del vector [14].
– Window: 2. Es el número de palabras que se encuentran antes o después de cada palabra objetivo (word target) con el fin de tomarlas en cuenta y entrenar el modelo.
– Min_count: 1. Es el mínimo de ocurrencias de una palabra para ser tomada en cuenta, ya que se define con un valor de 1, todas las palabras se incluyen en el modelo.
– Sg: 0 y 1. Es la arquitectura a utilizar; 1:Skip-gram y 0: CBOW. Se configuran ambas opciones con el fin de realizar experimentos con las dos arquitecturas de Word2vec.
– Iter: 40. Número de iteraciones (épocas) de entrenamiento sobre el conjunto de datos.

3.5. Configuración de red neuronal convolucional (CNN)

Las Redes Neuronales Convolucionales (en inglés, Convolutional Neural Networks) son arquitecturas de redes neuronales que cuentan con diversas capas ocultas, lo que provoca que cuente con altos niveles de profundidad. Típicamente, en una CNN, W_j es una convolución y p es un rectificador $\max(x, 0)$ o una función sigmoide.

En resumen, se puede ver a W_j como una pila de filtros convolucionales. De este modo, las capas son mapas de filtro y cada capa se puede formalizar como una suma de convoluciones de la capa anterior, tal y como se define en la Ecuación 1 [8]:

$$x_j(u, k_j) = p(\sum_k (x_{j-1}(., k) \cdot W_j, k_j(., k))(u)). \quad (1)$$

Para la clasificación de preguntas se utiliza una arquitectura basada en la propuesta por Kim [7]. Esta consta de una capa embedding (representación de palabras por vector), una capa convolucional, una capa max pooling, una capa dropout y una capa de salida fully-connected. En la Figura 3 se muestra la estructura general de un modelo de clasificación de preguntas con una red neuronal convolucional.

Para este estudio se implementó una red CNN para la clasificación de preguntas en español de un dominio de comercio electrónico. A continuación se describen las características del modelo:

– Capa Embedding: La dimensión del vector de entrada se define a la máxima longitud detectada en los textos de las preguntas N . Por lo que, el vector resultante se define con una dimensión de (55,300), dicho vector es la entrada a la red convolucional. La matriz de word embedding que se obtuvo mediante Word2vec se carga en esta capa con el fin de realizar un tipo de aprendizaje de transferencia a través del modelo pre-entrenado de word embeddings.

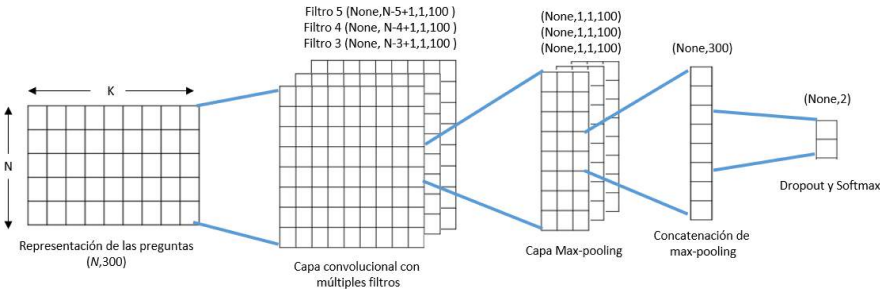


Fig. 3. Arquitectura de una red neuronal convolucional para la clasificación de preguntas. Basada en Kim [7].

– Capa Convolucional: Se utilizan tres filtros diferentes con tamaños de kernel k = [3, 4, 5] de un ancho de 100. Además, la función ReLU (Rectified Linear Unit) es utilizada como una función de activación no lineal para calcular el mapa de características de la capa de convolución.
– Capa Pooling: Para este problema se opta por la función max-pooling, debido a que es ampliamente adoptada para este tipo de tareas. Posteriormente, los valores máximos de todos los mapas que se hayan creado se concatenan para dar como resultado el vector de la pregunta.
– Capa Dropout: Esta capa mantiene activa solo una neurona durante el entrenamiento, para esto se debe determinar una probabilidad p=0.5.
– Capa Softmax: La capa softmax fully-connected nos permite transformar los valores de salida de la clase positiva y negativa en probabilidades normalizadas utilizando la función softmax definida por la Ecuación 2:

$$\hat{y}_i = \frac{e^{z_i}}{\sum_{t=0}^{1} e^{z_t}} \text{ para: } i = 0, 1. \tag{2}$$

4. Experimentos

Los experimentos tienen el objetivo de llevar a cabo el entrenamiento y evaluación de seis configuraciones de redes CNN. De acuerdo al escenario de evaluación definido se pueden obtener las métricas de evaluación para posteriormente realizar un análisis de los resultados obtenidos por los modelos de clasificación CNN.

4.1. Entrenamiento del modelo

En esta etapa se entrena el modelo propuesto y se utiliza el escenario de evaluación definido. En la Tabla 2 se muestran las seis configuraciones implementadas para este estudio.

– Implementación: La red neuronal CNN se implementó mediante los módulos de Keras[8] y Tensorflow[9] para Python 3.7.

[8] keras.io
[9] www.tensorflow.org/?hl=es-419

Tabla 2. Configuraciones de redes neuronales CNN para la clasificación de preguntas.

Conf.	Tamaño filtros	No. filtros	Optimizador	Factor de aprendizaje	Épocas	Tamaño Batch
CNN1-CB	3,4,5	300	adadelta	0.001	30	64
CNN2-CB	3,4,5	300	adadelta	0.001	25	50
CNN3-CB	3,4,5	300	adadelta	0.001	30	50
CNN4-SG	3,4,5	300	adadelta	0.001	30	64
CNN5-SG	3,4,5	300	adadelta	0.001	25	50
CNN6-SG	3,4,5	300	adadelta	0.001	30	50

– Configuración del modelo: Se configuraron diferentes modelos CNN con el fin de compararlos y seleccionar el modelo óptimo para la clasificación de preguntas en español. Se utilizaron las dos arquitecturas de Word2vec (CBOW y skip-gram) previamente entrenadas. Lo anterior, con el fin de conocer que arquitectura se adapta mejor a la tarea de clasificación de preguntas.

4.2. Métricas

Los modelos de clasificación de preguntas obtenidos se evalúan utilizando las siguientes métricas: precisión, cobertura, medida F y exactitud. Estas métricas son calculadas para cada una de las clases mediante las siguientes formulas:

$$\text{exactitud} = \frac{\text{total TP} + \text{total TN}}{\text{total ejemplos}}, \tag{3}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \tag{4}$$

$$\text{cobertura} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{5}$$

$$\text{medidaF} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}, \tag{6}$$

donde, TP denota el número de positivos reales, TN, los positivos negativos; FP, número de falsos positivos y FN denota los falsos negativos.

4.3. Predicción del modelo

En esta etapa se utiliza la técnica de validación cruzada de "k=10" iteraciones. Por lo que, el conjunto de datos de las preguntas se divide en 10 grupos o subconjuntos. El primer grupo se selecciona para pruebas y los grupos restantes para el entrenamiento.

En cada iteración, se utiliza el 80 % de los datos para entrenamiento y el 20 % para pruebas. Los resultados de exactitud, precisión, cobertura y medida F de cada iteración fueron promediados.

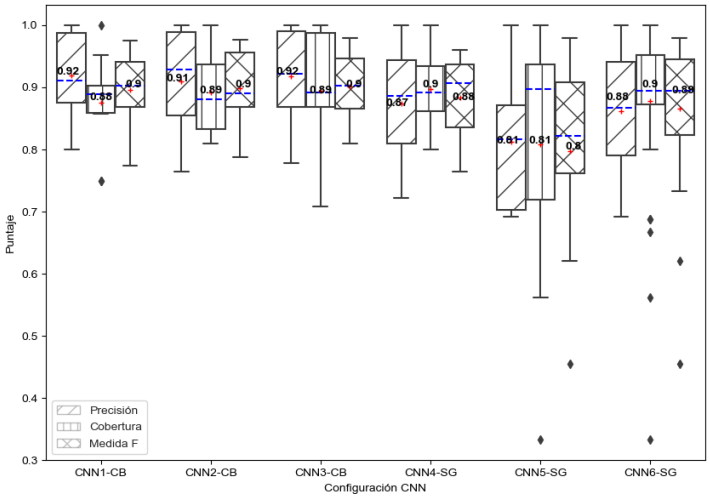


Fig. 4. Resultados de las métricas de Precisión, Cobertura y Medida F de la validación cruzada 10 iteraciones de las configuraciones de la red CNN.

5. Resultados y discusión

En la Figura 4 se presenta un diagrama de cajas con los resultados obtenidos en la evaluación usando validación cruzada de 10 iteraciones, por cada una de las configuraciones de las redes CNN. Los modelos CNN1-CB y CNN3-CB obtuvieron resultados muy similares. El modelo CNN1-CB obtuvo un 92 % de precisión, 88 % de cobertura y un 90 % de medida F. Por otro lado, el modelo CNN3-CB obtuvo un 92 % de precisión, 89 % de cobertura y 90 % de medida F.

Estos dos modelos cuentan con una alta precisión y una alta cobertura, llegando a tener más de un 80 % en ambas métricas. En la Tabla 3 se muestran los resultados de exactitud y error obtenidos durante el entrenamiento por los seis modelos de clasificación. Con los resultados obtenidos se observa que los modelos CNN1-CB, CNN2-CB y CNN3-CB muestran un desempeño similar. El modelo CNN3-CB obtuvo un 91 % de exactitud y un error del 0.339, los modelos CNN1-CB y CNN2-C obtuvieron un 90 % de exactitud con un error del 0.320 y 0.325 respectivamente.

Dado los resultados obtenidos con respecto a las métricas de exactitud, precisión, cobertura y medida F, se observó que los modelos que obtuvieron el mejor desempeño en la etapa de entrenamiento implementan un modelo Word2vec con arquitectura CBOW. Cabe mencionar que para que los modelos de clasificación basados en redes profundas puedan obtener mejores resultados se necesita de conjuntos de datos más grandes para construir un modelo Word2vec con una gran cantidad de palabras y por ende poder utilizar el modelo preentrenado en el entrenamiento de una red neuronal convolucional, con el fin de obtener un modelo más robusto.

Con los experimentos realizados se puede observar que para construir un modelo de clasificación que obtenga un alto nivel de precisión en la clasificación de preguntas en español se puede hacer uso de un enfoque de aprendizaje profundo y la arquitectura CBOW de Word2vec para obtener resultados por encima de un 80 % de precisión.

Tabla 3. Resultados de exactitud y error de los modelos CNN entrenados para la clasificación de preguntas.

Conf.	Exactitud	Error
CNN1-CB	0.90	0.320
CNN2-CB	0.90	0.325
CNN3-CB	0.91	0.339
CNN4-SG	0.89	0.374
CNN5-SG	0.82	0.484
CNN6-SG	0.90	0.371

6. Conclusiones

En este trabajo se propone un enfoque de aprendizaje profundo para el desarrollo de un modelo de clasificación de textos cortos en el idioma español. Dicho modelo permite representar y clasificar las preguntas de un dominio de comercio electrónico a una de las dos posibles clases: operativa o técnica.

La estrategia utilizada consiste en contar con un conjunto de datos etiquetado manualmente para posteriormente realizar un preprocesamiento y representación de los textos. Finalmente, se lleva a cabo el entrenamiento y evaluación de los modelos de clasificación basados en una Red Neuronal Convolucional.

Los modelos de aprendizaje profundo, como Word2Vec se utilizan para obtener mejores representaciones vectoriales de palabras y mejorar la precisión de los clasificadores entrenados con algoritmos tradicionales de aprendizaje automático.

Con base en los experimentos realizados se puede observar que para aplicar un enfoque de aprendizaje profundo que permita obtener modelos de clasificación por encima del 90 % de exactitud se debe utilizar un conjunto de datos mucho más grande, que contenga millones de preguntas. Lo anterior, con el fin de entrenar un modelo de word embeddings más robusto, que sea capaz de representar un mayor número de palabras referentes al dominio de estudio.

Como trabajo futuro se pretende identificar y etiquetar el conjunto de datos con más clases con el fin de desarrollar un modelo de clasificación de preguntas multiclase. Además, implementar otras arquitecturas de redes neuronales de aprendizaje profundo como las Redes Neuronales Recurrentes (RNN) y las Redes Long Short Term Memory (LSTM), con el fin de comparar su desempeño con respecto a las redes CNN.

Referencias

1. Blacksip México: Reporte de Industria: El e–commerce en México 2020. Blacksip México (2020)
2. Forbes: Comercio electrónico en México repuntó 23 % en 2022, revela estudio de la AMVO (2023) www.forbes.com.mx/comercio-electronico-en-mexico-repunto-23-en-2022-revela-estudio-de-la-amvo/
3. Gaffar-Khan, A.: Electronic commerce: A study on benefits and challenges in an emerging economy. Global Journal of Management and Business Research: B Economics and Commerce, vol. 16, no. 1, pp. 19–22 (2016)

4. Gupta, S., Borkar, D., Mello, C. D., Pati, S.: An E-Commerce website based chatbot. International Journal of Computer Science and Information Technologies, vol. 5, pp. 1483–1485 (2015)
5. Haddi, E., Liu, X., Shi, Y.: The role of text pre-processing in sentiment analysis. Procedia Computer Science, vol. 17, pp. 26–32 (2013) doi: 10.1016/j.procs.2013.05.005
6. Jivani, A. G.: A comparative study of stemming algorithms. International Journal of Computer Applications in Technology, vol. 2, no. 6 (2011)
7. Kim, Y.: Convolutional neural networks for sentence classification (2014) doi: 10.48550/ARXIV.1408.5882
8. Koushik, J.: Understanding convolutional neural networks (2016) doi: 10.48550/ARXIV.1605.09081
9. Kulkarni, A., Mehta, K., Garg, S., Bansal, V., Rasiwasia, N., Sengamedu, S.: ProductQnA: Answering user questions on E-commerce product pages. pp. 354–360 (2014) doi: 10.48550/ARXIV.1408.3829.
10. Kumar, S., Zymbler, M.: A machine learning approach to analyze customer satisfaction from airline tweets. Journal of Big Data, vol. 6, no. 1 (2019) doi: 10.1186/s40537-019-0224-1
11. Li, Y., Miao, Q., Geng, J., Alt, C., Schwarzenberg, R., Hennig, L., Hu, C., Xu, F.: Question answering for technical customer support. Natural Language Processing and Chinese Computing, Springer International Publishing, vol. 11108, pp. 3–15 (2018) doi: 10.1007/978-3-319-99495-6_1
12. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space (2013) doi: 10.48550/ARXIV.1301.3781
13. MonkeyLearn: Short Text Classification. MonkeyLearn (2020) monkeylearn.com/short-text-classification/
14. Parodi, G., Cantos-Gómez, P., Howe, C.: Lingüística de corpus en español. Routledge (2022)
15. Qasem, M., Thulasiram, R., Thulasiram, P.: Twitter sentiment classification using machine learning techniques for stock markets. In: International Conference on Advances in Computing, Communications and Informatics, pp. 834-840 (2015) doi: 10.1109/icacci.2015.7275714
16. Search Logistics: Ecommerce Statistics 2023. Search Logistics (2023) www.searchlogistics.com/grow/statistics/ecommerce-statistics
17. Sharma, R., Nigam, S., Jain, R.: Opinion mining of movie reviews at document level. International Journal on Information Theory, vol. 3, no. 3 (2014) doi: 10.48550/arXiv.1408.3829
18. Singla, Z., Randhawa, S., Jain, S.: Sentiment analysis of customer product reviews using machine learning. In: International Conference on Intelligent Computing and Control (2017) doi: 10.1109/i2c2.2017.8321910
19. Song, G., Ye, Y., Du, X., Huang, X., Bie, S.: Short text classification: A survey. Journal of Multimedia, vol. 9, no. 5 (2014) doi: 10.4304/jmm.9.5.635-643
20. Xin, L., Xuan-Jing, H., Li-de, W.: Question Classification using Multiple Classifiers. In: Proceedings of the Fifth Workshop on Asian Language Resources and First Symposium on Asian Language Resources Network (2005)

ISSN 1870-4069

Análisis de las características más importantes para la detección de noticias falsas en español

Sergio Damián, Hiram Calvo, Alexander Gelbukh

Instituto Politécnico Nacional, Centro de Investigación en Computación, México

{sdamians2019, hcalvo}@cic.ipn.mx, gelbukh@gelbukh.com

Resumen. El uso de modelos de lenguaje basados en Transformers para tareas del procesamiento del lenguaje natural (PLN) es cada día más común gracias a su potencial, su generalidad y gracias a que sus resultados superan de manera notoria otro tipo de estrategias y modelos. El principal problema de éstos es su gran número de parámetros y la complejidad que esto representa para la explicabilidad del modelo durante el proceso de resolución de una tarea en específico. El presente trabajo busca la explicabilidad para el funcionamiento del modelo BETO en la tarea de detección de noticias falsas utilizando un método alternativo al uso de pesos de atención encontrados en los mecanismos de atención de los Transformers. Se observa que las categorías gramaticales como la adposición, los determinantes, los sustantivos y los nombres propios representan los tokens más importantes que el modelo analiza para obtener las estimaciones o resultados. A su vez, el trabajo expone que para el corpus analizado, el uso de mayúsculas y el ignorar el entrenamiento de la capa de embeddings del modelo, mejora la comprensión de la tarea, presentando un valor F1 de 0.8653.

Palabras clave: Noticias falsas, explicabilidad, BETO, transformers.

Feature Importance Analysis for Fake News Detection in Spanish

Abstract. Using Transformer-based language models to solve Natural Language Processing (NLP) tasks is more common due to their potential, generalization, and the results that can surpass any other types of strategies and models. The main issue when using them is their huge amount of parameters and the complexity when trying to provide explainability of the model behavior. The present work analyses an alternative method to search explainability of BETO model behavior for a Fake News Detection task. It is observed that some part-of-speech subsets are the most relevant features in this case of study, like adposition, determinant, noun and proper noun labels. Besides, this work demonstrates uppercased tokens are relevant for this specific corpus and also freezing embedding layers can improve the metric scores, getting a F1 Score of 0.8653.

Keywords: Fake news, explainability, BETO, transformers.

1. Introducción

Los modelos de lenguaje basados en la arquitectura de Transformers han surgido como el actual estado del arte en múltiples tareas de procesamiento de lenguaje natural (PLN) desde su presentación en [9]. El primer modelo de lenguaje desarrollado con esta arquitectura fue BERT (Representación de Codificador Bidireccional de Transformadores por sus siglas en inglés), el cual fue lanzado un año después de la presentación de la arquitectura Transformer y cambió la forma en la que eran abordadas las tareas de PLN [3].

La gran mayoría de los mejores resultados en las tareas de PLN eran obtenidos mediante modelos basados en BERT. Después se elaboraron múltiples variantes del modelo, considerando diferentes arquitecturas, conjuntos de datos, procesos de entrenamiento e idiomas.

Una de las variantes fue BETO presentado por [1], cuya diferencia con BERT consiste en que fue entrenada para el idioma español. La arquitectura de BERT y BETO toma en cuenta solamente la parte del codificador de la arquitectura de Transformers para obtener representaciones de los textos (también llamadas embeddigs) de las secuencias de texto utilizadas como entradas para los modelos.

BETO y BERT en su versión base cuentan con una arquitectura de doce capas, donde a su vez, cada una de ellas cuenta con doce mecanismos de atención o heads, siendo estos la parte central de la arquitectura de los Transformers. Los embeddings o representaciones de los tokens de entrada son actualizados en cada una de las capas a partir de los parámetros del mecanismo de atención, proporcionando doce diferentes representaciones adicionales.

La figura 1 representa cómo funciona el mecanismo de atención. Consiste en una serie de operaciones de producto punto entre vectores, además de la construcción de una matriz de pesos de atención, cuya principal tarea es asignar un valor de peso o importancia para cada uno de los tokens (palabras o signos en los que es dividido el texto) de entrada con respecto a los demás. Es a partir de este mecanismo donde cada token obtiene en su representación un cierto contexto con respecto a los demás.

Uno de los principales retos de utilizar BERT o cualquiera de sus variantes es explicar lo que el modelo es capaz de aprender, tanto del idioma como de la tarea que se está resolviendo. Una forma de explicabilidad consiste en encontrar las propiedades más importantes que el modelo considera dentro de su arquitectura para su aprendizaje en la tarea a resolver.

Este trabajo presenta un análisis de la arquitectura de BETO y cuáles son las características o tokens más importantes para un caso de estudio en particular: la detección de noticias falsas en español, como un primer paso para exponer una posible interpretación de las estimaciones del modelo.

El presente trabajo se estructura de las siguientes secciones: La sección 2 presenta los trabajos relacionados, explicando cómo algunos autores estudian el comportamiento de modelos de lenguaje basados en BERT. La sección 3 describe el conjunto de datos utilizado en este trabajo y la descripción de la arquitectura de los experimentos. La sección 4 presenta los resultados obtenidos y un análisis de éstos.

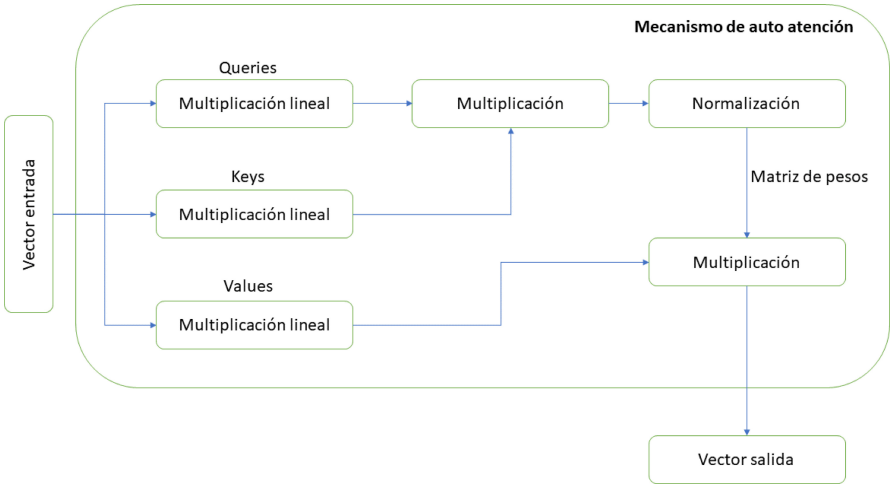


Fig. 1. Diagrama básico del mecanismo de auto atención que utiliza la arquitectura de Transformer. Se observa el punto donde se obtienen los pesos de atención, los cuales son generalmente utilizados para la explicabilidad del modelo.

Y finalmente la sección 5 concluye el presente trabajo, provee algunas conclusiones para el análisis de los modelos de lenguaje para la tarea de la detección de las noticias falsas y describe un posible trabajo a futuro.

2. Trabajo relacionado

Los valores de los pesos de atención encontrados en los mecanismos de atención son comúnmente utilizados para proporcionar la explicabilidad del modelo, ya que permiten presentar la relación que tienen los diferentes tokens respecto a los otros. [2] estudiaron el comportamiento de cada cabeza de atención por capa en BERT y observaron que existen múltiples características lingüísticas intrínsecas en los pesos de atención del modelo. Utilizando tareas de clasificación para diferentes tipos de sintaxis en el idioma inglés, demostraron que tokens especiales del modelo como [CLS] y [SEP] juegan un papel importante en el rol de detectar características sintácticas del lenguaje.

Sin embargo, trabajos posteriores han estudiado que no es suficiente utilizar únicamente los pesos de atención como forma de explicar el comportamiento de este tipo de modelos. [4] compararon las características más importantes encontradas por los pesos de atención contra las características encontradas mediante otros métodos de explicabilidad más tradicionales, como el uso de métodos de selección de características usando valores Chi^2, ganancia de información o information gain, entre otros.

Las características encontradas por los pesos de atención difieren de las encontradas por otros métodos de selección, lo que sugiere que los pesos de atención no proveen la suficiente información como para denotar realmente las características más importantes del modelo. [5] también demostraron que existe una correlación muy débil entre los pesos de atención y las medidas de importancia de características como aquellas basadas en el gradiente descendente.

Tabla 1. Información del corpus de noticias falsas y veridícas a utilizar.

Conjunto de datos	Noticias falsas	Noticias verídicas	Total
Entrenamiento	1036	1444	2480
Validación	133	185	318
Pruebas	311	434	745
Total	1480	2063	3543

Por otro lado, en [10] se concluye que encontrar las características más importantes por medio de los pesos de atención es solo una forma de explicar el comportamiento del modelo, y que simplemente no se debería de considerar como la única ni la mejor manera de proveer dicha información. Con esto presente, se presentaron algunos otros trabajos que proveen diferentes metodologías para explicar el comportamiento de los modelos.

El presente trabajo toma inspiración en el desarrollo presentado por [6], quienes ejecutan una nueva función a partir de los pesos de atención, la norma de los vectores values y una capa de pesos densa que permite la conexión de los valores de los doce mecanismos de atención en una capa.

Demostraron que sus resultados presentan una mejor claridad para entender el comportamiento del modelo en lugar de utilizar únicamente los pesos de atención. La ecuación 1 representa la nueva forma de representar el valor de un vector de un token de entrada, utilizando la nomenclatura de [9]:

$$f(x) := (xW^V + b^V)W^O. \tag{1}$$

En [6] se explicó que en primer lugar se calculan los pesos de atención α, posteriormente los nuevos valores de cada vector x utilizando $f(x)$, después realiza la suma de los pesos y finalmente realiza el cálculo de la norma de $\mathrm{sum}(\alpha f(x))$. La ecuación 2 representa la transformación final del valor que se utiliza en sustitución del solo uso de los pesos de atención presentada en el trabajo anteriormente mencionado:

$$\mathrm{katt}_i = \left\| \sum_{j=1}^{n} \alpha_{i,j} f(x_j) \right\|, \tag{2}$$

donde cada token x de una entrada es transformado por la función $f(x)$ presentada en la ecuación 1, posteriormente se multiplica por su correspondiente peso de atención $\alpha_{i,j}$, luego los valores de todos los tokens son sumados y finalmente representan la atención katt_i obteniendo la norma de ese vector, donde i representa la i-ésima secuencia de entrada de tokens para el modelo.

2.1. Contribuciones de este trabajo

En la actualidad no existe una metodología única o precisa para la explicabilidad de los resultados producidos por modelos de lenguaje. Existen trabajos como el de [6] o el de [4], donde se explica que los pesos de atención no han sido una métrica suficiente para presentar explicabilidad del modelo, ya que no tienden a presentar resultados

Tabla 2. Tipos de modelos utilizados en este trabajo.

Nombre	Freezing embeddings	Uso de mayúsculas
all_layers	No	No
all_layers_cased	No	Sí
all_layers_no_emb	Sí	No
all_layers_no_emb_cased	Sí	Sí

equiparables con otros métodos de explicabilidad o de selección de características más convencionales. [6] requirieron modificar el código base de BERT para que pudieran aplicar los cálculos necesarios para su experimentación. El presente trabajo presenta un análisis inspirado en el trabajo de [6]. En lugar del análisis del modelo BERT, se utiliza BETO, ya que el caso de estudio a analizar se encuentra en el idioma español.

La tarea en cuestión es la detección de noticias falsas, la cual se aborda como una tarea de clasificación binaria. La hipótesis es que mediante la relación del uso de los valores de vectores con los pesos de atención, es suficiente para encontrar una mejor interpretación del modelo que la presentada por el solo uso de pesos de atención.

3. Desarrollo de la solución

3.1. Corpus

El corpus consiste en un conjunto de documentos (textos de noticias) etiquetados de manera binaria como contenido de noticias falsas y verídicas. El corpus consiste en una combinación de los documentos recolectados por [7] y por [8]. La tabla 1 muestra información cuantificada sobre el corpus a utilizar.

3.2. Preprocesamiento del texto

La longitud máxima para cada secuencia de entrada en BETO es de 512 tokens. Significa que incluso cada símbolo de puntuación es convertido en un token y la longitud máxima puede ser alcanzada fácilmente en textos relativamente largos. [2] demostraron que símbolos de puntuación como puntos y comas pueden ser relevantes para el modelo de lenguaje BERT, por lo que se determinó no removerlos.

Además dentro del corpus, el conjunto presentado por [7] contiene enmascarados todos los caracteres numéricos con la etiqueta *NUMBER*, algo que BETO no considera de forma nativa y termina por transformar dicha etiqueta en 3 tokens diferentes. Por lo tanto, estas etiquetas fueron cambiadas por la etiqueta num para simplificar la cantidad de tokens.

3.3. Arquitectura del experimento

Se consideró un proceso de entrenamiento tipo ajuste fino o fine tuning con diez épocas, para dos tipos de modelos: BETO cased y BETO uncased. El modelo BETO cased trabaja con tokens en mayúsculas y minúsculas, mientras que el modelo BETO uncased transforma todos los caracteres a minúsculas.

Tabla 3. Resultados de los experimentos en el corpus de validación.

Modelo	Precisión	Recall	Exactitud	Valor F1
all_layers	0.8377	0.8113	0.8270	0.8243
all_layers_cased	0.8377	0.8113	0.8270	0.8243
all_layers_no_emb	0.8431	0.8113	0.8302	0.8269
all_layers_no_emb_cased	**0.8506**	**0.8239**	**0.8396**	**0.8371**

Tabla 4. Resultados de los experimentos en el corpus de pruebas.

Modelo	Precisión	Recall	Exactitud	Valor F1
all_layers	0.8592	0.8221	0.8443	0.8402
all_layers_cased	0.8595	0.8571	0.8591	0.8583
all_layers_no_emb	0.8508	0.8302	0.8430	0.8404
all_layers_no_emb_cased	**0.8736**	**0.8571**	**0.8671**	**0.8653**

También se experimentó con una variante donde se utiliza el proceso freezing o congelamiento de capas, donde evitamos la actualización de los párametros en ciertas capas de la arquitectura.

En este trabajo se presenta una variante de los modelos donde no se actualizaron los valores de los parámetros en las capas utilizadas para el ajuste de los embeddings, para analizar solamente los valores de las capas que incluyen los mecanismos de atención. La tabla 2 muestra en resumen los tipos de variantes del modelo BETO y la nomenclatura utilizada en las secciones posteriores del trabajo.

3.4. Explicabilidad de los resultados

Para la observación y explicabilidad de los modelos entrenados, se determina la estrategia de encontrar las características o tokens más importantes que el modelo considera en cada una de sus doce capas.

Para la búsqueda de las características más importantes, se obtienen los valores de los pesos de anteción y los vectores values. La ecuación 3 representa el cálculo para la obtención los pesos de atención de acuerdo con [9], lo cual se mantiene en este trabajo:

$$\alpha = \frac{QK^T}{\sqrt{\dim_k}}, \tag{3}$$

donde las matrices Q y K representan los módulos de la multiplicación lineal de Queries y Keys presentadas en la figura 1 y $\dim_k$ es la dimensión o cantidad de tokens k. Entonces se obtienen los vectores values para cada token x y son multiplicados por la matriz de pesos de atención calculada en la ecuación anterior, como se observó en [6]. En este trabajo, se experimenta simplificando la ecuación 1 por la ecuación 4, donde la matriz W^O era la principal causante de recodificar la arquitectura del modelo del lenguaje:

$$g(x) := xW^V + b^V, \tag{4}$$

donde x representa un token de entrada, W^V y b^V son las matrices de pesos y bias de la multiplicación lineal en el módulo values del mecanismo de atención respectivamente.

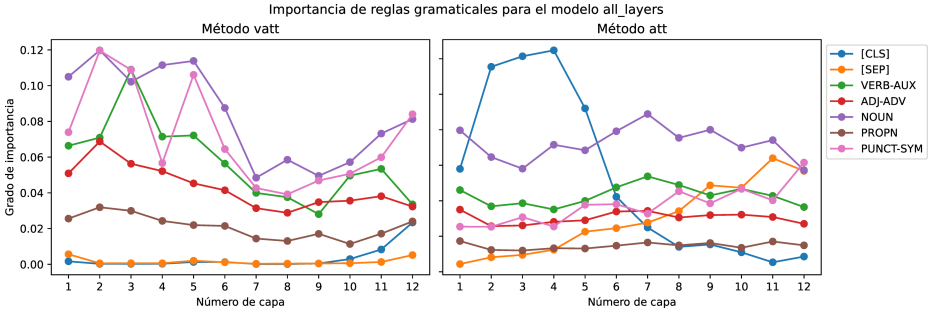


Fig. 2. Comparación de la importancia de categorías gramaticales en los métodos **vatt** y att para el modelo all_layers.

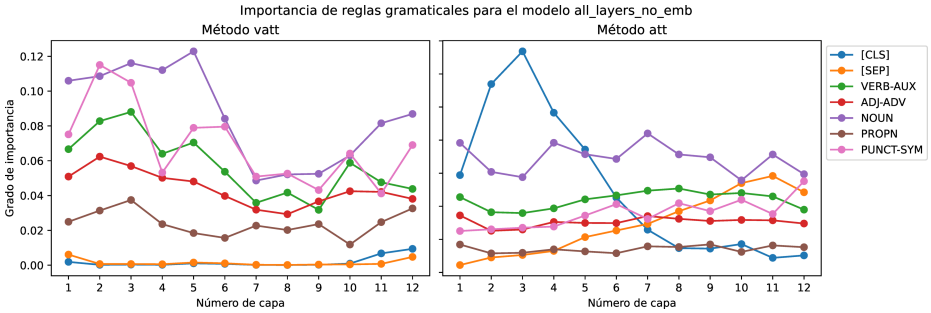


Fig. 3. Comparación de la importancia de categorías gramaticales en los métodos **vatt** y att para el modelo all_layers_no_emb.

Cada capa del modelo tiene doce diferentes mecanismos de atención o heads, cuyos valores son promediados y finalmente son multiplicados por los vectores de $g(x)$, de tal forma que la ecuación 5 presenta la forma en la que este trabajo analiza la explicabilidad del modelo, donde $\alpha'_{i,j}$ representa el valor promedio del peso de atención:

$$\text{vatt}_i = \left\| \sum_{j=1}^{n} \alpha'_{i,j} g(x_j) \right\|. \tag{5}$$

Durante las siguientes secciones, al método convencional de interpretabilidad que hace uso de los pesos de atención, se le nombra como **método att** y al método propuesto se le nombra como **método attv** para su comparación. La interpretabilidad es presentada mediante subconjuntos de los tokens organizados por categorías gramaticales, extrayéndolas con la librería SpaCy.

4. Experimentos y resultados

La tabla 3 y la tabla 4 muestran los resultados obtenidos para el conjunto de validación y pruebas de los experimentos para cada tipo de variante del modelo BETO.

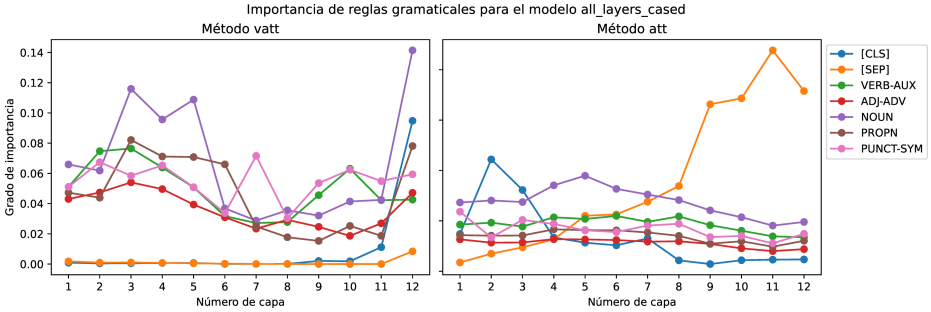


Fig. 4. Comparación de la importancia de categorías gramaticales en los métodos **vatt** y att para el modelo all_layers_cased.

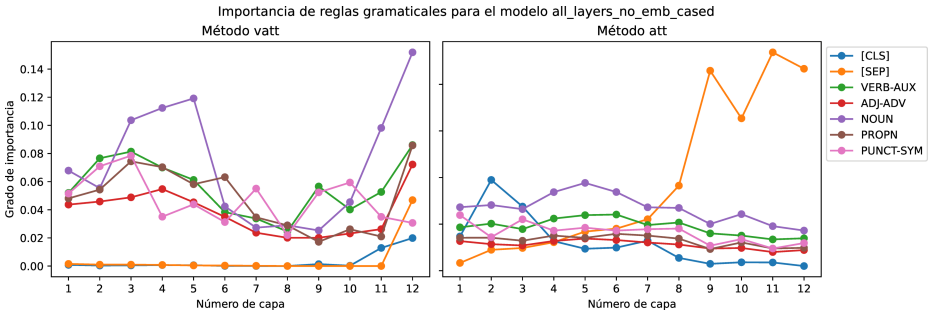


Fig. 5. Comparación de la importancia de categorías gramaticales en los métodos **vatt** y att para el modelo all_layers_no_emb_cased.

El modelo all_layers_no_emb_cased es el que obtuvo los mejores resultados, lo que significa que el uso de mayúsculas y el mantener los valores preentrenados de los embeddings mejora el proceso de detección de noticias falsas para el corpus analizado en este trabajo. Las figuras 2, 3, 4 y 5 muestran una comparación de categorías gramaticales y su importancia en cada una de las doce capas de la arquitectura del modelo BETO. Se presenta el promedio de los resultados obtenidos para cien entradas del conjunto de pruebas.

En adición a las categorías gramaticales se consideran de manera individual los tokens especiales [CLS] y [SEP], ya que para el método **att** estos suelen obtener importancia significativa en ciertas capas del modelo, incluso superando el valor de importancia de las categorías gramaticales.

Sin embargo, se puede observar que los tokens especiales carecen de importancia en el método **vatt**, proporcionando una interpretación del comportamiento del modelo completamente diferente, pero a su vez, conserva coherencia, ya que categorías como sustantivos (NOUN), verbos (VERB-AUX) y signos de puntuación (PUNCT-SYM) presentar un alto nivel de importancia para el modelo.

En particular, la figura 5 muestra el análisis del mejor modelo presentado en este trabajo, donde se aprecia que la categoría gramatical de sustantivos fue de suma importancia para las capas 3-6 y las capas 11-12.

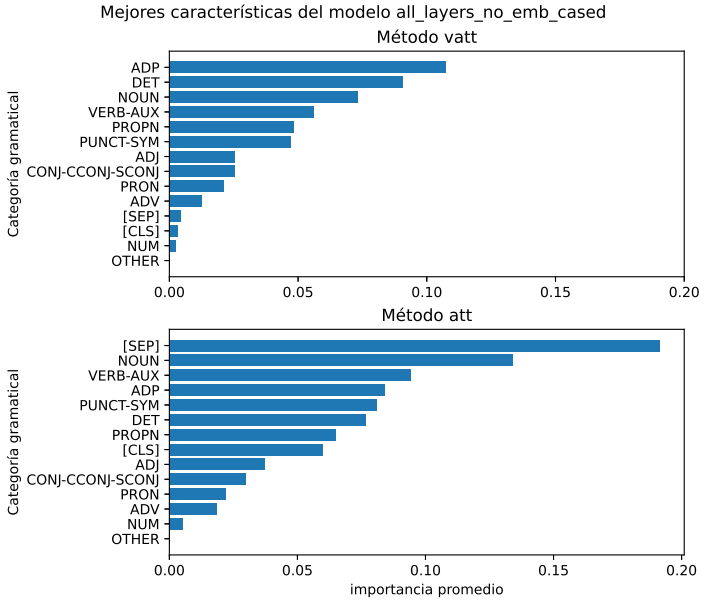


Fig. 6. Comparación de la importancia de todas las categorías gramaticales presentes en el modelo all_layers_no_emb_cased para los métodos **vatt** y att.

Por el contrario, en la figura 2 se observa que no hubo una diferencia tan clara para la categoría gramatical de sustantivos, por lo que puede considerarse como una forma de explicar el por qué ese modelo no obtuvo los mejores resultados. Una vez observada la diferencia entre ambos modelos, lo siguiente es encontrar explicabilidad con el método **vatt** para el mejor modelo presentado en este trabajo. La figura 6 presenta un análisis general de las categorías gramaticales presentes en el modelo all_layers_no_emb_cased, representando el promedio de todas las capas. Para el método **vatt** se concluye que las categorías adposición (ADP) y determinante (DET) son las más importantes en promedio para el modelo.

Ambas categorías cumplen el objetivo de contribuir a la semántica de la oración, por lo que se intuye que el modelo considera que encontrar el contexto de los textos es la parte más importante para la solución a esta tarea. No obstante, para el método **att**, el token especial [SEP] tiene la mayor relevancia, por lo que se puede generar otro tipo de explicabilidad completamente diferente. En este caso, se puede considerar que [SEP] mantiene un contexto de la entrada importante, por lo que el modelo determina que debe tener un peso de atención alto en la mayor parte de la arquitectura.

5. Conclusiones y trabajo a futuro

El presente trabajo presenta un análisis de cómo identificar de manera general las características o tokens más relevantes para la tarea de detección de noticias falsas en textos en español cuando se utilizan modelos de lenguaje basados en Transformers (específicamente en el modelo BETO).

La mejor variante del modelo BETO que se analizó fue aquella que maneja tokens en mayúsculas y minúsculas y no considera actualizar los pesos de los embeddings previamente entrenados. El valor F1 encontrado es de 0.8653 para la clase de noticias falsas. El análisis de un subconjunto de los resultados finales mostró que categorías gramaticales como los sustantivos (NOUN), la adposición o partículas gramaticales (ADP) y determinantes (DET) son las características más importantes para el modelo BETO en la tarea de detección.

El método utilizado para encontrar las características está basado en el trabajo de [6], simplificando en parte la función que se presenta, pero manteniendo resultados en donde los tokens especiales [CLS] y [SEP] presentan importancia insignificante en comparación con tokens más específicos para la tarea.

Como trabajo a futuro, se pretende analizar otros tipos de modelos basados en Transformers, con la finalidad de observar el comportamiento de la metodología propuesta. Además, es posible el análisis de otros tipos de tareas de PLN además de la clasificación, como la generación de textos, entre otras. También, es de interés el experimentar con diferentes idiomas además del español, ya que las categorías gramaticales pueden ser variadas y los modelos puedan atender a diferentes características para una misma tarea.

Referencias

1. Cañete, J., Chaperon, G., Fuentes, R., Ho, J. H., Kang, H., Pérez, J.: Spanish pre-trained BERT model and evaluation data. In: Practical ML for Developing Countries Workshop and International Conference on Learning Representations, pp. 1–10 (2020)
2. Clark, K., Khandelwal, U., Levy, O., Manning, C. D.: What does BERT look at? An analysis of BERT's attention (2019) doi: 10.48550/ARXIV.1906.04341
3. Devlin, J., Chang, M. W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding (2018) doi: 10.48550/ARXIV.1810.04805
4. Garcia-Silva, A., Gomez-Perez, J. M.: Classifying scientific publications with BERT - is self-attention a feature selection method? In: Lecture Notes in Computer Science, pp. 161–175 (2021) doi: 10.1007/978-3-030-72113-8_11
5. Jain, S., Wallace, B. C.: Attention is not explanation (2019) doi: 10.48550/arxiv.1902.10186
6. Kobayashi, G., Kuribayashi, T., Yokoi, S., Inui, K.: Attention is not only a weight: Analyzing transformers with vector norms. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (2020) doi: 10.18653/v1/2020.emnlp-main.574
7. Posadas-Durán, J. P., Gómez-Adorno, H., Sidorov, G., Moreno-Escobar, J. J.: Detection of fake news in a new corpus for the Spanish language. Journal of Intelligent and Fuzzy Systems, vol. 36, no. 5, pp. 4869–4876 (2019) doi: 10.3233/jifs-179034
8. Tretiakov, A.: Noticias falsas en español (2020)
9. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems, vol. 30 (2017)
10. Wiegreffe, S., Pinter, Y.: Attention is not not explanation (2019) doi: 10.48550/arXiv.1908.04626

ISSN 1870-4069

Agente de navegación web para detección de noticias falsas usando aprendizaje profundo por refuerzo y listas de argumentos

Alexis Fernando Aguilera Valderrama[1], Iván Vladimir Meza Ruiz[2]

[1] Universidad Nacional Autónoma de México,
Facultad de Ingeniería,
México

[2] Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas,
Departamento de Ciencias de la Computación, Ciudad de México,
México

alexisav2000@comunidad.unam.mx,
ivanvladimir@turing.iimas.unam.mx

Resumen. El crecimiento del número de noticias falsas en esta nueva era de la información ha sido tal que está causando daños a la sociedad en diferentes ámbitos, por ello es necesario el desarrollo de herramientas que permitan detectar este tipo de noticias con el fin de contrarrestar los efectos de la desinformación. Este trabajo se desarrolla un agente de Inteligencia Artificial impulsado por Aprendizaje por Refuerzo Profundo que es capaz de navegar y analizar páginas de internet relacionadas a una noticia para determinar si esta es verdadera o falsa. Para lograr estos se propone un método de fact-checking que consiste en dos listas de argumentos, con las cuales el agente puede guardar respectivamente fragmentos de texto que demuestren que la noticia es falsa o verdadera.

Palabras clave: Aprendizaje por refuerzo profundo, noticias falsas, procesamiento de lenguaje natural, fact-checking.

Web Navigation Agent for Detecting Fake News Using Deep Reinforcement Learning and Argument Lists

Abstract. The growth in the number of fake news in this new information age has been such that it is causing damage to society in different areas, which is why it is necessary to develop tools to detect this type of news to appease the effects of misinformation. In this work we develop an Artificial Intelligence agent powered by Deep Reinforcement Learning that is capable of browsing and analyzing internet pages related to a piece of news to determine if the information presented is true or false. This proposed fact-checking method relies on two lists of arguments, with which the agent can respectively list text fragments that support if the news is true or false.

Keywords: Deep reinforcement learning, fake news, natural language processing, fact-checking.

1. Introducción

Con el crecimiento acelerado de las tecnologías de la información, la proliferación y alcance de las noticias falsas han sido tan amplificada a tal grado que han causado malestares en la sociedad en diferentes sectores, tales como la salud, economía, cultura y política principalmente [11, 6, 19]; inclusive, se ha estudiado que en sociedades donde la información fraudulenta no tiene tanto alcance, sufren en menor medida las consecuencias de la misma [17, 13].

Por la gravedad que con lleva esto, se han investigado los formatos con los que pueden aparecer noticias falsas en internet, la estructura que tienen, forma de proliferación, la psicología que llevan para convencer a personas y otros factores que las pueden hacer identificables [18, 14, 16].

Una de las primeras propuestas que se consolidó para combatir la creciente desinformación es el fact-checking. Este es un proceso que tiene sus orígenes en el área periodístico, el cual consiste en verificar la autenticidad de una noticia comparando los datos que brinda, con hechos que ya son previamente conocidos [19].

Además, se puede llevar acabo si se hace un análisis que englobe el estilo de escritura de la información, el contexto en el que se encuentra, imágenes o vídeos que acompañan a la noticia, comentarios de usuarios y más características que den información sobre la noticia [15, 20].

Con el desarrollo de los algoritmos de inteligencia artificial, fue intrínseco que se propusiera la automatización del fact-checking y la detección de noticias falsas. Para el caso de detección de noticias falsas ha sido ampliamente estudiada usando diferentes algoritmos pertenecientes al área de procesado de lenguaje natural, aprendizaje por máquina y aprendizaje profundo [4, 1, 9, 8].

Una limitante que tienen estos métodos es que la gran mayoría no ofrecen un mecanismo de fact-checking que justifique la decisión tomada, ya que solamente procesan todo el texto del cual se obtiene directamente la decisión. Por otro lado, la automatización del fact-checking trata de abordar este problema, pero todavía sigue siendo un tema en desarrollo [7] .

En este trabajo se propone un sistema que permita clasificar la veracidad de la noticia y muestre los argumentos (fragmentos de texto) en los que se basó para tomar la decisión. Esto impulsado por aprendizaje por refuerzo ya que ésta técnica brinda un alto dinamismo de lo que se puede hacer con la información analizada.

2. Trabajos relacionados

El uso de aprendizaje por refuerzo se ha enfocado a detectar noticias falsas haciendo un análisis de la información que transita en medios sociales [3, 5] . Se hace un enfoque en el estudio de la actividad de los usuarios, sus preferencias, su relación con otros usuarios y la información que comparten para que un agente sea capaz de clasificar, detectar y mitigar las noticias fraudulentas que pueden llegar a compartirse.

Estas investigaciones han demostrado resultados alentadores con diferentes volúmenes de datos, sin embargo, estas técnicas no ofrecen una retroalimentación explícita de fact-checking.

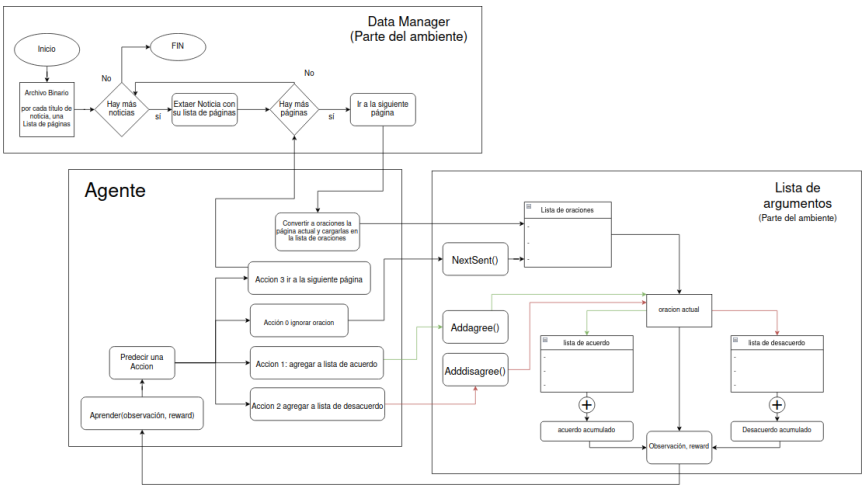


Fig. 1. Arquitectura propuesta para algoritmo de aprendizaje por refuerzo.

No obstante, el trabajo presentado en [12] es una de las principales inspiraciones para esta investigación. En él se implementa un sistema de extracción de información por aprendizaje por refuerzo profundo, el cual tiene como tarea extraer datos relevantes de paginas de internet de noticias que están relacionadas a tiroteos públicos.

Con dicha extracción se busca conseguir el valor de datos muy puntuales, tales como lugar del incidente, responsables, número de víctimas y lesionados. Con cada página que se analiza, el agente tiene la posibilidad de cambiar los valores de los datos dependiendo del contenido de la noticia y de las relaciones que puede llegar a tener con anteriores que se han visitado; entonces, de esta manera se logra obtener los datos correctos sobre un acontecimiento. Por lo tanto, esta lógica de extracción de información mediante la navegación de páginas de internet se extrapoló en este trabajo para poder hacer detección de noticias falsas.

3. Sistema propuesto

El sistema propuesto consiste en que un agente sea capaz de analizar el contenido de páginas de internet que están relacionadas con alguna noticia, con el fin de determinar si hay evidencia suficiente para clasificar la noticia como verdadera o falsa, es decir, hacer fact-checking. Este análisis incluye todo el texto de la página, como pueden ser artículos periodísticos, anuncios, pies de páginas, hipervínculos, comentarios, descripciones de imágenes, entre otros.

Por cada página relacionada, el agente podrá fragmentar todo el texto en pequeñas oraciones para procesar una por una y clasificarlas de tres maneras diferentes: si la oración apoya a la noción de que la noticia sea verdadera, falsa o que no es relevante. Para los casos de que la clasificación sea verdadera o falsa, el agente guardará las oraciones en conjuntos respectivos a la clasificación, para que cuando se termine de analizar las páginas suficientes, se comparen los conjuntos y se determine la fidelidad de la noticia.

Tabla 1. Distribución de datos de entrenamiento y de evaluación.

Conjunto de datos	No. de Noticias	No. de Noticias verdaderas	No. de páginas verdaderas	No. de Noticias falsas	No. de páginas falsas
Entrenamiento	1, 164	563	5, 626	604	5, 994
Evaluación	206	104	1, 038	102	1, 020
Total	1,370	667	6,664	703	7,014

A estos conjuntos de oraciones se les llama listas de argumentos. La clasificación de una oración no solo va a depender del texto de la misma, sino también de otros factores propios o ajenos a ella, por ejemplo: la similitud que tiene con respecto a las oraciones que ya estaban guardadas, la cantidad de oraciones que ya se guardaron en la lista de argumentos, la cantidad de palabras de la oración insertada, y por último, si se ha repetido muchas veces un mismo comportamiento en un intervalo de tiempo.

El agente también podrá ser capaz de decidir hasta qué punto dejar de leer la página e ir a la siguiente. Esto se implementa por dos razones: para tener una simulación del comportamiento humano más precisa y para evitar procesar todo el texto de todas las páginas, lo cual puede llegar a ser muy costoso computacionalmente.

Para la implementación de todas las características mencionadas, se elaboró una arquitectura de software basada en tres módulos principales: el agente, manejador de datos y listas de argumentos. Estos dos últimos definen al ambiente en el cual el agente puede llevar acabo acciones, en el sentido del Aprendizaje Supervisado. En la figura 1 se ilustra la comunicación que tienen las partes mencionadas. Cada uno de estos módulos se profundizarán más a detalle en las siguientes secciones.

3.1. Conjunto de datos

Para buscar en internet páginas con respecto a noticias falsas y verdaderas, se uso un conjunto de datos previamente clasificado[3], el cual contiene 17,903 noticias clasificadas, por lo que se tomó solamente el título y la etiqueta de veracidad de las primeras 1370 noticias. El título servirá para buscar páginas relacionadas a la noticia.

El escenario ideal es realizar la recolección de páginas de internet de una noticia mediante el uso de un motor de búsqueda mientras se entrena o evalúa. Sin embargo, este proceso resulta muy lento para el aprendizaje del agente, debido a que se gasta mucho tiempo en la carga de cada página.

Entonces, para resolver este problema, se optó por primero buscar una gran cantidad de páginas de manera paralela mediante técnicas de Web scarping y guardar todos los resultados en un único conjunto de datos. La división en conjunto de entrenamiento y de evaluación de las páginas recabadas se puede apreciar en la tabla 1.

Con la información presentada en la tabla 1 se puede constatar que la proporción entre páginas de noticias verdaderas y falsas es en buena medida proporcional, por lo tanto, no habrá un sesgo significativo hacia alguno de los dos conjuntos a la hora del entrenamiento.

[3] Clément Bisaillon. (2016, Diciembre). Fake and real news dataset, 8.82. Recuperado el 2022, Agosto desde: www.kaggle.com/datasets/clmentbisaillon/fake-and-real-news-dataset

3.2. Ambiente

El ambiente lo definen los módulos del manejador de datos y el de listas de argumentos. Ambos componentes sirven para administrar el manejo de las noticias y sus respectivas páginas para que sean usados por el agente de manera muy intuitiva. A continuación se ofrece una explicación breve de cada entidad.

a) **Manejador de datos:** Esta entidad es la encargada de cargar el conjunto de datos e ir extrayendo de él las noticias con sus respectivas páginas. Para poder tener un control básico de la extracción de las noticias y sus páginas se tienen los siguientes comportamientos:

   – Obtener siguiente noticia: Extrae la información de una noticia, es decir, título, veracidad y lista de páginas.

   – Ir a siguiente página: Se obtiene la siguiente página de internet que se recabó con respecto a la noticia revisada por el agente.

b) **Lista de argumentos:** Esta es la estructura de datos que define las lista de acuerdo (LA), de desacuerdo (LD) y una lista que contiene todas las oraciones por revisar de la página actual (LO). Esta entidad es usada por el agente para ir leyendo la página e ir guardando y clasificando las oraciones que sean de relevancia. Este módulo tiene las siguientes funciones principales:

   – Cargar oraciones: Una vez que el agente ha dividido una página en oraciones, estas son guardadas en una lista de oraciones (LO) con el fin de poder ser iteradas en orden por el mismo agente.

   – Agregar oración a la lista de acuerdo o desacuerdo: Cuando el agente decide clasificar la oración actual, esta es guardada en la lista correspondiente.

   – Obtener el vector acumulado de las listas de acuerdo y desacuerdo: Para tener una representación del estado actual de cada lista que sea compatible con la entrada de una red neuronal, se usa la representación a través de embeddings de cada oración, para cada lista se obtiene un promedio de todas las oraciones que posee cada lista.

Es de gran importancia mencionar que las oraciones, aparte de tener una representación en texto plano, cuentan con una vectorial de dimensionalidad 300 ($\mathbb{R}^{300}$), dicha representación se obtiene con base a vectores de palabras previamente entrenados por la librería Spacy[4].

Con lo anterior mencionado, el estado (S) del ambiente en un determinado tiempo consiste en un vector de 900 escalares reales ($\mathbb{R}^{900}$). Los primeros 300 valores corresponden al vector acumulado de la lista de acuerdo, los siguientes 300 a la oración actual y los últimos 300 al vector acumulado de la lista de desacuerdo.

En la ecuación 1 se define una expresión matemática para la formación del estado del ambiente, teniendo en cuenta que el iterador i hace referencia al número de la oración que está leyendo el agente sobre la página actual:

[4] English · spaCy Models Documentation, spacy.io/models/en.

$$S = \text{concat}\left(\sum_{m=0}^{|\,\text{LA}\,|}(\text{LA}_m),\quad \text{LO}_i,\quad \sum_{k=0}^{|\,\text{LD}\,|}(\text{LD}_k)\right). \tag{1}$$

3.3. Agente

Este es el módulo que define el comportamiento a aprender a través del aprendizaje profundo que modela la política usada para la clasificación de las oraciones en sus respectivas listas. El agente es el encargado de llevar las acciones dentro del ambiente, obtener recompensas positivas y/o negativas y ajustar los parámetros de la red neuronal que predice la probabilidad de la siguiente acción. El agente lo definen tres elementos: la acciones a realizar en el ambiente, algoritmo de aprendizaje por refuerzo y la política.

– **Acciones:** En la figura 1 se puede determinar que el agente cuenta con 4 acciones diferentes: Ignorar la oración actual(0), agregar la oración a la lista de acuerdo (1), agregar la oración a la lista de desacuerdo (2) y pasar a la siguiente página (3).

– **Algoritmo de aprendizaje por refuerzo y política:** La librería utilizada para implementar algoritmos de aprendizaje por refuerzo profundo fue stable-baselines 3[5]. En ella se implementan diferentes algoritmos, como lo son: Proximal Policy Optimization (PPO), Advantage Actor Critic (A2C), Deep Deterministic Policy Gradient (DDPG), Deep-Q-Network (DQN), entre otros [2]. Para fines de este trabajo, se tomó al algoritmo DQN como algoritmo para el agente. Así mismo, la política $\pi_\theta(s, a)$ utilizada es un perceptrón de dos capaz con 64 neuronas cada una (MlpPolicy).

Además, el algoritmo implementa una memoria de reproducción de experiencia (Experience Replay) [10], el cuál es un buffer que, teniendo en cuenta un tiempo t, va guardando tuplas de la forma $(S_t, A_t, R_{t+1}, S_{t+1})$ con el fin de que el agente pueda reutilizar experiencias pasadas para mejorar su predicción. Esta memoria tiene como objetivo acelerar la velocidad de aprendizaje y convergencia del agente.

3.4. Sistema de recompensas

Para poder tener una clasificación de oraciones mas enriquecida, es decir, que se tengan oraciones que brinden información relevante para la determinación de la veracidad, fue necesario implementar un sistema de recompensas que castiga o premia al agente. El sistema lo conforman 5 criterios, los cuales están modelados completamente con una función sigmoide de la siguiente forma:

$$\text{r(x)} = \frac{\alpha}{1 + (e + \beta)^{-(x - \text{movx})}} + \text{movy}, \tag{2}$$

donde α, β, movx, movy son parámetros que se pueden ajustar para cada recompensa con el fin de obtener el comportamiento esperado.

[5] Stable-Baselines3 Docs - Reliable Reinforcement Learning Implementations; Stable Baselines3 1.8.0a10 documentation, stable-baselines3.readthedocs.io/en/master/.

Tabla 2. Resultado del proceso de entrenamiento de 500 mil acciones sobre modelos de tipo DQN.

	Banderas	Empates	Aciertos	Errores	Aciertos/Errores	Recompensa Total
0	**00100**	**185**	**2257**	**1252**	**1.80**	**4020.90**
1^*	11101	403	2478	2016	1.23	521065.56
2^*	10111	177	1217	1022	1.19	134290.48
3	**01101**	**470**	**2480**	**2168**	**1.14**	**214040.08**
4	01110	145	1067	1148	0.93	-85930.09
5	00110	170	1112	1118	0.99	-158470.64
6^*	11111	164	1106	1150	0.96	143827.32
7	01111	192	1057	1222	0.86	-55673.99
8^*	**10101**	**337**	**2508**	**1907**	**1.32**	**444249.23**
9^*	10110	180	1129	996	1.13	107028.92
10	01100	203	1127	1306	0.86	69829.52
11^*	11110	176	1116	1041	1.07	132727.24
12	00111	144	1188	1198	0.99	-136178.49
13	00101	676	2798	2532	1.11	116510.56
14^*	11100	225	1662	1201	1.38	275583.66
15^*	10100	206	1506	1312	1.15	337458.05

La razón del por qué se escogió una senoidal es para procurar que las recompensas estén acotadas en un cierto intervalo y evitar que el agente explote un comportamiento en especifico que lo haga conseguir una recompensa muy grande o pequeña. A continuación se ofrece una explicación de cada criterio de recompensa:

1. Longitud de las listas: Tiene el objetivo de procurar que las listas no queden vacías al momento de la decisión de la veracidad y evitar indeterminaciones (empates).
2. Repetición de la acción 3 (salto de página): La recompensa consiste en limitar los cambios de páginas que hace el agente, con el propósito de promover la lectura de la información.
3. Decisión final de la veracidad de la noticia: Cuando se han terminado de revisar todas las páginas de una noticia, se calcula la diferencia de longitud entre las listas; si la lista de acuerdo tiene más oraciones, la noticia se considera verdadera, en cambio, si la lista de desacuerdo tiene más elementos se considera falsa.

   Si tienen la misma longitud se considera como una indeterminación (empate). Dicha diferencia es la variable x de la ecuación 2, donde $\mathrm{r(x)}$ representa la certeza que tiene el agente de su decisión, de tal forma que si falla en el pronóstico se tendrá un castigo -$\mathrm{r(x)}$, pero si acierta se tendrá una recompensa $\mathrm{r(x)}$.
4. Longitud de la oración: Para evitar que las oraciones sean muy cortas.
5. Similitud de la oración: Se compara que tan similar es la oración insertada con respecto a las demás oraciones previamente guardadas. La similitud con cada oración se calcula con la similitud del coseno, la cual consiste en calcular el ángulo que forma el vector de la oración insertada con el vector de las oración que ya están en la lista.

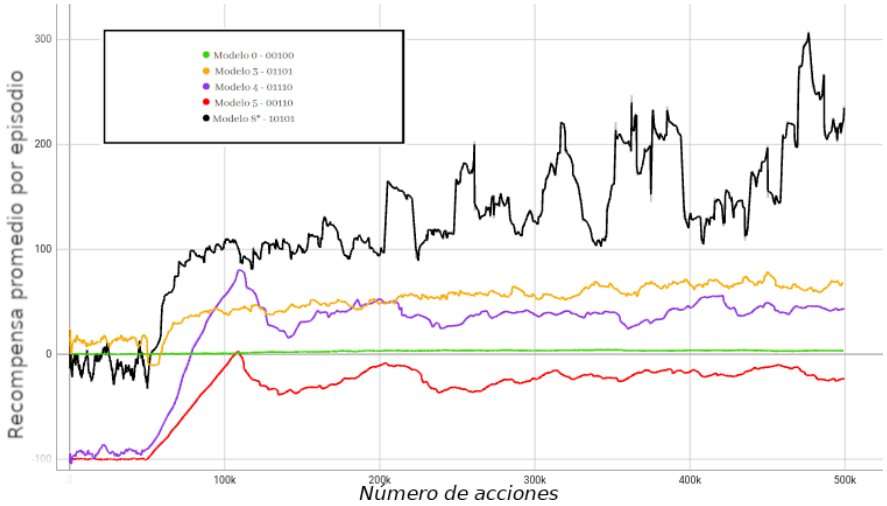


Fig. 2. Gráfica de recompensa a lo largo de los 500 mil acciones de los modelos.

Es de crucial importancia mencionar que para el proceso de entrenamiento, cada vez que el agente pasa de página (acción 3), se le permite ver la veracidad real de la noticia y la compara con la predicción que lleva hasta el momento, para así generar una recompensa muy pequeña, es decir, se le va guiando al agente en el proceso de clasificación de oraciones.

Sin embargo, en un escenario real, la veracidad real de la noticia no va a estar definida, por lo tanto, en el proceso de evaluación ya no se lleva acabo esta guía. Con esto se observó que si en el proceso de entrenamiento no se realiza la guía, el agente no aprende óptimamente; por otra parte, si en el proceso de evaluación se quita o no la guía, los resultados son los mismos para ambos casos.

4. Resultados

Para evaluar el sistema propuesto se crearon modelos que tenían ciertos criterios de recompensas activados y otros desactivados, pero siempre con la recompensa de decisión final activada, debido a que es de interés que se determine si una noticia es falsa o no. Por lo tanto, se crearon 16 modelos diferentes con la alternación de recompensas.

Para poder diferenciar cada modelo, se le asigno a cada uno un código binario que representa las recompensas activadas y desactivadas. La notación va de derecha a izquierda de tal forma que se tiene lo siguiente:

- 2^0 = Recompensa por tamaño de lista.
- 2^1 = Recompensa por repetición de acción de paso de página.
- 2^2 = Recompensa de decisión de veracidad de noticia. Siempre está activada.
- 2^3 = Longitud de la oración insertada.
- 2^4 = Similitud de la oración en la inserción a la lista de argumento.

Tabla 3. Tabla con matriz de confusión y métricas por cada modelo.

Modelo	Banderas	Empates	TN	FP	FN	TP	Exactitud	F1-score
0	**00100**	**1**	**85**	**17**	**25**	**78**	**0.795**	**0.788**
1^*	11101	18	32	59	25	72	0.553	0.632
2^*	10111	7	37	63	38	61	0.492	0.547
3	**01101**	**7**	**62**	**34**	**10**	**93**	**0.779**	**0.809**
4	01110	4	3	98	17	84	0.431	0.594
5	00110	2	70	31	65	38	0.529	0.442
6^*	11111	3	30	72	15	86	0.571	0.664
7	01111	6	13	87	19	81	0.470	0.604
8^*	**10101**	**6**	**54**	**45**	**23**	**78**	**0.660**	**0.696**
9^*	10110	3	22	78	23	80	0.502	0.613
10	01100	16	27	69	29	65	0.484	0.570
11^*	11110	8	21	75	27	75	0.485	0.595
12	00111	3	86	13	75	29	0.567	0.397
13	00101	9	4	91	2	100	0.528	0.683
14^*	11100	22	32	53	20	79	0.603	0.684
15^*	10100	10	18	77	32	69	0.444	0.559

Independientemente de las recompensas, las características más importantes que comparten todos los modelos son: una taza de aprendizaje (ϵ) de $0{,}0001$, coeficiente de actualización suave (τ) de $1{,}0$, un factor de descuento (γ) de $0{,}99$, frecuencia de entrenamiento de 4 pasos y un tamaño de buffer de reproducción de experiencia de $50,000$ pasos.

4.1. Proceso de entrenamiento

Para entrenar los modelos se usaron 500 mil acciones. El agente puede ejecutar tantas épocas como sean necesarias sobre el conjunto de entrenamiento definido en la tabla 1 para alcanzar el número objetivo de acciones. En la tabla 2 se puede ver el resultado final después de ejecutar los 500 mil pasos.

El contenido de la tabla consiste en las banderas utilizadas en cada modelo, las indeterminaciones (empates) que hubo, los aciertos y error totales, el cociente de estos dos anteriores y la recompensa total que se obtuvo después de todas las acciones. Los modelos marcados con un asterisco, son aquellos que no lograron una convergencia después de todos los pasos.

En la tabla de entrenamiento se puede obtener una primera caracterización de los modelos, y por lo tanto, también sobre los criterios de recompensa. Algunas connotaciones relevantes sobre los datos presentados son las siguientes:

1. Los modelos no convergentes tienen la característica común de tener la recompensa de similitud por inserción activada (2^4), lo que quiere decir que para tener recompensas no tan variables, se deben sintonizar los parámetros de dicha recompensa, de tal forma que no se tengan valores con grandes cambios.

2. Si el cociente de aciertos y errores es mayor a uno, quiere decir que en algún momento el modelo comenzó a clasificar correctamente las noticias, o al menos en los últimos pasos del entrenamiento. El modelo que mostró mejor cociente fue el 0, el cual solo depende de la decisión final basada en el tamaño de las listas.

   No obstante, el modelo 3 también presenta un buen cociente y una recompensa mucho mayor al modelo 0, por lo tanto, se puede decir que el modelo 3 fue el que mejor cumplió la tarea de fact-checking; aparte de ser convergente. Cabe mencionar que el modelo 8 presenta el mejor coeficiente y recompensa entre todos los modelos no convergentes.

3. La recompensa de repetición de pasos (2^4) genera una recompensa extremadamente negativa, este efecto se puede ver claramente en los modelos 4, 5, 6, 7 y 12. Esto ocasionó que los modelos no aprendieran adecuadamente y no pudieran hacer la clasificación óptimamente.

En la figura 2 se puede ver el promedio de recompensa obtenida por cada episodio (por cada noticia clasificada) de los modelos a lo largo de todo el entrenamiento (500 mil acciones). Para fines de simpleza de la gráfica, se seleccionaron modelos representativos con características comunes a los otros, es decir, se seleccionó un modelo que no sea convergente (modelo 8), otro con recompensa negativa (modelo 5), con mayor número de aciertos y recompensa positiva (modelo 0 y 3), y finalmente uno con un salto de recompensa muy grande (modelo 4).

4.2. Proceso de evaluación

Se utilizaron 206 noticias para evaluar los modelos. En la tabla 3 se en listaron los elementos de una matriz de confusión como lo son los verdaderos negativos (TN), falsos positivos (FP), falsos negativos (FN) y verdaderos positivos (TP). Aparte, se calcularon las métricas de exactitud y f1-score.

Para el cálculo de estas no se consideraron los empates. Una característica importante sobre los modelos, es la recompensa producida por la diferencia que hay entre la lista de acuerdo y de desacuerdo para determinar la veracidad de la noticia (2^2).

Si la magnitud de la recompensa es grande, la diferencia de las listas también lo es, por lo que se puede decir que la certeza de decisión es grande. Extrapolando, cuando la magnitud de recompensa es pequeña, hay menos certeza en la decisión. La certeza se produce independientemente de que el agente se equivoque o no.

En la tabla 4 se caracterizaron estos valores de certeza por cada clasificación de la matriz de confusión mediante la media y la desviación estándar. La mayor certeza que se puede obtener es de 3.4 debido a la cota superior de la sigmoide que modela la recompensa. Lo que se busca en los valores de la tabla 4 es que las desviaciones estándar sean pequeñas, debido a que esto representa que el agente aprendió patrones bien definidos que le permiten tener valores de certeza mayormente constantes.

Por otra parte, el comportamiento deseado para la media de cada clasificación es tal que para las noticias en TN y TP sea lo más grande posible, ya que esto quiere decir que el agente puede clasificar correctamente con mucha certeza; de forma contraria, para las noticias mal clasificadas, es decir FP y FN, se espera que el valor de la media sea baja, porque con ello se podría decir que el agente estuvo cerca de clasificar correctamente.

Tabla 4. Tabla con medias y desviaciones estándar para la certeza de los modelos.

	Banderas	TN-μ	TN-σ	FP-μ	FP-σ	FN-μ	FN-σ	TP-μ	TP-σ
0	**00100**	**3.221**	**0.644**	**2.418**	**1.383**	**3.126**	**0.753**	**2.670**	**1.152**
3	01101	2.564	1.188	2.066	1.412	2.333	1.122	2.512	1.253
6	11111	2.503	1.276	2.799	1.162	2.458	1.121	2.840	1.127
8*	10101	2.357	1.212	2.110	1.497	2.010	1.210	2.430	1.238
14*	11100	1.988	1.315	1.561	1.493	1.935	1.305	2.221	1.392
15*	10100	2.335	1.298	1.504	1.411	2.381	1.092	2.257	1.370

Las indeterminaciones (empates) mostrados en la tabla 3 están reflejados en los valores de la tabla 4. Los modelos que casi no presentan empates, como lo son el 0, 3, 6 y 8, tienen medias muy grandes y casi sin variaciones. De manera contraria, los modelos 14 y 15, los cuales tienen un gran número de empates, poseen medias muy pequeñas y desviaciones muy grandes, lo que quiere decir que los agentes no desarrollaron una política que marque patrones claros para la identificación de noticias falsas.

4.3. Análisis de lista de argumentos del mejor modelo

El modelo a analizar va a ser el número 3, debido a que tiene un buen puntaje aciertos/errores, tiene una recompensa acumulada grande, tiene 3 sistemas de recompensa activados, tiene una exactitud del 78 %, tiene medias y desviaciones estándar de certeza aceptables y además es convergente. Para las listas de argumentos solo se pusieron algunos elementos relevantes.

- Para noticia verdadera:
  - Título: House lifts block on Google-hosted apps, Yahoo Mail remains blacklisted.
  - Lista de acuerdo: [”Wednesday, May 18, 2016. An illustration picture shows the logos of Google and Yahoo connected with LAN cables in a Berlin office October 31, 2013.”, ”10:59pm EDT House lifts block on Google apps, Yahoo Mail remains blacklisted. An illustration picture shows the logos of Google and Yahoo connected with LAN cables in a Berlin office October 31, 2013.”, ”Bangladesh asks SWIFT to give access to technicians on cyber heist.”, ...] [13 elementos].
  - Lista de desacuerdo: [’REUTERS Pawel Kopczynski.’, ’AgainView Next.’, ’Bangladesh has asked SWIFT to help its police question.technicians sent by the global financial network to Dhaka to connect a new bank’,...] [8 elementos].
- Para noticia falsa:
  - Título: Seth Meyers Torches Trump’s NAFTA Flip-Flop With Awesomely Dirty Joke (VIDEO).
  - Lista de acuerdo: [”Seth Meyers Ridicules Trump’s Hurricane Guns Theory Menu We’ve Got Hollywood Covered Log in .”, ’Steve Pond Movie Reviews Box Office Toronto Toronto Video Studio Sundance Cannes Awards.’, ’Power Women Summit TheGrill Screenings Screenings RSVP Webinars Archive BE Conference 2021 Videos’,...’] [11 elementos].

– Lista de desacuerdo: ['According to one Trump official, "It was almost too stupid for words" — and Meyers largely agreed.', 'But what hit Meyers the most wasn't the idea of the hurricane gun itself; it was that Trump apparently took that as a forgone conclusion and had a series of steps he wanted to take in response',...] [18 elementos].

Analizando las listas de la noticia anterior y de otros del proceso de evaluación, se pudo identificar un comportamiento muy característico. Cuando la noticia es verdadera, el agente tiende a llenar la lista de acuerdo con oraciones de lenguaje muy formal que proporcionan información extra sobre la noticia a investigar, mientras que en la lista de desacuerdo se guardan oraciones que no ofrecen informacion adicional a la noticia, en cambio, se guarda texto que está contenido en la página.

Para el caso de que la noticia sea falsa, el comportamiento anterior descrito se invierte, es decir, la lista de acuerdo se llena con información que puede ser irrelevante para la noticia, pero en la lista de desacuerdo se llena con oraciones de un lenguaje más informal, que están escritas en primera o segunda persona y cuyo objetivo es declarar falsamente contra alguna entidad o persona.

Cabe destacar que el decir que una oración no aporta información a la noticia, no implica que sea insignificante para la detección de la veracidad de la misma, ya que hay que recordar que no solo se esta tomando en cuenta el texto de la noticia, sino el contexto de la página de internet que la contiene.

5. Conclusiones y trabajo a futuro

En este trabajo se desarrolló un agente impulsado por Aprendizaje por Refuerzo Profundo que es capaz de recabar información relevante de una noticia y clasificar si la misma es falsa o verdadera, tomando como fuente de información páginas de internet que fueron recabadas por algún motor de búsqueda.

Esto con la ayuda de dos listas de oraciones, una que especifique oraciones que apoyen a la idea de que la noticia sea verdadera y otra con oraciones que apoyen a la idea de que sea falsa. Para lograr que el agente desarrollara comportamientos deseados para tener un fact-checking y clasificación óptimos, se produjeron 5 criterios de recompensas, que activándolos y desactivándolos en diferentes combinaciones, se generaron 16 modelos diferentes, así analizando el comportamiento producido para cada combinación.

Esta profundización llegó a ser algo complicada en cierto punto, ya que algunas recompensas no sirven bien aisladamente, pero en combinación con otras pueden mejorar el desempeño de un agente. Como resultado de dicho estudio se obtuvo un modelo con un $78\,\%$ de exactitud a la hora de la clasificación y que recaba óptimamente oraciones con ciertas características que permiten tener un contexto mayor de la noticia.

Además, se determinó la buena certeza de decisión y convergencia del mejor modelo. Para trabajo futuro de mejora del sistema se pueden optar por diferentes estrategias. Un cambio muy directo al sistema, que puede resultar en mejoras significativas, es la sintonización de los parámetros de los modelos matemáticos que definen a las recompensas para conseguir mejores comportamientos.

Esto sería necesario porque algunos criterios producían recompensas muy grandes o muy pequeñas y aparte variantes. O en su defecto, agregar otros criterios de recompensa que se crean pertinentes para mejorar el fact-cheking, la investigación y clasificación del agente.

Referencias

1. Aphiwongsophon, S., Chongstitvatana, P.: Detecting fake news with machine learning method. In: 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, pp. 528–531 (2018) doi: 10.1109/ecticon.2018.8620051
2. Arulkumaran, K., Deisenroth, M. P., Brundage, M., Bharath, A. A.: Deep reinforcement learning: A brief survey. IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26–38 (2017) doi: 10.1109/msp.2017.2743240
3. Aymanns, C., Foerster, J., Georg, C. P.: Fake news in social networks. SSRN Electronic Journal (2022) doi: 10.2139/ssrn.4173312
4. de-Oliveira, N. R., Pisa, P. S., Lopez, M. A., de Medeiros, D. S. V., Mattos, D. M. F.: Identifying fake news on social networks based on natural language processing: Trends and challenges. Information, vol. 12, no. 1, pp. 38 (2021) doi: 10.3390/info12010038
5. Farajtabar, M., Yang, J., Ye, X., Xu, H., Trivedi, R., Khalil, E., Li, S., Song, L., Zha, H.: Fake news mitigation via point process based intervention. In: Proceedings of the 34th International Conference on Machine Learning, vol. 70, pp. 1097–1106 (2017) doi: 10.48550/arXiv.1703.07823
6. Farkas, J., Schou, J.: Fake news as a floating signifier: Hegemony, antagonism and the politics of falsehood. Javnost - The Public, vol. 25, no. 3, pp. 298–314 (2018) doi: 10.1080/13183222. 2018.1463047
7. Hanselowski, A., Stab, C., Schulz, C., Li, Z., Gurevych, I.: A richly annotated corpus for different tasks in automated fact-checking (2019) doi: 10.48550/ARXIV.1911.01214
8. Jwa, H., Oh, D., Park, K., Kang, J., Lim, H.: exBAKE: Automatic fake news detection model based on bidirectional encoder representations from transformers (BERT). Applied Sciences, vol. 9, no. 19, pp. 4062 (2019) doi: 10.3390/app9194062
9. Kumar, S., Asthana, R., Upadhyay, S., Upreti, N., Akbar, M.: Fake news detection using deep learning models: A novel approach. In: Transactions on Emerging Telecommunications Technologies, vol. 31, no. 2 (2019) doi: 10.1002/ett.3767
10. Lin, L.: Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine Learning, vol. 8, pp. 293–321 (2004) doi: 10.1007/BF00992699
11. Naeem, S. B., Bhatti, R., Khan, A.: An exploration of how fake news is taking over social media and putting public health at risk. Health Information and Libraries Journal, vol. 38, no. 2, pp. 143–149 (2020) doi: 10.1111/hir.12320
12. Narasimhan, K., Yala, A., Barzilay, R.: Improving information extraction by acquiring external evidence with reinforcement learning. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 2355–2365 (2016) doi: 10.18653/v1/D16-1261
13. Nieves-Cuervo, G. M., Manrique-Hernández, E. F., Robledo-Colonia, A. F., Grillo, E. K. A.: Infodemia: Noticias falsas y tendencias de mortalidad por COVID-19 en seis países de América Latina. Revista Panamericana de Salud Pública, vol. 45, pp. 1 (2021) doi: 10.26633/rpsp.2021.44
14. Pennycook, G., Rand, D. G.: The psychology of fake news. Trends in Cognitive Sciences, vol. 25, no. 5, pp. 388–402 (2021) doi: 10.1016/j.tics.2021.02.007

15. Rashkin, H., Choi, E., Jang, J. Y., Volkova, S., Choi, Y.: Truth of varying shades: Analyzing language in fake news and political fact-checking. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics, pp. 2931–2937 (2017) doi: 10.18653/v1/d17-1317
16. Tandoc, E. C., Lim, Z. W., Ling, R.: Defining "fake news". Digital Journalism, vol. 6, no. 2, pp. 137–153 (2017) doi: 10.1080/21670811.2017.1360143
17. Wasserman, H., Madrid-Morales, D.: An exploratory study of "fake news" and media trust in Kenya, Nigeria and South Africa. African Journalism Studies, vol. 40, no. 1, pp. 107–123 (2019) doi: 10.1080/23743670.2019.1627230
18. Zhang, X., Ghorbani, A. A.: An overview of online fake news: Characterization, detection, and discussion. Information Processing and Management, vol. 57, no. 2, pp. 102025 (2020) doi: 10.1016/j.ipm.2019.03.004
19. Zhou, X., Zafarani, R.: A Survey of fake news. ACM Computing Surveys, vol. 53, no. 5, pp. 1–40 (2020) doi: 10.1145/3395046
20. Zlatkova, D., Nakov, P., Koychev, I.: Fact-checking meets fauxtography: Verifying claims about images. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Association for Computational Linguistics, pp. 2099–2108 (2019) doi: 10.18653/v1/d19-1216

ISSN 1870-4069

Diseño de un módulo para la detección de ciberbullying en la red social Twitter utilizando lenguaje natural en una aplicación móvil Android implementado en un entorno universitario

Lissete Rosete Rosas[1], Luis Ángel Reyes Hernández[1], Beatriz Alejandra Olivares Zepahua[1], Ignacio López Martínez[1], Laura Angélica Décaro Santiago[2]

[1] Instituto Tecnológico de Orizaba, División de Investigación y Estudios de Posgrado, México

[2] Universidad Autónoma del Estado de México, Centro Universitario UAEM, México

{m16011208, luis.rh, beatriz.oz, ignacio.lm}@orizaba.tecnm.mx, ladecaros@uaemex.mx

Resumen. El ciberacoso es el tipo de acoso que se presenta a través de dispositivos digitales y es un riesgo permanente al que están expuestos los jóvenes debido a la cercanía que tienen con las redes sociales y el Internet. En respuesta al aumento y prevalencia del ciberacoso se ha apostado por implementar técnicas que prevengan este tipo de incidencias, sin embargo, pocas soluciones se encargan de ofrecer una detección temprana de indicios de este fenómeno entre los jóvenes. En este artículo, se propone una arquitectura para el desarrollo de un módulo que detecte situaciones de ciberacoso en la red social Twitter empleando procesamiento de lenguaje natural, dicho modulo será implementado en una aplicación móvil base desarrollada para Android, que tiene como objetivo detectar ocurrencias de bullying tradicional. Además, se describe la obtención de la bolsa de palabras comprendidas en el lenguaje verbal violento de las redes sociales empleado por jóvenes universitarios; así como los diseños de interfaces para el desarrollo del módulo de una aplicación de monitorización de casos de ciberacoso para contribuir en la reducción de ocurrencias de este fenómeno.

Palabras clave: Aplicación móvil, bolsa de palabras, ciberacoso, procesamiento de lenguaje natural.

Design of a Module for the Detection of Cyberbullying in the Social Network Twitter Using Natural Language in an Android Mobile Application Implemented in a University Environment

Abstract. Cyberbullying is the type of bullying that occurs through digital devices and is a permanent risk to which young people are exposed due to the

proximity they have to social networks and the Internet. In response to the increase and prevalence of cyberbullying, there has been a commitment to implement techniques to prevent this type of incident, however, few solutions are responsible for providing early detection of signs of this phenomenon among young people. In this article, we propose an architecture for the development of a module that detects situations of cyberbullying in the social network Twitter using natural language processing, this module will be implemented in a mobile application developed for Android, which aims to detect occurrences of traditional bullying. In addition, it is described the obtaining of the bag of words included in the violent verbal language of social networks used by young university students; as well as the interface designs for the development of the module of an application for monitoring cases of cyberbullying to contribute to the reduction of occurrences of this phenomenon.

Keywords: Mobile app, bag of words, cyberbullying, natural language processing.

1. Introducción

En la actualidad la sociedad se encuentra sumergida en una constante evolución tecnológica, donde el Internet y las comunicaciones son los protagonistas. Estos cambios en la sociedad impactan en el contexto social donde los jóvenes o adolescentes se desenvuelven e interactúan; así como existen numerosas ventajas que trae consigo la pequeña brecha entre la tecnología y los adolescentes, también existe la preocupación de que el entorno se vuelva dañino, pues pueden ocurrir situaciones de riesgo en las que el estado emocional, físico o mental de los adolescentes se vea afectado. Por otro lado, la violencia y agresión en el sector educativo es un problema persistente alrededor del mundo; existe evidencia de esfuerzos para prevenir el bullying y ciberbullying, además de acciones para corregir esta grave situación.

En primer lugar, el cyberbullying ha tomado diversas definiciones a lo largo de su prevalencia y como es mencionado en el trabajo de Chun [1] aún existen debates sobre la definición exacta o final del término de ciberbullying; una definición de ciberbullying desarrollada a partir de la definición de bullying tradicional, es la propuesta por Hinduja y Patchin [2] que conceptualizan al fenómeno como "un daño intencional y repetido infligido mediante el uso de computadoras, teléfonos celulares y otros dispositivos electrónicos". Por lo que, sin duda, el objetivo principal de este fenómeno consiste en el uso de medios digitales para acosar, intimidad, amenazar, amedrentar o molestar a una persona o un grupo de personas mediante ataques personales o divulgación de información personal privada o falsa.

De acuerdo con el módulo sobre ciberacoso 2021 del Instituto Nacional de Estadística y Geografía (INEGI) [3] el 21.7% de la población usuaria de Internet fue víctima de ciberacoso. Además, se estima que el ciberacoso más frecuente se encuentra relacionado con el aspecto físico, a la forma de vestir y al estilo de vida.

El desarrollo de aplicaciones móviles ha ido en constante aumento, pues cada vez existen más aplicaciones que tratan de resolver una problemática en la sociedad y a su vez, son herramientas con una gran accesibilidad, debido a su uso en dispositivos informáticos inalámbricos pequeños, como teléfonos o *tablets* [4]. Por consiguiente, debido al incremento y perseverancia del ciberacoso es necesario crear herramientas

que prevengan o detecten este tipo de situaciones, con la finalidad de disminuir la cantidad de personas afectadas. Las redes sociales son el lugar más común donde las personas son intimidadas, acosadas o ridiculizadas, lo que puede tener graves consecuencias para la salud mental y emocional de las víctimas. Twitter es de las redes sociales más violentas, pues en su plataforma existen miles de casos de ciberacoso; además, la poca o nula censura en los tweets pueden indicar un alto potencial de incidentes de ciberacoso [5].

Por lo que, se pretende desarrollar un módulo que detecte situaciones de ciberacoso en la red social Twitter, este módulo será implementado en una aplicación base desarrollada para Android en el tema "Aplicación móvil de monitorización de bullying en salones de clase empleando DSM-V y el algoritmo FOAF" culminado por el alumno egresado de la maestría en sistemas computacionales, René Navarrete Tenco. La aplicación base se enfoca como medio de recolección de datos para realizar la detección de eventos de bullying dentro de un salón de clases, por medio de la observación del docente, para posteriormente hacer la monitorización del caso hasta llegar a la posible solución del evento.

En la literatura analizada existen trabajos que implementan soluciones que tienen como objetivo prevenir situaciones de ciberbullying, pero tienen la limitante de solo ser informativas. De igual forma, las aplicaciones desarrolladas para que identifiquen ciberbullying [6–11], se encuentran destinadas a usarse en países diferentes a México, por lo que el idioma es un limitante, ya que el vocabulario cambia y también cambia la manera en la que las personas se expresan de forma maliciosa.

La estructura de este artículo es la siguiente: la sección 2 abarca el estado del arte englobando los trabajos más relacionados con la detección y puntos clave del ciberacoso, en la sección 3 especifica la descripción de la arquitectura propuesta para el desarrollo del módulo sobre ciberbullying, en la sección 4 se presenta la construcción del instrumento y los resultados obtenidos para formar la bolsa de palabras que contemple las palabras o frases empleadas por los adolescentes para agredir u ofender a alguien más, a través de redes sociales; los mockups realizados para ilustrar una aproximación a las interfaces del módulo de la aplicación se exponen en la sección 5, en la sección 6, se presenta la discusión de los resultados obtenidos y finalmente en la sección 7, las conclusiones y el trabajo a futuro.

2. Trabajos relacionados

En esta sección se analiza la literatura identificada más relacionada con el ciberacoso, los factores involucrados y los trabajos de previas propuestas para la prevención y detección de este fenómeno.

En primer lugar, en [12] se obtuvieron perfiles psicológicos de adolescentes donde se enfatizaba que los jóvenes con un alto índice en conducta antisocial, tenían mayor uso de las estrategias agresivas para la resolución de conflictos y un mayor nivel en las posibilidades de estar implicados en situaciones de bullying/cyberbullying; en cualquiera de los roles posibles, víctimas, agresores y observadores. Por otro lado, en [13] se propuso un léxico de ciberacoso compuesto de 13 palabras clave (tomadas de Enchanted Learning) acompañadas de su definición y su categoría. Se propuso que el léxico presentado, tiene la capacidad de ser utilizado como diccionario en las redes

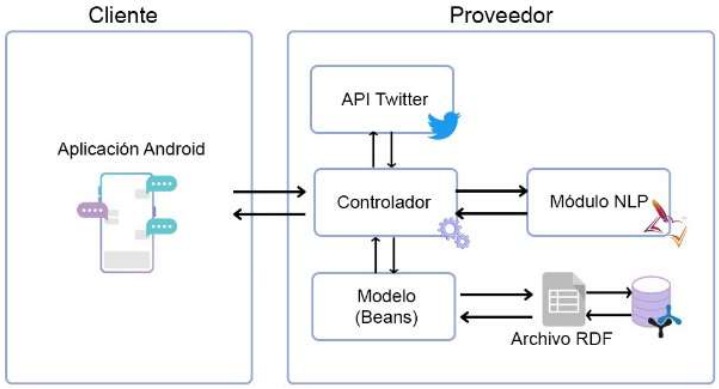


Fig. 1. Patrón arquitectónico MVC propuesto para el desarrollo del módulo de la aplicación.

sociales para considerar palabras o frases que insinúen una situación de posible ciberacoso. Noviantho *et al*. [14] construyó un modelo de clasificación para identificar conversaciones de ciberacosadores, empleando el método de minería de texto Naive Bayes y Máquinas de Vector Soporte.

El método de clasificación obtuvo 4 clases con una precisión promedio del 92.81% para Naive Bayes y un 97.22% para SVM; se concluyó que el modelo más óptimo para separar la muestra en diferentes clases fue SVM. Por su parte, en Upadhyay, *et al*. [6] se diseñó un sistema que emplea las publicaciones y los comentarios de los usuarios en Facebook, para clasificar el contenido en alguna de las categorías que se plantearon. El análisis se llevó a cabo mediante el algoritmo de Procesamiento del Lenguaje Natural permitiendo detectar los sentimientos de los usuarios y bloquear de inmediato el contenido que sea dañino para los usuarios.

De acuerdo con las investigaciones de Farag *et al*. [15] se determinó que los métodos distintos a los supervisados empleados para la detección del ciberacoso eran: codificadores automáticos, aprendizaje profundo, semisupervisado, modelado de series temporales y clustering (agrupamiento), pues se emplearon en diferentes plataformas, tales como MySpace, Twitter, YouTube, Formspring, ask.fm e Instagram para detectar situaciones de acoso.

El resultado de la investigación, demostró que el método para detectar situaciones de cyberbullying depende de la plataforma que se esté analizando. En [7] se desarrolló un modelo automatizado para identificar y medir el grado de cyberbullying en las redes sociales, además del diseño de una aplicación de Facebook que notificaría a los padres en caso de que los adolescentes sean víctimas de ciberacoso.

Como resultado del proyecto, se obtuvo la aplicación *BullyBlocker*, que identifica si un adolescente se encuentra implicado en una situación de cyberbullying y brinda información a los padres del mismo para tomar las medidas adecuadas. De forma similar, Salawu *et al*. [8] propuso una aplicación móvil diseñada para detectar y prevenir el ciberacoso en las redes sociales.

Dicha aplicación se integró por una arquitectura en la que se utiliza un modelo de aprendizaje profundo generalizado, empleado para identificar casos de ciberacoso a partir de la información básica del usuario; el modelo se entrenó para predecir etiquetas de ciberacoso y cuando lleguen nuevos mensajes o comentarios estos se clasifican de

acuerdo al modelo y si son indicios de algún tipo de ciberacoso, se eliminan o se bloquea al remitente temporalmente.

En [9] se diseñó y desarrolló una aplicación móvil (nombrada #StopBully) enfocada en mejorar la comprensión de los conceptos de bullying y cyberbullying entre los estudiantes a través del entretenimiento educativo. La aplicación se diseñó en Android y cuenta con un botón de emergencia, el cual comunica al usuario con alguna autoridad capacitada para denunciar algún caso de bullying/cyberbullying.

Foong y Oussalah [10] propusieron un sistema online que permitió la detección y el seguimiento de casos de ciberacoso en foros y comunidades en línea; este sistema se conformó por componentes básicos del lenguaje natural, contemplando insultos, amenazas y oraciones escritas en segunda persona. Se usó un sistema de aprendizaje automático y ontologías para clasificar la aparición de ese tipo de vocabulario, con la finalidad de que se emita un mensaje a la seguridad del sitio y este tome medidas necesarias.

El objetivo de [11] fue examinar los métodos existentes de detección del ciberacoso para integrarlos en una aplicación móvil que alerta a los padres en caso de que su hijo sea una potencial víctima o autor de ciberacoso. La aplicación se conformó por dos componentes, el de la aplicación móvil y el componente de aprendizaje automático. El modelo que se empleó para incorporar al aprendizaje automático fue una recolección de datos extraídos de Twitter. Teniendo como resultado buenas métricas incluidas la precisión, exhaustividad y exactitud que mostraron una eficacia notable del modelo propuesto.

3. Arquitectura

En esta sección se presenta la arquitectura propuesta para el desarrollo del módulo. La aplicación base está diseñada empleando el patrón de diseño MVC (*Model-View-Controller*, Modelo-Vista-Controlador); por consiguiente, el módulo continúa empleando esta arquitectura. MVC es un patrón de diseño ampliamente utilizado para crear aplicaciones web y es usado en casi todos los marcos de desarrollo web [16]. La arquitectura se muestra en la figura 1.

Para efectos de este proyecto se empleará la red social Twitter debido a las ventajas que proporciona, como sus pocas restricciones de privacidad ya que la gran mayoría de usuarios tiene público su perfil. La API de Twitter se puede utilizar para recuperar y analizar datos de Twitter de forma programática [17], además de que proporciona facilidades para ejecutar sin restricciones un gran número de acciones.

En MVC, el modelo consta de los objetos de dominio que modelan los problemas del mundo real. Las vistas contienen el código que permitirá la visualización de las interfaces con las que el usuario interactúa y los controladores contienen el código que permite responder a las acciones que se solicitan a la aplicación y que, de este modo, se vean reflejadas en las vistas [18].

La aplicación se encuentra desarrollada en dos partes, el lado del cliente y el lado del proveedor, lo que permite una escalabilidad factible para implementar nuevas necesidades que se requieran.

3.1. Cliente

Se le conoce de esta forma al dispositivo que realiza peticiones y consume los servicios generados por el proveedor a través de la aplicación. El cliente dispone la vista de la aplicación que está conformada por interfaces compuestas por formularios, listas, imágenes, tablas, por mencionar algunos; contiene procedimientos relacionados con la interfaz de usuario [19].

La aplicación móvil base se encuentra desarrollada para el sistema operativo Android por medio del Entorno de Desarrollo Integrado (IDE) Android Studio, que integra archivos XML (*Extensible Markup Language* – Lenguaje de Marcado Extensible); además se plantea utilizar Java como lenguaje de programación para el consumo de servicios.

3.2. Proveedor

La aplicación base se encuentra gestionada por servicios web basados en REST. Existen diferentes alternativas de tecnologías para desarrollar este tipo de servicios, para el desarrollo del módulo se optará por el marco de servicios *Jersey RESTful Web Service in Java* por medio del lenguaje de programación Java.

- Modelo: Representa la lógica de la aplicación en distintas clases encargadas de estructurar e interactuar con la información de los usuarios y eventos de posible ciberacoso.
- Controlador: Este componente se encarga de gestionar, atender y procesar las solicitudes recibidas del cliente. Con esto, el modelo y la vista se comunican para solicitar, procesar y pasar los datos necesarios a la vista para que pueda mostrarlos. Al ser un proveedor de servicios web basados en REST, tanto el proveedor como el cliente, devolverán y recibirán los datos en formato ligero para el intercambio de datos Notación de Objetos JavaScript (JSON). En el controlador se contempla la implementación de la conexión con la API de Twitter, en donde a través del registro del username de cada alumno se obtendrán los tweets emitidos respectivamente; por lo cual, para realizar solicitudes HTTP se necesita de una conexión segura utilizando la autenticación OAuth 2.0 por medio del Baerer Token, el cual es un tipo de token de seguridad compuesto de una serie de caracteres alfanuméricos que se utiliza para identificar al usuario y otorgar acceso a recursos protegidos en la plataforma de Twitter.
- Vista: El cliente descrito en la sección 3.1 será responsable de disponer la interfaz de la aplicación visible para el usuario. La vista (archivo de diseño XML) muestra los datos del modelo al usuario.

Por otro lado, para el módulo de NLP (*Natural Language Processing*), se seleccionó la herramienta Apache OpenNLP, debido a la gran cantidad de documentación y herramientas que dispone, además de la buena combinación con el entorno de desarrollo NetBeans que provee Apache. Esta tecnología, cuenta con la detección de idiomas, por lo que es adecuada para el desarrollo de este proyecto debido a que se enfocará en el idioma español.

Los tweets recuperados de los alumnos, serán preprocesados para limpiar el texto de ruido y estandarizar dicho texto para que sea fácil de comparar con respecto a la bolsa

Violencia verbal entre jóvenes

Objetivo: El objetivo del presente cuestionario, es identificar frases o palabras ofensivas dentro del entorno virtual-digital como parte del fenómeno del ciberbullying.

Los resultados serán utilizados únicamente con fines académicos y de investigación. Su contribución apoyará al desarrollo de estrategias para disminuir este fenómeno entre jóvenes, por lo que su participación es muy importante. Cabe mencionar, que los datos obtenidos serán utilizados de forma anónima.

Fig. 2. Formulario creado en Google Forms.

1. ¿Qué frases o palabras has identificado, con el propósito de discriminar a una persona?	2. ¿Qué chistes has identificado cuyo objetivo sea ridiculizar a una persona?	3. Si has observado alguna ironía que tenga por objetivo humillar a una persona, descríbela.	4. ¿Con que frases o palabras has identificado que una persona amenaza a otra?
negro, negra, azteca, deberías s	esquizofrénico, así hacía mi primo	Cuando dicen "Quiero tu autoestim	Cuando dicen que los van a dox
Negro, gorda, anorexica, delgada	Eres un enano	Cuando el chic@ es pobre y se en	Te asesinare
Naco, pendejo, estupido	Burlas hacia el físico de las perso	Cuando hablan sobre una mujer	Tu que
Negrata, jodido, indio, puto, mari	Conozco 5 gordos y tú eres 4 de e	Prófugo del ácido folico.	Desearás que tu madre te haya
Negro, mexicano, retrasado	Eres una mierda	Decirle que no tiene cerebro o que	Te voy a matar
Las mujeres no pueden porque s	Ninguno	Evidenciarlo delante de gente	Aquí no te hago nada porque ha
Negro,Gei,pobre,feo	Machistas,clasistas,humor negro e	No sé que es ironía	No quiero que hables con nadie
Comentarios machistas	Chistes sobre el color de piel de la	No la he notado	No he visto amenazas
Ese es bastante feo	Jajajajaja mira este chocolate se p	No ninguna	Vete con cuidado conmigo
Negro	Te ves muy demacrado	Cuando te tiran una indirecta de có	Te voy a agarrar a golpes

Fig. 2. Previsualización de las respuestas obtenidas.

de palabras. Cabe señalar que en la etapa actual del desarrollo del proyecto se encuentra realizándose el análisis para la selección de la técnica NLP más adecuada a las necesidades del proyecto.

4. Bolsa de palabras del lenguaje verbal violento

En esta sección se detalla la creación del instrumento para obtener las palabras o frases con las que los jóvenes se expresan violentamente a través de medios digitales, así como el resultado obtenido, contemplado en una lista o bolsa de palabras que se empleará para alimentar el algoritmo que permita identificar eventos de ciberacoso.

Ya que el objetivo de este proyecto es la detección de ciberbullying a través de lenguaje natural, será analizada la parte textual de los medios digitales y es necesario obtener la bolsa de palabras empleadas por los jóvenes para agredir, humillar, desacreditar u ofender a alguien más.

En México no se contemplan diccionarios o bolsas de palabras que permitan identificar las agresiones verbales a través de medios digitales por parte de los jóvenes o adolescentes. Para lo cual, como primer punto se clasificaron las dimensiones a abordar en relación con las manifestaciones del ciberbullying; de esta manera, se contempló la dimensión de burla, amenaza e insulto.

La tabla 1 expone cada una de las dimensiones a abordar y para que dichas dimensiones se puedan operacionalizar se estableció su significado dirigido a este proyecto, así como los medios a través de los cuales se manifiesta y los resultados o consecuencias del acto.

Tabla 1. Dimensiones de las manifestaciones de ciberbullying.

Dimensión	**Significado**	**Medio**	**Resultado**
Burla	Es el acto o conducta de provocar la vergüenza de una persona por diversión para ridiculizarlo; puede ser divertido u ofensivo según la ambigüedad de la situación, debiéndose a las interacciones personales de los humanos.	– Chistes – Bromas – Ironía	– Herir – Humillar – Ridiculizar – Desacreditar – Discriminar
Amenaza	Se refiere a la acción de expresar o hacer algo que sugiere la posibilidad de causar daño o peligro a alguien o algo. El objetivo de la amenaza es infundir miedo o intimidación en la otra persona para lograr algún resultado deseado.	– Frases que infunden miedo. – Ataque a la vulnerabilidad de la persona.	– Intimidar – Controlar
Insulto	Los insultos son utilizados de manera intencional para causar daño emocional a otra persona. Puede ser una palabra, frase, comentario o gesto que se utiliza para atacar a alguien de manera ofensiva e hiriente.	– Apodos – Palabras anti sonantes – Palabras despectivas	– Ofender – Menospreciar – Herir – Humillar – Discriminar – Excluir

Tabla 2. Ítems clasificados por dimensión.

Dimensión	Ítems
Burla	En las redes sociales, por medio de los chats, en comentarios, publicaciones… – ¿Qué frases o palabras has identificado, con el propósito de discriminar a una persona? – ¿Qué chistes has identificado cuyo objetivo sea ridiculizar a una persona? – Si has observado alguna ironía que tenga por objetivo humillar a una persona, descríbela.
Amenaza	En las redes sociales, por medio de los chats, en comentarios, publicaciones… – ¿Con que frases o palabras has identificado que una persona amenaza a otra? – ¿Qué frase considerarías una amenaza? – ¿Qué frase considerarías una amenaza con la finalidad de controlar a otra persona?
Insulto	En las redes sociales, por medio de los chats, en comentarios, publicaciones… – ¿Cuál es el insulto más común que hayas visto? – ¿Cuál es el insulto más ofensivo que hayas visto? – ¿Qué groserías has identificado? – ¿Qué insultos, a través de abreviaturas has visto? Y ¿Cuál es su significado? – ¿Qué apodos ofensivos has visto?

Ya delimitadas las manifestaciones de ciberacoso a abordar, se formularon una serie de preguntas enfocadas a cada una de las dimensiones, teniendo un total de 11 ítems (tabla 2) creadas con el propósito de formar un cuestionario.

Objetivo del cuestionario: Recolectar palabras y frases con la finalidad de insultar, burlarse o amenazar a través de las redes sociales por parte de jóvenes universitarios.

Palabra	Total	Largo	1. Dimensión burla	%	2. Dimensión amenaza	%	3. Dimensión insulto	%	Total %
pendejo	179	7	8	0.21%	3	0.07%	168	4.16%	1.96%
puta	130	4	0	0.00%	3	0.07%	127	3.15%	1.42%
madre	127	5	0	0.00%	19	0.47%	108	2.68%	1.39%
negro	110	5	64	1.66%	1	0.03%	45	1.11%	1.20%
verga	99	5	1	0.03%	5	0.12%	93	2.30%	1.08%
hijo	82	4	2	0.05%	2	0.05%	78	1.93%	0.90%
puto	80	4	10	0.26%	0	0.00%	70	1.73%	0.88%
gordo	65	5	29	0.75%	0	0.00%	36	0.89%	0.71%
idiota	61	6	6	0.16%	1	0.03%	54	1.34%	0.67%
persona	54	7	30	0.78%	11	0.27%	13	0.32%	0.59%
chinga	53	6	0	0.00%	1	0.03%	52	1.29%	0.58%
color	51	5	45	1.17%	0	0.00%	6	0.15%	0.56%
ctm	45	3	0	0.00%	0	0.00%	45	1.11%	0.49%
físico	36	6	19	0.49%	1	0.03%	16	0.40%	0.39%
mierda	36	6	1	0.03%	1	0.03%	34	0.84%	0.39%
más	35	3	8	0.21%	14	0.34%	13	0.32%	0.38%
ninguno	35	7	8	0.21%	0	0.00%	27	0.67%	0.38%
gorda	33	5	15	0.39%	0	0.00%	18	0.45%	0.36%

Fig. 3. Previsualización de la bolsa de palabras.

Posteriormente se añadieron las preguntas a un cuestionario en Google Forms (figura 2) con la finalidad de que su distribución fuera más accesible y se distribuyó el enlace con jóvenes universitarios de la licenciatura en Administración de la Universidad Autónoma del Estado de México y de estudiantes de las ingenierías de Sistemas Computacionales, Gestión Empresarial y Química del Instituto Tecnológico de Orizaba. Enlace al cuestionario: https://forms.gle/t8sXH1pHbP4wHkDX9.

La recopilación de la información fue autoadministrada y se llevó a cabo en un periodo de aproximadamente dos semanas; gracias a la herramienta de Google Forms, las respuestas se almacenaron en hojas de cálculo de Excel y se obtuvo una muestra total de 213 jóvenes universitarios que respondieron a cada uno de los cuestionamientos planteados (figura 3). Con el propósito de incrementar la bolsa de palabras, se pretende obtener actualizaciones contemplando un incremento de participantes; por consiguiente, se agregarán al análisis las nuevas entradas de respuestas al cuestionario.

Las respuestas obtenidas fueron tratadas a través del software Atlas.ti [20] realizando un análisis cualitativo para determinar la repetitividad de las palabras y contemplar las palabras más comunes en el lenguaje verbal violento empleado en el grupo poblacional comprendido por jóvenes universitarios.

De igual forma se analizó la repetitividad de las palabras en cada una de las dimensiones comprendidas y posteriormente se realizó la depuración de aquellas palabras que no se relacionaban directamente con el propósito inicial como, por ejemplo, conectores y artículos. Con la finalidad de formar la bolsa de palabras a utilizar para la detección de ciberacoso (figura 4).

De igual forma se realizó el análisis por cada dimensión abordada, obteniendo una lista final por cada una de estas.

– Dimensión Burla: Se obtuvo una mayor cantidad de palabras, ya que en esta categoría se obtuvieron frases más largas por parte de los encuestados; finalmente la lista de palabras contemplo un total de 931 palabras.

Fig. 4. Vista de carga, inicio de sesión para la aplicación y menú principal.

Fig. 5. Menú principal del módulo, interfaz de grupo y envío de solicitud.

– Dimensión Amenaza: Filtrando y depurando conectores y artículos, se obtuvo un total de 522 palabras.
– Dimensión Insultos: En esta categoría se contemplan 519 palabras.

Cabe señalar que en el conteo se contemplaron palabras en singular, plural y sustantivos o adjetivos tanto en masculino como en femenino, ya que son las diferentes maneras en las que se pueden presentar en un contexto determinado.

5. Mockups

A continuación, se presentan los mockups utilizados para representar cómo se verá el diseño final del módulo de la aplicación en su contexto real. Cabe señalar que se continuará trabajando con el diseño de la aplicación base, por lo que se utiliza la misma paleta de colores.

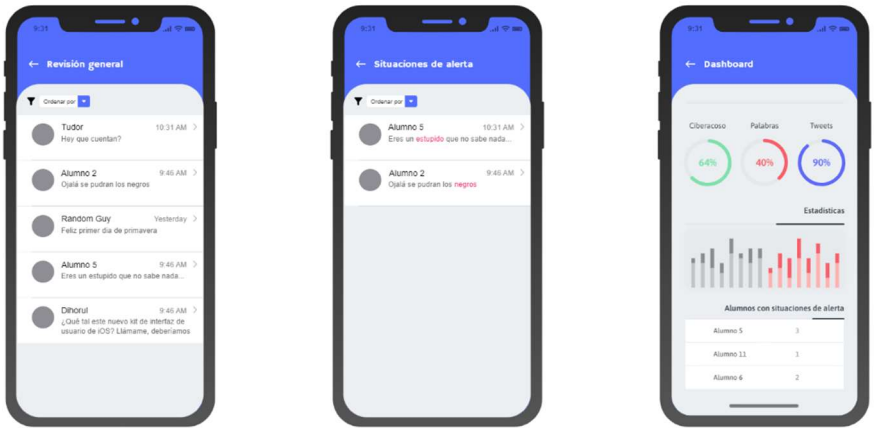


Fig. 6. Interfaz de la revisión general, situaciones de alerta y *dashboard*.

Interfaz de carga de aplicación (*Splash Screen*) que muestra la leyenda identificativa del proyecto "Monitorización de bullying y ciberbullying". Por otro lado, la interfaz de inicio de sesión muestra la solicitud de los campos necesarios para ingresar a la aplicación (correo electrónico y contraseña). Una vez que se autentifique el usuario, se muestra la pantalla principal de la aplicación con las herramientas utilizadas para la sección comprendida por el bullying.

En la parte inferior se muestran cuatro opciones, contemplando la opción de ciberacoso, cuya navegación se dirige al módulo del proyecto (figura 5). En la interfaz del menú principal del módulo sobre ciberacoso, se muestran las herramientas principales acompañadas de la bienvenida al usuario; estas herramientas, son: Grupo, Revisión general, Situaciones de alerta y *Dashboard*.

En la sección de grupo la interfaz presentada, mostrará la lista de los alumnos del grupo, con la opción de añadir su perfil de la red social (*Twitter*), la cual mostrará una ventana flotante con el campo de entrada de texto para colocar el *username* del alumno y el botón de enviar solicitud para que el alumno acepte los permisos necesarios (figura 6).

En la sección de "Revisión general" se mostrarán todas las publicaciones o comentarios que han realizado los alumnos, con la posibilidad de ordenarlas por más recientes o más antiguas comprendidas en un lapso de tiempo. En la interfaz de "Situaciones de alerta", se mostrarán aquellas publicaciones que detecten eventos de violencia verbal y que permita llevar un seguimiento para determinar un posible caso de ciberacoso. Por último, se presenta el *Dashboard* con algunos datos significativos (figura 7).

Es importante hacer mención de que pueden existir nuevas interfaces de acuerdo a las necesidades que se presenten a lo largo del desarrollo del proyecto; además se contempla añadir reportes generados por la aplicación. Las interfaces que si fueron mostradas se consideran las más importantes a incluir para el desarrollo del módulo de la aplicación.

6. Discusión

La arquitectura propuesta para el desarrollo del módulo sobre detección de ciberacoso describe cómo los diferentes componentes se relacionan entre sí y cómo se comunican entre sí para lograr los objetivos del sistema, la arquitectura de software es importante porque ayuda a planificar y organizar el desarrollo del software de manera efectiva y eficiente.

Por otro lado, la clasificación de las manifestaciones comprendidas por el ciberacoso, permitió identificar la delimitación de este proyecto, contemplando las manifestaciones de violencia verbal a través de textos; además, permitió la construcción del instrumento de recolección de las palabras o frases empleadas por los adolescentes para expresarse de forma agresiva u ofensiva en las redes sociales o medios digitales.

Los resultados obtenidos del análisis cualitativo aplicado a las respuestas correspondientes a la muestra a través de Atlas.ti, ayudó en la determinación de la bolsa o lista de palabras a emplear para identificar o detectar posibles situaciones de ciberacoso por medio del procesamiento del lenguaje natural. De igual forma, los mockups presentados brindan una apreciación previa de la distribución de elementos, diseños e interfaces con los que contará el módulo de la aplicación, respetando el diseño de la aplicación móvil base.

7. Conclusiones y trabajo a futuro

A pesar que el fenómeno de cyberbullying no es algo nuevo, tiene un aumento considerable en los últimos años, ya que la pandemia de COVID-19 iniciada en 2020 provoco que los adolescentes se encontraran más presentes en las redes sociales y a su vez más expuestos a sufrir algún tipo de acoso.

En el análisis presentado se hace notar que actualmente existen aplicaciones que brindan información sobre la prevención del ciberacoso, dando consejos o recomendaciones que tienen como objetivo prevenir este tipo de acoso entre los jóvenes; asimismo, existen aplicaciones que detecten casos de ciberacoso pero dichas aplicaciones se encuentran destinadas a usarse en otros países, lo que conlleva que el lenguaje sea diferente al español, cambiando también las palabras o términos que se suelen usar para ofender o humillar a una persona.

Debido a esto y con el objetivo de disminuir el ciberacoso, en este artículo se presentó la arquitectura para desarrollar un módulo implementado en una aplicación móvil base, que mediante el procesamiento de lenguaje natural identifique ocurrencias de ciberacoso; de igual forma, se abordó la obtención de la bolsa de palabras que se empleará para el algoritmo que permita detectar posibles situaciones de este fenómeno ya abordado, así como los mockups principales que se involucran en las vistas del sistema abarcado por el módulo a desarrollar.

Como trabajo a futuro se contempla realizar la conexión con la API de la red social Twitter a través del protocolo oAuth con las credenciales correspondientes, además de integrar algoritmos que sean capaces de detectar posibles casos de ciberbullying, lo que consecuentemente dirige al desarrollo el módulo de la aplicación.

Agradecimientos. Se agradece al Tecnológico Nacional de México por el apoyo otorgado, mencionando al Instituto Tecnológico de Orizaba por ser el anfitrión del desarrollo de este proyecto. Este proyecto cuenta con el apoyo del Consejo Nacional de Ciencia y Tecnología (CONACyT).

Referencias

1. Chun, J. S., Lee, J., Kim, J., Lee, S.: An international systematic review of cyberbullying measurements. Computers in Human Behavior, vol. 113 (2020) doi: 10.1016/j.chb.20 20.106485
2. Hinduja, S., Patchin, J. W.: Cyberbullying: An exploratory analysis of factors related to offending and victimization. Deviant Behavior, vol. 29, no. 2, pp. 129–156 (2008) doi: 10.1080/01639620701457816
3. INEGI: Módulo sobre ciberacoso. Comunicado de prensa, no. 364 (2022)
4. Weichbroth, P.: Usability of mobile applications: A systematic literature study. IEEE Access, vol. 8, pp. 55563–55577 (2020) doi: 10.1109/ACCESS.2020.2981892
5. Ho, S. M., Kao, D., Chiu-Huang, M. J., Li, W., Lai, C. J.: Detecting cyberbullying "Hotspots" on Twitter: A predictive analytics approach. Forensic Science International: Digital Investigation, vol. 32 (2020) doi: 10.1016/j.fsidi.2020.300906
6. Upadhyay, A., Chaudhari, A., Arunesh, Ghale, S., Pawar, S.: Detection and prevention measures for cyberbullying and online grooming. In: Proceedings of the International Conference on Inventive Systems and Control, ICISC, pp. 1–4 (2017) doi: 10.1109/ICISC. 2017.8068605
7. Silva, Y. N., Hall, D. L., Rich, C.: BullyBlocker: Toward an interdisciplinary approach to identify cyberbullying. Social Network Analysis and Mining, vol. 8, no. 18, pp. 1–15 (2018) doi: 10.1007/s13278-018-0496-z
8. Salawu, S., He, Y., Lumsden, J.: BullStop: A mobile app for cyberbullying prevention. In: Proceedings of the 28th International Conference on Computational Linguistics: System Demonstrations, 70–74 (2021). doi: 10.18653/v1/2020.coling-demos.13
9. Neo, H. F., Teo, C. C., Han-Boon, J. L.: Mobile edutainment learning approach: #stopbully. In: ICDTE: Proceeding of ten 2nd International Conference on Digital Technology in Education, pp. 6–10 (2018) doi: 10.1145/3284497.3284500
10. Foong, Y. J., Oussalah, M.: Cyberbullying system detection and analysis. In: Proceedings - 2017 European Intelligence and Security Informatics Conference, EISIC, pp. 40–46 (2017) doi: 10.1109/EISIC.2017.43
11. Thun, L. J., Teh, P. L., Cheng, C. Bin: CyberAid: Are your children safe from cyberbullying? Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 7, pp. 4099–4108 (2021) doi: 10.1016/j.jksuci.2021.03.001
12. Garaigordobil, M.: Conducta antisocial: Conexión con bullying/cyberbullying y estrategias de resolución de conflictos. Psychosocial Intervention, vol. 26, no. 1, pp. 47–54 (2017) doi: 10.1016/j.psi.2015.12.002
13. Hang, O. C., Dahlan, H. M.: Cyberbullying lexicon for social media. In: International Conference on Research and Innovation in Information Systems, ICRIIS. pp. 1–6 (2019) doi: 10.1109/ICRIIS48246.2019.9073679
14. Noviantho., Isa, S. M., Ashianti, L.: Cyberbullying classification using text mining. In: Proceedings - 2017 1st International Conference on Informatics and Computational Sciences, ICICoS, pp. pp. 241–245 (2017)
15. Farag, N., McKee, G., El-Seoud, S. A., Hassan, G.: Bullying hurts: A survey on non-supervised techniques for cyber-bullying detection. ACM International Conference Proceeding Series, pp. 85–90 (2019) doi: 10.1145/3328833.3328869

16. Microsoft: Información general sobre ASP.NET MVC https://docs.microsoft.com/es-es/aspnet/mvc/overview/older-versions-1/overview/asp-net-mvc-overview
17. Twitter Inc.: Developer Platform, https://developer.twitter.com/en/docs/twitter-api
18. Sharan, K.: Model-view-controller pattern. Learn JavaFX, vol. 8, pp. 419–434 (2015)
19. Bertocco, M., Ferraris, F., Offelli, C., Parvis, M.: A client–server architecture for distributed measurement systems. IEEE Transaction on Instrumentation and Measurement, vol. 47, no. 5, pp. 1143–1148 (1998)
20. ATLAS.ti: ATLAS.ti, https://atlasti.com/es

ISSN 1870-4069

Diferenciación de la región macular dentro de la retina utilizando operaciones morfológicas simples orientado al prediagnóstico en etapas tempranas de retinopatía diabética

Emanuel de-la-Cruz-Espinosa, Rita Q. Fuentes-Aguilar, Eduardo Morales-Vargas

Tecnológico de Monterrey,
Instituto de Materiales Avanzados para la Manufactura Sostenible,
Campus Guadalajara,
México

{a01150821, rita.fuentes, emoralesv}@tec.mx

Resumen. La diabetes es una enfermedad con presencia global y una tasa de mortalidad alta, causando un gran impacto socioeconómico. Uno de los efectos negativos más significativos de la diabetes es la ceguera permanente causada por la retinopatía diabética. Los métodos actuales para identificar pacientes que necesitan ser atendidos por un especialista, y así prevenir la pérdida visual, son el examen de detección de la retinopatía diabética y la tomografía de coherencia óptica. Sin embargo, el número de oftalmólogos y equipos para realizar tomografías de coherencia óptica no son suficientes para atender a toda la población que padece diabetes. Por esta razón, los esfuerzos de investigación se enfocan en disminuir el tiempo invertido en el examen de detección de la retinopatía diabética usando métodos computacionales. Existe un gran interés en el análisis de la región de la mácula porque pueden encontrarse signos de daños por la retinopatía diabética en etapas tempranas, debido a la alta concentración de células encargadas de la agudeza visual. Por lo tanto, este trabajo se centra en el desarrollo de un algoritmo para la segmentación de la mácula ocular, usando técnicas simples de procesamiento de imagen para mantener un costo computacional bajo. Logrando obtener un procedimiento preliminar que permita realizar en trabajos futuros una examinación exhaustiva de la mácula, capaz de detectar anomalías relacionadas a la retinopatía diabética, y realizar una clasificación del grado de severidad de este padecimiento. El algoritmo que se desarrolló se validó comparando el centro estimado de la macula y el centro de la macula conocido, consiguiendo un error promedio de $1.22 \pm 0.99\,\%$ en una muestra que se tomó de la base de datos pública Kaggle.

Palabras clave: Retinopatía diabética, segmentación de la mácula, operaciones morfológicas.

Emanuel de-la-Cruz-Espinosa, Rita Q. Fuentes-Aguilar, Eduardo Morales-Vargas

Differentiating the Macula Region from the Retina Using Simple Morphological Operations Towards a Pre-Diagnosis in the Early Stages of Diabetic Retinopathy

Abstract. Nowadays, diabetes is a disease with a worldwide presence and a high mortality rate, causing a significant social and economic impact. One of the more significant adverse effects of diabetes is visual loss due to diabetic retinopathy. Current methods to identify patients who need to be seen by a specialist to prevent vision impairment are diabetic retinopathy screening and optic coherence tomography. However, the number of ophthalmologists and optic coherence tomography devices is insufficient to cover the diabetic population. Thus, recent research efforts focus on improving screening times using computational methods. An increasing interest is in the analysis of the macula region because early damage signs of diabetic retinopathy can be found in this area since there is a high concentration of cells in charge of visual acuity. Therefore, this work focuses on developing an algorithm for ocular macula segmentation using simple image processing techniques to maintain a low computational cost. Obtaining a preliminary procedure to enable a future insightful examination of the macula for the detection of anomalies related to diabetic retinopathy toward the classification of its stages. The algorithm was validated by measuring the distance between the segmented macula and the ground truth, achieving a mean error of 1.22 ± 0.99 % in a sample obtained from the Kaggle public database.

Keywords: Diabetic retinopathy, macula segmentation, morphological operations.

1. Introducción

La diabetes es una enfermedad con presencia global y una alta tasa de mortalidad. Alrededor de 422 millones de personas sufren de este padecimiento, y 1.6 millones de muertes son relacionadas directamente a la enfermedad cada año [18]. De las muertes reportadas en México en el año 2020, $14\,\%$ son a causa de la diabetes, siendo la tasa más grande de los últimos 10 años en el país [9].

Por esta razón, la diabetes tiene un gran impacto socioeconómico, puesto que su tratamiento tiene un periodo largo y está enfocado en controlar los síntomas de la enfermedad y no en erradicar la enfermedad perse. Uno de los efectos negativos más significativos de la diabetes es la ceguera permanente causada por Retinopatía Diabética (RD), $2.6\,\%$ de la ceguera en el mundo es a causa de ella [18], particularmente en adultos de 20 a 74 años de edad en países de mediano y alto desarrollo [5].

Por esta razón, es necesario detectar a tiempo a aquellos pacientes que necesitan ser atendidos por un especialista, comúnmente se hace a través de un estudio de Detección de Retinopatía Diabética (DRD) y complementando con una Tomografía de Coherencia Óptica (TCO) para proveer el tratamiento apropiado y evitar la pérdida de visión permanente [16, 13].

(a) (b) (c) (d) (e)

Fig. 1. Ejemplos de imágenes de cada clase en la base de datos. (a) No aparente RD, (b) Leve RDNP, (c) Moderado RDNP, (d) Severo RDNP y (e) RDP.

Desafortunadamente, el número de oftalmólogos y equipos de TCO no es suficiente para dar un sistema de salud adecuado a la población que padece diabetes. Además, el presupuesto en algunos países para realizar estudios rápidos no es suficiente para brindar atención especializada a aquellos pacientes que la necesitan porque demanda equipos de precio alto que además son operados solo por especialistas [2].

Existe un gran interés en estudiar la mácula ocular localizada en el área de la fóvea de la retina porque tiene la mayor concentración de células encargadas de la agudeza visual [1, 3]. Debido a la importancia de estudiar la macula ocular, algunos grupos de investigación se están enfocando en la detección de la presencia de los primeros signos de daño causados por RD.

Por esta razón hay un interés en desarrollar métodos de segmentación automática para localizar la mácula ocular en imágenes de fondo de ojo como un paso inicial al prediagnóstico de RD en un estado temprano.

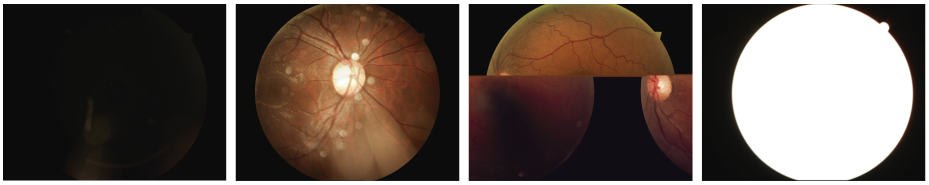
Trabajos actuales localizan la mácula ocular considerando la posición geométrica de las estructuras retinales. Un trabajo reciente aplica morfología matemática sobre el área temporal del Disco Óptico (DO) guiándose con la posición de los vasos sanguíneos en la imagen [17]. Pese que el tiempo promedio de procesamiento por imagen es de 0.64s, la distancia euclidiana entre el centro estimado de la mácula y el real es mayor en comparación a otros trabajos [14, 4]; por lo que aún hay áreas de mejora.

Por otro lado, otros trabajos utilizan técnicas de procesamiento de imagen con una complejidad más alta, como el crecimiento de regiones o el cálculo de mapas de prominencia, pero el tiempo promedio de procesamiento por imagen supera los 20s, dificultando su uso práctico en aplicaciones en tiempo real [4, 14].

Dado que los métodos actuales logran reducir el error, aumentando el tiempo promedio de procesamiento; este trabajo propone una metodología par atacar este problema manteniendo una complejidad baja mientras se reduce la distancia entre el centro de la mácula estimado y el real.

Se presenta un algoritmo con operaciones morfológicas simples para detectar el centro de la región macular en imágenes de fondo de ojo como Region De Interés (RDI), teniendo en cuenta factores externos en la adquisición de imágenes, como la iluminación, el enfoque, y el ruido.

Es importante validar las imágenes y seleccionar aquellas que cuentan con las características adecuadas para ser analizadas, y evitar aquellas con condiciones como corrimiento de pantalla, sobre exposición, exposición baja, y reflejo; debido a que complican el análisis para el médico y para el método computacional.



(a) (b) (c) (d)

Fig. 2. Imágenes descartadas de la base de datos Kaggle después de realizar el paso de validación. Las imágenes fueron descartadas debido a las condiciones de adquisición. (a) Imagen con exposición baja, (b) Imagen con DO en el centro de la retina, (c) Imagen con corrimiento de pantalla, y (d) Imagen con sobre exposición.

Después de ser validadas, las imágenes son tratadas con escalamiento, realce, binarización, y operaciones morfológicas para encontrar el contorno de la mácula y consecuentemente su centroide, usando operaciones simples, ya que en la mayoría de los casos, las computadoras de los consultorios médicos no cuentan con suficiente poder computacional para realizar una rápida y precisa segmentación a través de modelos computacionales complejos.

La segmentación permite a los médicos realizar revisiones profundas en búsqueda de anomalías relacionadas a RD, como los son los micro-aneurismas, exudados, hemorragias, y manchas algodonosas, teniendo como propósito canalizar a los pacientes de manera oportuna [1, 3].

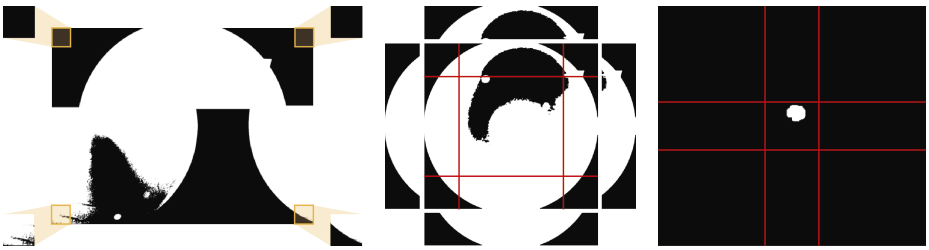
2. Materiales y métodos

Esta sección describe brevemente los conceptos requeridos para entender el desarrollo de la metodología. Hay varias técnicas de procesamiento de imágenes, entre ellas se incluyen técnicas para mejorar la representación de características, como aquellas que aumentan la resolución espacial de una imagen y reducen el ruido, adicionalmente a técnicas de morfología matemática.

2.1. Morfología matemática

La Morfología Matemática (MM) aplica las disciplinas de teoría de conjuntos, geometría, y álgebra de patrones para definir técnicas de procesamiento de imagen con la finalidad de transformar imágenes binarizadas o de escala a grises con base en la forma de objetos geométricos. Las técnicas de MM pueden filtrar y segmentar mediante operaciones básicas como unión, intersección y complemento, comparando la imagen procesada con un Elemento Estructural (EE) [7, 15].

El EE es una matriz con valores 0 ó 1 que determina cuáles píxeles son considerados y cuáles no en el procesamiento de una imagen. En la mayoría del tiempo, el EE se determina utilizando el conocimiento previo sobre los objetos en la imagen que se analiza. Las operaciones principales en MM son la erosión y la dilatación, que se representan como $I \ominus S$ y $I \oplus S$, respectivamente.



(a) (b) (c)

Fig. 3. Análisis de valor promedio de los píxeles en imágenes de fondo de ojo binarizadas descartadas de la base de datos kaggle. (a) Análisis en las esquinas de una imagen con corrimiento de pantalla, (b) Análisis en la orillas de una imagen con reflejo, y (c) Análisis en el centro de una imagen con DO centrado en la retina.

La erosión reduce I trasladando S_x sobre I y preservando solo aquellos píxeles que pertenecen a I y al elemento estructural S_x trasladado cuando sus orígenes convergen. Por el otro lado, la dilatación agrega píxeles a I donde S_x converge en un punto previamente definido con la imagen I [11].

2.2. Umbralización

La umbralización consiste en asignar una etiqueta a cada píxel de la imagen I cumpliendo una condición, la cual puede ser un umbral o nivel de similitud que sirve para agrupar áreas con valores de grises constantes.

Por ejemplo, la umbralización puede segmentar una imagen en dos o más secciones, principalmente en la RDI y el fondo. En el caso de la binarización, se establece un valor de 1 en la RDI y un valor de 0 en el fondo para cada píxel en la imagen, definido en Eq. 1 [11]. Una desventaja de la binarización radica en seleccionar un valor correcto del umbral u para delimitar correctamente la RDI del fondo.

El método de Otsu [12] es un algoritmo comúnmente utilizado para la estimación del umbral u mediante el análisis estadístico de la varianza entre clases de la imagen, asumiendo una distribución bimodal con dos regiones en la imagen, la RDI, y el fondo, buscando la máxima divisibilidad de las clases:

$$I_{bn}(x,y) = \begin{cases} 1, \text{ si } (x,y) \geq u, \\ 0, \text{ en caso contrario.} \end{cases} \tag{1}$$

2.3. Ecualización adaptativa de histograma limitada por el contraste

El objetivo de la Ecualización Adaptativa de Histograma Limitada por el Contraste (CLAHE) es mejorar el contraste entre las regiones en una imagen sin incrementar el ruido [19]. A diferencia de la Ecualización Adaptativa de Histograma (AHE), el arreglo de la distribución del nivel de grises se restringe por una cantidad limitada de píxeles establecida en cada sección relacionado a histogramas locales.

(a) (b) (c)

Fig. 4. Imagen de fondo de ojo con no aparente RD de la base de datos Kaggle. El acercamiento es en el área de la fóvea. Canales (a) Verde, (b) Azul, y (c) Rojo.

Por otro lado, la ecualización del histograma consiste en redistribuir los valores en una imagen de acuerdo a su histograma mediante un escalamiento y mapeo utilizando una función de distribución acumulada definida en la Eq. 2, tal que $P(X \leq x)$ es la probabilidad de que una variable aleatoria X sea menor o igual que un valor dado x:

$$F(x) = P(X \leq x). \tag{2}$$

3. Experimentos y resultados

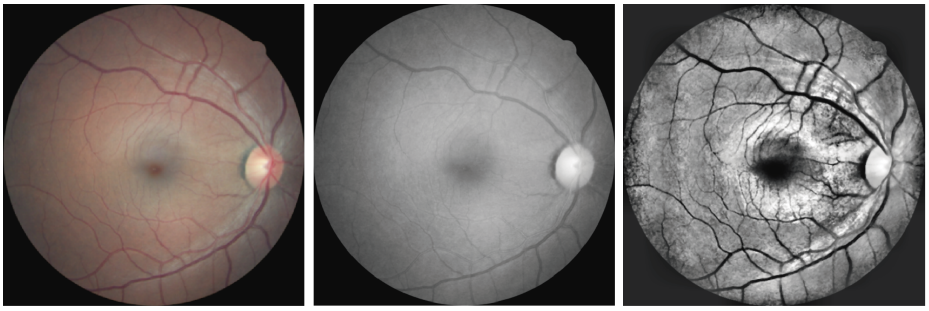
Esta sección presenta los experimentos y resultados obtenidos para la localización del centroide de la mácula ocular. La base de datos y su validación se presentan en la primer subsección, seguida por la explicación de la técnica de procesamiento de imagen junto con su evaluación.

3.1. Base de datos

Actualmente, a través de los esfuerzos de la investigación, múltiples bases de datos públicas de imágenes de fondo de ojo proveen información para diseñar, implementar y evaluar métodos de procesamiento de imagen, con el objetivo de generar modelos computacionales que puedan extraer características difíciles de visualizar para el ojo humano con la finalidad de brindar soporte en diagnósticos médicos.

Se propone utilizar la base de datos pública Kaggle extraída de Eye Picture Archive Communication System (EyePACS) para validar la metodología de procesamiento de imágenes. El principal factor decisivo fue la diversidad de factores y la cantidad de imágenes que provee [6]. EyePACS es la base de datos pública más grande para la detección de RD [10]. Se alimenta por un sistema en línea de DRD con licencia gratuita para facilitar la adquisición de imágenes, su distribución, y examinación [6].

La base de datos está compuesta por $35,126$ imágenes de fondo de ojo de alta resolución que tienen presente la mácula en la retina. Etiquetadas con un número de identificación del sujeto de prueba y la orientación, izquierdo o derecho. Las imágenes fueron adquiridas en diferentes condiciones y con diversas cámaras como Centervue DRD, Optovue iCam, Canon CR-1/DGi/CR-2, y Topcon NW usando $45°$ de campo de visión. Alrededor del $40\,\%$ de las imágenes fueron midriáticas con un promedio de edad de los pacientes de 54.4 años, de los cuales el $42.6\,\%$ son mujeres [8].



(a) (b) (c)

Fig. 5. Mejora de visualización de una imagen de fondo de ojo de clase 0 - no aparente RD. (a) Imagen inicial, (b) Canal verde de (a) después del proceso de eliminación de ruido, y (c) Imagen mejorada con CLAHE.

Cada fotografía se clasificó en una escala de 0 a 4 de acuerdo a la presencia de RD de la clasificación internacional de RD con el apoyo de especialistas y un algoritmo: 0 - No aparente RD (Fig. 1a), 1 - Leve RDNP (Fig. 1b), 2 - Moderado RDNP (Fig. 1c), 3 - Severo RDNP (Fig. 1d), 4 - RDP (Fig. 1e).

El Edema Macular Diabético (EMD) es clasificado con base en lesiones de Exudados Duros (ED), que son consistentemente relacionados con el engrosamiento adyacente de retina y predice la presencia de EMD porque el protocolo de DRD de EyePACS no usa imágenes estereoscópicas, por lo que no es posible evaluar engrosamiento de la retina en ellas [6]. Esta base de datos no es balanceada porque $73.5\,\%$ $(25,810)$ de las imágenes son clasificadas como 0 - No aparente RD, $7\,\%$ $(2,443)$ como 1 - Leve RDNP, $15.1\,\%$ $(5,292)$ como 2 - Moderado RDNP, $2.5\,\%$ (873) como 3 - Severo RDNP y $2\,\%$ (708) como 4 - RDP.

La base de datos de Kaggle podría contener imágenes con baja calidad o con problemas de adquisición que necesitan ser examinados. Tal como baja exposición, sobre exposición, e imágenes con el DO centrado en la retina o con corrimiento de pantalla (Fig. 3). Las imágenes con exposición baja (Fig. 2a) son un problema porque no es posible delimitar el área de la retina del fondo. Por el contrario, las imágenes con sobre exposición (Fig. 2d) no hacen posible la distinción de características después de la binarización.

Por otro lado, hay otros problemas relacionados a la localización de la mácula, por ejemplo, las imágenes con el DO en el centro de la retina, ya que provocan que el área de la fóvea se encuentre en las orillas de la retina, lo cual es un problema porque hay una pérdida parcial de información de la mácula ocular. Considerando las condiciones de las imágenes se propone realizar un proceso de validación para preservar solo aquellas que se puedan analizar por un método computacional en condiciones normales, descartando aquellas imágenes con problemas de baja calidad.

El primer paso del proceso de validación consiste en calcular el valor promedio de los píxeles de la imagen Rojo Verde Azul (RVA), y aquellas que cuenten con un valor por debajo de 15 se descartan dado que se consideran imágenes con baja exposición. Después, los canales de color se dividen para tomar solo el canal rojo y se escale para reducir el tiempo de procesamiento.

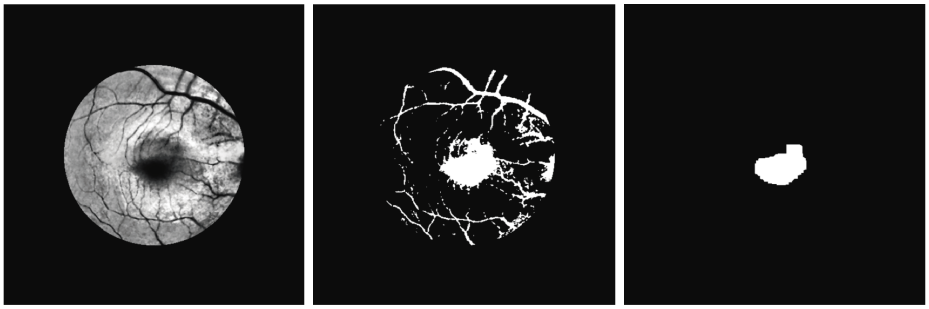


Fig. 6. Segmentación de una imagen de fondo de ojo después del proceso de mejora de visualización. (a) Imagen con máscara circular al centro, (b) Binarización invertida de (a), y (c) Operación de apertura en (b).

La altura de la imagen escalada es de 512 píxeles y la anchura es calculada con la Eq. 3. Donde W_n es el nuevo valor de anchura, y la altura y anchura originales son H_o y W_o respectivamente. El canal rojo se eligió después de una comparación visual de varias imágenes, ya que la retina se distingue mejor del fondo en este canal, comparado con el verde y azul:

$$W_n = \frac{512}{H_o} \times W_o. \tag{3}$$

El último paso de la validación consiste en la binarización de la imagen para analizar las esquinas (Fig. 3a), las orillas (Fig. 3b), y el centro (Fig. 3c). Este proceso descarta aquellas imágenes con corrimiento de pantalla, uno de los problemas explicados al principio de la sección e ilustrado en Fig. 2c.

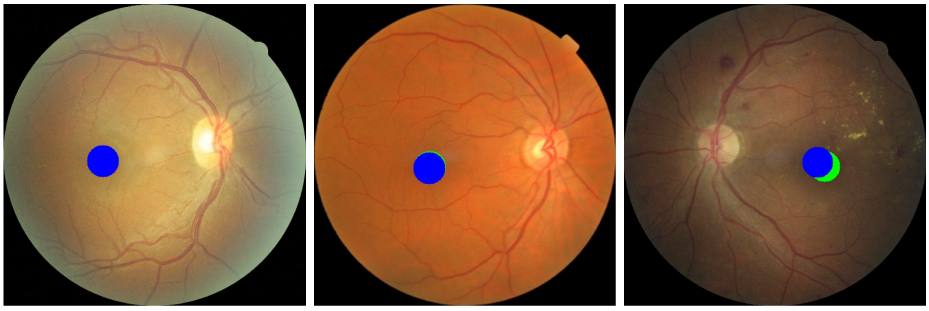
3.2. Localización de la mácula

El método propuesto para la localización de la mácula se compone de tres etapas: i) preprocesamiento de la imagen para aumentar el contraste de las regiones del fondo del ojo. Este paso es importante dado que permite a las siguientes etapas identificar de una manera más precisa el área de la fóvea, ii) segmentación del área de la fóvea y eliminación de vasos sanguíneos de la retina para facilitar la detección de la macula, y iii) localización y reconstrucción.

Esta etapa final encuentra los contornos en la imagen segmentada y calcula el centroide de la mácula. Las imágenes utilizadas para validar la metodología son aquellas seleccionadas de la base de datos de Kaggle después del proceso de validación que descarta aquellas imágenes con baja calidad.

La localización se evaluó midiendo la distancia euclidiana en píxeles entre los centroides como se describe en la Eq. 4, y calculado el porcentaje de error relativo considerando las dimensiones de la imagen como se define en la Eq. 5, donde C_{xl}, C_{yl} son las coordenadas del centroide de la mácula, estimadas visualmente y C_{xp}, C_{yp} son las coordenadas del centroide de la mácula estimadas con nuestra metodología. W_I y H_I son las dimensiones de la imagen:

$$\text{Distance} = \sqrt{(C_{xl} - C_{xp})^2 + (C_{yl} - C_{yp})^2}, \tag{4}$$



(a) (b) (c)

Fig. 7. Representación visual en color azul de la predicción del centroide de la mácula estimada con el método propuesto, y la representación visual de la etiqueta verdadera en color verde en imágenes de fondo de ojo de la base de datos Kaggle. (a) Imagen de fondo de ojo de clase 1 - leve RDNP, (b) Imagen de fondo de ojo de clase 0 - no aparente RDNP, y (c) Imagen de fondo de ojo clase 3 - severo RDNP.

$$\text{Error} = \sqrt{\frac{(C_{xl} - C_{xp})^2}{(W_I)^2} + \frac{(C_{yl} - C_{yp})^2}{(H_I)^2}} \times 100. \tag{5}$$

El método propuesto comienza con una etapa de preprocesamiento para aumentar el contraste del área de la fóvea con los otros elementos en la imagen (Fig. 5). Este proceso se realiza para distinguir la mácula ocular y los vasos sanguíneos de la retina. Primero se realiza un suavizado con un filtro que considera el valor de la mediana en una ventana de tamaño 3×3 sobre el canal verde (Fig. 5b) con la finalidad de disminuir el ruido en la imagen.

Después se aumenta el contraste entre la retina y la mácula aplicando una CLAHE con un límite de mejora de contraste de 40 y un número de mosaicos de 8 (Fig. 5c). El procesamiento se realizó en el canal verde porque en él se aprecia un mayor contraste en la región de interés (ver Fig. 4a). La imagen después del preprocesamiento se muestra en la Fig. 5c.

Después el método continuó con la localización y reconstrucción del centroide de la mácula. Esta etapa se realiza mediante una umbralización invertida (Eq. 6) utilizando Otsu (Fig. 6b) para estimar el umbral u seguido de una apertura morfológica con un elemento estructural de forma cuadrada con tamaño 5×5. De esta manera es posible diferenciar el área de la fóvea y eliminar los vasos sanguíneos. El resultado se ilustra en la Fig. 6c:

$$I_{bn}(x, y) = \begin{cases} 1, \text{ si } (x, y) \leq u, \\ 0, \text{ en caso contrario .} \end{cases} \tag{6}$$

Finalmente, como última etapa, el centroide de la mácula se localiza analizando los contornos y calculando los momentos del polígono. Un círculo azul se dibuja usando las coordenadas del centroide sobre la imagen RVA haciendo posible comparar visualmente los resultados conseguidos. La representación visual parece indicar que es posible detectar con precisión el centroide en la clase 0.

Tabla 1. Error entre la predicción de la localización del centroide de la mácula y la etiqueta verdadera. Distancia en píxeles y porcentaje de error relativo al tamaño de la imagen.

Clase	Distancia (px)	Error (%)
0 - No aparente RD	4.78 ± 3.10	0.90 ± 0.60
1 - Leve RDNP	6.16 ± 5.41	1.12 ± 1.02
2 - Moderado RDNP	5.26 ± 3.15	0.98 ± 0.60
3 - Severo RDNP	8.47 ± 7.16	1.55 ± 1.32
4 - RDP	8.23 ± 5.85	1.53 ± 1.07
Promedio	6.58 ± 5.34	1.22 ± 0.99

En cambio, las lesiones grandes relacionadas a la RD presentes en la retina, hacen más difícil para el método propuesto la localización del centroide porque llegan a sobre ponerse en la mácula. La salida del método propuesto es una imagen de fondo de ojo a color con la representación de la predicción de la localización del centroide de la mácula y la representación de la etiqueta verdadera.

Para propósitos de visualización, la etiqueta verdadera y la predicción del centroide de la mácula se muestran en la Fig. 7. Se realizó una comparación del porcentaje de error en píxeles respecto al tamaño de la imagen para respaldar las conclusiones. Los resultados se muestran en la Tabla. 1. El análisis se realizó para cada una de las clases de nivel de RD calculando el valor promedio y la desviación estándar. Los resultados sugieren que es posible localizar el centroide de la mácula con un error de $0.90\,\%$ a $1.53\,\%$. En la clase 0 se obtuvo el menor error el cual aumenta conforme el nivel de RD aumenta. Se acumula evidencia de que el error del método propuesto aumenta para la clase 3 - Severo RDNP y 4 - RDP.

Adicionalmente, se realizaron experimentos con la base de datos Messidor [**?**] para poder comparar el desempeño de la metodología propuesta con el de otros trabajos. Se calculó la media del error entre los centros de la macula estimados y los verdaderos con el algoritmo propuesto, y se comparan los resultados con aquellos reportados que obtuvieron menores tiempos promedio de procesamiento por imagen. Se presentan los resultados de la media sin dividir los datos por nivel de severidad debido a que el trabajo con el que se realiza la comparación sólo reporta la media general.

El método propuesto obtuvo un valor promedio de 6.15 píxeles de distancia euclidiana entre el centroide de la predicción y el centroide verdadero, y un tiempo promedio de procesamiento por imagen de 0.096 s. Estos resultados muestran que el método propuesto podría ser una opción a considerar si se requiere encontrar la macula ocular ya que los valores representan una mejora. Se logró disminuir 2.55 píxeles de distancia euclidiana promedio y 0.544 s de tiempo promedio por imagen comparados con 8.7 y 0.64 s respectivamente.

4. Conclusiones y trabajo a futuro

Este trabajo presentó una metodología para identificar la mácula en imágenes de fondo de ojo utilizando morfología matemática y procesamiento de imágenes de complejidad baja. El proceso comienza con un paso de validación para descartar todas aquellas que no cumplen con un criterio de calidad.

Las imágenes con corrimiento, bajo o alto tiempo de exposición y con el DO al centro de la imagen se descartaron. De manera general se puede concluir que es posible identificar el centro la mácula utilizando la metodología propuesta con un porcentaje promedio de error de $1.22 \pm 0.99\,\%$ con la base de datos Kaggle. Por otro lado, una de las desventajas del método se encuentra al analizar imágenes con lesiones oculares tales como hemorragias, especialmente en estados más avanzados de la enfermedad.

Aun así, el método permite una localización de la mácula con un porcentaje de error de hasta el $1.55 \pm 1.32\,\%$, haciendo posible una examinación más profunda por parte de los médicos en imágenes recortadas de la mácula con un tamaño que considere el área de análisis del protocolo de DRD de EyePACS, de 1 a 2 diámetros del DO para pre-diagnosticar los niveles de severidad de edema macular. Además, el método propuesto logra obtener mejores resultados que otros trabajos reduciendo el error y el tiempo promedio de procesamiento por imagen.

Como trabajo futuro sería importante poder encontrar la correlación entre la presencia de anomalías en la RDI y los estados de la enfermedad, ya que esta actividad es una tarea difícil para médicos generales y especialistas, sobre todo en etapas tempranas; por lo que se podria proponer la implementación de algoritmos de aprendizaje automático para la detección de RD en estados iniciales, permitiendo así referir de manera oportuna a los pacientes con la enfermedad con un costo menor comparado con las técnicas actuales como lo es la TCO.

Referencias

1. Abramoff, M. D., Garvin, M. K., Sonka, M.: Retinal imaging and image analysis. IEEE Reviews in Biomedical Engineering, vol. 3, pp. 169–208 (2010) doi: 10.1109/rbme.2010.2084567
2. Bilal, A., Sun, G., Mazhar, S.: Survey on recent developments in automatic detection of diabetic retinopathy. Journal Français d'Ophtalmologie, vol. 44, no. 3, pp. 420–440 (2021) doi: 10.1016/j.jfo.2020.08.009
3. Bird, A. C., Bok, D.: Why the macula? Eye, vol. 32, no. 5, pp. 858–862 (2017) doi: 10.1038/eye.2017.247
4. Chalakkal, R. J., Abdulla, W. H., Thulaseedharan, S. S.: Automatic detection and segmentation of optic disc and fovea in retinal images. IET Image Processing, vol. 12, no. 11, pp. 2100–2110 (2018) doi: 10.1049/iet-ipr.2018.5666
5. Cheung, N., Mitchell, P., Yin-Wong, T.: Diabetic retinopathy. The Lancet, vol. 376, no. 9735, pp. 124–136 (2010) doi: 10.1016/s0140-6736(09)62124-3
6. Cuadros, J., Bresnick, G.: EyePACS: An adaptable telemedicine system for diabetic retinopathy screening. Journal of Diabetes Science and Technology, vol. 3, no. 3, pp. 509–516 (2009) doi: 10.1177/193229680900300315
7. Ćurić, V., Landström, A., Thurley, M. J., Luengo Hendriks, C. L.: Adaptive mathematical morphology – a survey of the field. Pattern Recognition Letters, vol. 47, pp. 18–28 (2014) doi: 10.1016/j.patrec.2014.02.022
8. Gulshan, V., Peng, L., Coram, M., Stumpe, M. C., Wu, D., Narayanaswamy, A., Venugopalan, S., Widner, K., Madams, T., Cuadros, J., Kim, R., Raman, R., Nelson, P. C., Mega, J. L., Webster, D. R.: Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. Journal of the American Medical Association, vol. 316, no. 22, pp. 2402 (2016) doi: 10.1001/jama.2016.17216

9. INEGI: Estadísticas a propósito del día mundial de la diabetes. Instituto Nacional de Estadística y Geografía (2021) www.inegi.org.mx/app/saladeprensa/noticia.html?id=6923
10. Ishtiaq, U., Abdul-Kareem, S., Mohd-Faizal-Abdullah, E. R., Mujtaba, G., Jahangir, R., Yasir-Ghafoor, H.: Diabetic retinopathy detection through artificial intelligent techniques: A review and open issues. Multimedia Tools and Applications, vol. 79, no. 21-22, pp. 15209–15252 (2019) doi: 10.1007/s11042-018-7044-8
11. Morales-Vargas, E., Sosa-Martinez, J., Peregrina-Barreto, H., Rangel-Magdaleno, J., Ramirez-San-Juan, J.: A morphological approach for locating blood vessels in laser contrast speckle imaging. In: IEEE International Instrumentation and Measurement Technology Conference (I2MTC), pp. 1–6 (2018) doi: 10.1109/I2MTC.2018.8409778
12. Otsu, N.: A threshold selection method from gray-level histograms. IEEE Transactions on Systems, Man, and Cybernetics, vol. 9, no. 1, pp. 62–66 (1979) doi: 10.1109/TSMC.1979.4310076
13. Rebolleda, G., Diez-Alvarez, L., Casado, A., Sánchez-Sánchez, C., de Dompablo, E., González-López, J. J., Muñoz-Negrete, F. J.: OCT: New perspectives in neuro-ophthalmology. Saudi Journal of Ophthalmology, vol. 29, no. 1, pp. 9–25 (2015) doi: 10.1016/j.sjopt.2014.09.016
14. Romero-Oraá, R., García, M., Oraá-Pérez, J., López, M. I., Hornero, R.: A robust method for the automatic location of the optic disc and the fovea in fundus images. Computer Methods and Programs in Biomedicine, vol. 196, pp. 105599 (2020) doi: 10.1016/j.cmpb.2020.105599
15. Soille, P.: Morphological image analysis: Principles and applications, Springer Berlin Heidelberg (2013) books.google.com.mx/books?id=ZFzxCAAAQBAJ
16. Vujosevic, S., Aldington, S. J., Silva, P., Hernández, C., Scanlon, P., Peto, T., Simó, R.: Screening for diabetic retinopathy: New perspectives and challenges. The Lancet Diabetes and Endocrinology, vol. 8, no. 4, pp. 337–347 (2020) doi: 10.1016/S2213-8587(19)30411-5
17. Wibawa, H. A., Harjoko, A., Sumiharto, R., Sasongko, M. B.: Efficient and robust method to detect the location of macular center based on optimal temporal determination. Journal of Imaging, vol. 8, no. 12, pp. 313 (2022) doi: 10.3390/jimaging8120313
18. Zhou, B., Lu, Y., Hajifathalian, K., Bentham, J., Cesare, M. D., Danaei, G., Bixby, H., Cowan, M. J., Ali, M. K., Taddei, C., Lo, W. C., Reis-Santos, B., Stevens, G. A., Riley, L. M., Miranda, J. J., Bjerregaard, P., Rivera, J. A., Fouad, H. M., Ma, G., Mbanya, J. C., et al.: Worldwide trends in diabetes since 1980: A pooled analysis of 751 population-based studies with 4·4 million participants. The Lancet, vol. 387, no. 10027, pp. 1513–1530 (2016) doi: 10.1016/s0140-6736(16)00618-8
19. Zuiderveld, K.: Contrast limited adaptive histogram equalization. Graphics Gems IV, pp. 474–485 (1994)

ISSN 1870-4069

Identificación automática de divulgadores de noticias falsas mediante el perfilado de autor

Cesar Macias, Miguel Soto,
Hiram Calvo, José E. Valdez-Rodríguez

Instituto Politécnico Nacional,
Centro de Investigación en Computación,
Laboratorio de Ciencias Cognitivas Computacionales,
México

{cmaciass2021,msotoh2021,
hcalvo,jvaldezr2018}@cic.ipn.mx

Resumen. El campo del procesamiento de lenguaje natural tiene una tarea encargada de realizar el perfilado de autores. Su objetivo es extraer características semánticas y de estilo de los textos que se analizan, para identificar a quién los escribió. En el marco del congreso *Conference and Labs of the Evaluation Forum* (CLEF) del 2020, los organizadores propusieron una tarea en conjunto con el grupo *Web Technology & Information Systems Group* (Webis), líderes de la organización PAN. Dicha tarea constaba en realizar el perfilado de los autores que divulgan noticias falsas en Twitter. En el presente artículo, se exploran diversos algoritmos de aprendizaje automático, con múltiples combinaciones de características del texto. Una de las ideas aquí exploradas es darle a los clasificadores la capacidad de generalizar la información, para su futura implementación en distintas plataformas sociales.

Palabras clave: Perfilado de autor, procesamiento de lenguaje natural, redes sociales, noticias falsas.

Automatic Identification of Fake News Spreaders through Author Profiling

Abstract. The field of natural language processing has a task of author profiling, which aims to extract stylistic and semantic features from the texts being analyzed in order to identify who wrote them. During the 2020 Conference and Labs of the Evaluation Forum (CLEF), the organizers proposed a task in conjunction with the Web Technology and Information Systems Group (Webis), leaders of the PAN organization. The task was to profile authors who spread fake news on Twitter. In this paper, various machine learning algorithms, with multiple combinations of text features, are explored. One of the ideas explored here is to give the classifiers the ability to generalize the information for future implementation on different social platforms.

Keywords: Author profiling, natural language processing, social networks, fake news.

1. Introducción

El perfilado de autor (PA) es una tarea importante en el procesamiento del lenguaje natural (NLP, por sus siglas en inglés), que consiste en inferir características de los autores de un texto a partir de su contenido, basándonos en las características de estilo y aquellas propias del texto.

La edad, el sexo, la personalidad y la ocupación del autor son algunas de las características que se suelen analizar. Esta tarea tiene múltiples aplicaciones, tales como en la verificación de la identidad de los autores, en la detección de la autoría de textos anónimos, en la clasificación de usuarios en redes sociales y en el análisis de la opinión pública, entre otras.

Desde su creación, la Internet se ha convertido en una de las principales herramientas para la consulta y divulgación de información. Actualmente, con el desarrollo y expansión de las redes sociales, la forma en la que los seres humanos interactuamos se ve muy afectada, ya que nos hemos acostumbrado a tener relaciones interpersonales exclusivamente en línea, mediante el uso de las redes sociales. Aunado a esto, la reciente pandemia de COVID-19 ha causado una revolución en este fenómeno, ya que se nos ha forzado a dialogar utilizando medios digitales exclusivamente.

Aproximadamente $4,760$ millones de habitantes utilizamos activamente las redes sociales como Facebook, Twitter o Reddit, lo que representa el $59{,}4\,\%$ de la población mundial; en promedio, cada uno de los usuarios activos invertimos dos horas, treinta y un minutos diariamente en esta actividad, de acuerdo con Digital 2023[1].

Las plataformas en línea intervienen cada vez más en el discurso público y los algoritmos nos ayudan a unirnos a grupos sociales, a clasificar el ruido del discurso público y estar al tanto de la actualidad [1].

Las redes sociales permiten a sus usuarios compartir fácilmente sus publicaciones favoritas con cientos de sus contactos, y, con el rápido crecimiento de las mismas, los motores de búsqueda nos facilitan la diversidad de voces al ofrecernos acceso a una amplia variedad de opiniones e información.

Pero, con tal cantidad de información, la variedad se mezcla con la ambigüedad, que al final produce un flujo de información impreciso. El manual *Journalism, "Fake news" & Disinformation* [8] clasifica al periodismo en:

- Periodismo de calidad. Cumple con los estándares profesionales y de ética.
- Periodismo débil. No cumple con los estándares profesionales ni éticos.
- Desinformación. Intentos deliberados de confundir o manipular a las personas mediante la entrega de información deshonesta.
- Información errónea. Información engañosa, creada o diseminada, sin intención manipuladora o maliciosa.

Las noticias falsas abarcan la desinformación y la información errónea. En [9], Kumar y Shah categorizan a la información falsa como basada en opinión (*opinion-based*) en la que no existe una verdad sólida y se presenta en casos como las reseñas de productos, o como basada en hechos (*fact-based*) que consiste en mentiras

[1] https://wearesocial.com/es/blog/2023/01/digital-2023/

sobre entidades que tienen un valor de verdad fundamental como las noticias falsas y los rumores.

Los autores involucrados en la creación de información, aprovechan las redes sociales para hacer la difusión de la misma, aprovechando la dinámica existente en las redes en la que los usuarios comparten las publicaciones que más les gustan o que les parecen más interesantes con cientos de sus contactos en muy poco tiempo.

Esta dinámica convierte a las redes sociales en el medio ideal para propagar la información. Los autores cuyo objetivo es crear desinformación o engañar a sus lectores integran a sus filas ejércitos de bots, encargados de hacer que las noticias falsas sean consideradas por la plataforma como temas en tendencia, ampliando el alcance que pueden llegar a tener [9], es por ello que es importante darle solución a esta problemática.

El resto del trabajo se encuentra organizado de la siguiente manera: la sección 2 describe los trabajos anteriormente desarrollados que mantienen una estrecha relación con el tema aquí tratado. La sección 3 describe la metodología propuesta para desarrollar la investigación, y sus experimentos.

La sección 4 muestra los resultados obtenidos tras la experimentación, la comparación entre los resultados aquí obtenidos y aquellos mostrados en el estado del arte. Finalmente, en la sección 5 proveen las conclusiones y se listan las ideas para desarrollar como trabajo futuro.

2. Estado del arte

La basta propagación de noticias falsas en años recientes ha causado mucha atención por parte de la comunidad científica y de la industria. A lo largo de esta sección, describiremos algunas de las más recientes aportaciones que se han realizado para contrarrestar este problema.

En 2021, Sahoo y Gupta [19], recuperan información de noticias falsas, perfiles que comparten este tipo de noticias y características del contenido compartido mediante la *Application Programming Interface* (API) de Facebook[2] y utilizan esta información para entrenar modelos de aprendizaje automático clásicos y de aprendizaje profundo; sus mejores resultados fueron obtenidos con un modelo *Long sort-term memory* (LSTM) [6] con una exactitud de 99.4 %.

En ese mismo año, Zhang y colegas [21], proponen usar características de emoción duales, las cuales hacen referencia a la emoción de quien publica la noticia y de los lectores. Para la clasificación hicieron uso de distintos modelos basados en *transformers* [20] los cuales ensamblaron, de tal manera que su mejor aproximación obtuvo un 93.2 %.

Para 2022, Raza y Ding [18], proponen un enfoque basado en *transformers* para la detección de noticias falsas, utilizando un conjunto de noticias de múltiples fuentes y los respectivos contextos sociales de los consumidores de estas noticias.

Esta información fue combinada de tal manera que se generó una representación vectorial, la cual sería su bloque de entrada para su bloque del *transformer* y

[2] https://www.facebook.com

posteriormente esta información sería enviada a una capa de clasificación. Su modelo, al que llaman *Fake News Detection through News content and Social context* (FND-NS) obtuvo una exactitud de 74.8 %.

Davoudi y colegas [5], presentan un modelo profundo híbrido para la detección de noticias falsas mediante el uso de un árbol de propagación y un sistema de apoyo de decisiones (DSS). Dichos componentes se ejecutan de manera simultánea, y con ellos buscan encontrar las características con mayo valor discriminativo, la diferencia que existe entre el patrón y las características extraídas a lo largo del tiempo; su modelo obtuvo una exactitud de 98.4 %.

2.1. Enfoques basados en el perfilado de autor

La intención de estas aproximaciones es construir la reputación de un autor al compartir una noticia y con base en las características extraídas de este autor, identificar si es un posible dispersor de noticias falsas.

A lo largo de la última década, el CLEF, en conjunto con el PAN, han propuesto distintas tareas para el PA. Durante el marco del congreso CLEF del año 2020 [15], serían unos de los pioneros en proponer la identificación de noticias falsas mediante el PA, y propusieron una tarea titulada *"Profiling Fake News Spreaders on Twitter"* en el cual participaron 66 equipos, los cuales propusieron el uso de distintas combinaciones de características para perfilar autores.

Por mencionar algunas aproximaciones de este congreso, Pizarro [13] combinó n-gramas de palabras y caracteres, y obtuvo una exactitud promedio de 77.75 %. Por otro lado, Manna y colegas [10] combinaron el número promedio de emojis (clasificados en categorías como afecto, emoción, consternado, etc.), el número de signos de puntuación, *tags*, espacios, enlaces web, etcétera, y obtuvieron una exactitud promedio de 66 %.

Sin embargo, su mejor aproximación para esta tarea ese año fue llevada a cabo por Buda y Bolonyai [3] quienes obtuvieron una exactitud de 77.75 % y lo lograron realizando un ensamble de modelos de aprendizaje automático clásicos haciendo uso de n-gramas con algunas estadísticas para los tuits, así como de la longitud promedio de la diversidad léxica del conjunto de datos.

Sin embargo, el enfoque de PA para la detección de noticias falsas ha continuado hasta la fecha, ya que un año más tarde, en 2021, Rathod [17], retoma esta línea de investigación, enfocándose en una construcción más sólida de un perfil de autor utilizando características de autoría, *Named Entity Recognition* (NER), sentimiento y emociones y estilometría.

A diferencia de las aproximaciones realizadas en CLEF 2020, en este trabajo utilizaron un conjunto distinto de datos y obtuvieron una exactitud de 85 %. En 2022, Cervero y colegas [4] continúan con el uso de la información utilizada en [15]; uno de los grandes aportes de este trabajo es que logran extraer información visual de los tuits del conjunto de datos mediante la API de Twitter. Con esta información lograrían un aumento de datos y consigo una exactitud promedio de 80.5 %.

Tabla 1. Distribución de clases, conjunto de entrenamiento y prueba

Conjunto	**Inglés**	**Español**	
Entrenamiento	300	300	
Prueba	200	200	
Total	500	500	1,000

3. Metodología

En esta sección se describirá la metodología seguida para desarrollar la investigación del presente trabajo. Se presenta una variedad de algoritmos clásicos de clasificación del aprendizaje automático, se describe el preprocesamiento que se le dio a los textos, y los métodos de extracción de características: TF·IDF y los embeddings de palabras. Se pretende abordar la tarea "*Profiling Fake News Spreaders on Twitter 2020*" propuesta en el marco del congreso *Conference and Labs of the Evaluation Forum* (CLEF) en su edición 2020.

3.1. Corpus

El corpus seleccionado para realizar la tarea del perfilado de autores, fue el desarrollado por PAN[3] para el congreso CLEF 2020 [16]. El corpus contiene un total de $1,000$ autores, de los cuales 500 son autores de textos en inglés y 500 son autores de textos en español.

Para cada autor se recolectaron un total de 100 tuits etiquetados como 0 o 1, indicando si el texto corresponde a una noticia falsa o no. La distribución de clases del corpus se muestra en la Tabla 1. La descripción detallada del desarrollo del corpus y sus componentes se encuentra en [15, 16].

3.2. Preprocesamiento

Para el preprocesamiento de los textos, se utilizó un script desarrollado por los autores en conjunto[4]. Las operaciones realizadas en el script se describen de forma general a continuación.

- Entidades HTML: se remueven las entidades HTML que contenga el texto.
- Saltos de línea: se quitan los saltos de línea.
- Hashtags: En caso de haber hashtags, se separa el texto contenido en los mismos (p.e. #NoticiasFalsas $\rightarrow$ Noticias Falsas).
- Entidades de Twitter: se les dice así a las entidades que se utilizan propiamente en Twitter para denotar usuarios, etiquetas, hashtags y retuits, cada uno de estos tiene un identificador especial (@User, rt, #hashtag), se identifican estas entidades y se remueven del texto.
- URLs: se identifican y se remueven del texto.
- Las letras se convierten a minúsculas para homogeneizar el texto.

[3] https://pan.webis.de/

[4] https://github.com/CCogS-Mx/text-preprocessing

Tabla 2. Configuraciones de la experimentación.

Vectorizadores		**Modelos**	
TF·IDF	Secuencias de n-gramas de 1 a 3 (1G, 2G, 3G). Frecuencia mínima de aparición de palabras de 1 y 3 (1FM y 3FM).	LR	`penalty`: 'L2' `solver`: 'liblinear' `max_iter`: 10000
		LSVC	`penalty`: 'L2' `max_iter`: 10000
Word embeddings	Inglés: fasttext-english-twitter-100d [11] Español: fasttext-english-twitter-100d [11]	RF	`n_estimators`: 100 `max_depth`: 100

– Palabras auxiliares: en caso de que así se requiera, se remueven las palabras auxiliares que contenga el texto.
– Lematización: si se requiere, las palabras son lematizadas utilizando la librería spaCy [7].
– Apóstrofes: tras la lematización, se remueven los apóstrofes del texto, conservando el caracter sin el apóstrofe (p.e observación → observacion).
– Puntuación: se remueven los caracteres utilizados para puntuar el texto (puntos, comas, punto y comas, etc.).
– Caracteres repetidos: en caso de que un caracter se repita más de dos veces, este se corta a dos repeticiones (p.e. Holaaaaaaa → Holaa).
– Palabras alfanuméricas: si el texto contiene palabras compuestas por letras y números, como en el leet speaking, estas se remueven (p.e. P3*r4).
– Caracteres especiales: se remueven todos los caracteres especiales, signos de admiración, interrogación, etc.
– Espacios en blanco: en caso de que exista más de un espacio en blanco entre palabras, estos se remueven para homogeneizar el texto.

3.3. Extracción de características

Una vez terminado el preprocesamiento, se realizó la extracción de características y tokenización del texto. Para tal efecto, se utilizaron los métodos TF·IDF y embeddings de palabras.

TF·IDF. EL método TF·IDF hace uso de la frecuencia de los términos (TF, por sus siglas) y la frecuencia inversa de documento (IDF, por sus siglas). Para el cálculo de TF, se genera una bolsa de palabras con el vocabulario del conjunto de todos los documentos que van a ser analizados, posteriormente se obtiene el total de apariciones de la palabra en el documento analizado. Para calcular IDF se calcula el logaritmo del cociente de la cantidad de documentos en los que la palabra analizada aparece y la cantidad total de documentos en el conjunto analizado. Finalmente, se realiza el producto entre TF e IDF para cada una de las palabras del texto analizado, generando un vector de características.

***Embeddings* de palabras.** El *embedding* de una palabra, es la representación, usualmente representada por un vector n-dimensional, en la que se codifica el significado de la palabra, asignando una posición dentro de un espacio de

Tabla 3. Mejores resultados para los modelos vectorizados por TF·IDF medidos por la métrica *accuracy*.

		LR		**LSMV**		**RF**	
Característicasdel texto	Combinación	EN	ES	EN	ES	EN	ES
Texto en crudo	1g + 1FM	0.62	0.65	0.61	0.63	0.60	0.67
	1g + 2g + 1FM	**0.63**	**0.68**	**0.63**	**0.66**	**0.61**	0.67
	1g + 2g + 3g + 1FM	0.62	0.67	**0.63**	**0.66**	**0.61**	0.66
	1g + 3FM	0.61	0.65	0.60	0.63	0.60	0.67
	1g + 2g + 3FM	0.62	**0.68**	0.62	0.65	0.60	**0.68**
	1g + 2g + 3g + 3FM	0.62	**0.68**	0.61	0.65	0.59	**0.68**
Texto con preprocesamiento	1g + 1FM	0.60	0.65	0.60	0.63	**0.61**	0.67
	1g + 2g + 1FM	0.62	**0.68**	0.62	**0.66**	**0.61**	0.67
	1g + 2g + 3g + 1FM	0.62	0.67	0.62	**0.66**	**0.61**	0.66
	1g + 3FM	0.60	0.65	0.60	0.63	**0.61**	0.67
	1g + 2g + 3FM	0.62	0.67	0.61	0.65	**0.61**	**0.68**
	1g + 2g + 3g + 3FM	0.62	0.67	0.62	0.65	**0.61**	**0.68**

características, utilizando las palabras con significados más similares. Este vector de características, nos indica la zona del espacio de características en la que se ubica la palabra analizada, por su similitud con otras palabras.

3.4. Modelos propuestos

Para realizar la clasificación, se propusieron tres diferentes algoritmos de clasificación del aprendizaje automático clásicos, además estos modelos son continuamente utilizados en la literatura para afrontar esta tarea.

Los algoritmos de clasificación son: regresión logística (LR), máquina de soporte vectorial con kernel lineal (LSVM) y bosque aleatorio (RF). A continuación se da una descripción general de cada uno de los algoritmos y su funcionamiento.

Regresión logística (LR). Es un modelo estadístico que estima la probabilidad de que un evento ocurra. Para el caso binario, este modelo calcula la probabilidad $[0, 1]$ de que una muestra i pertenezca a la clase y_i. La representación matemática del modelo es la siguiente [12]:

$$P(y_i = 1|X_i) = \frac{1}{1 + exp(-X_i w - w_0)}, \tag{1}$$

donde X representa el tuit; y denota la etiqueta de clase; $w \in \mathbf{R}^n$ es el vector de pesos.

Máquina de soporte vectorial con kernel lineal (LSVM). La máquina de soporte vectorial con kernel lineal (LSVM) hace uso del kernel lineal para permitir a la SVM, operar en espacios de alta dimensión, haciendo la transformación de dimensiones con este kernel. Las SVMs son algoritmos de clasificación cuyo objetivo es encontrar el hiperplano de separación óptimo entre clases. Este hiperplano actúa como frontera de decisión para asignar la clase final a un dato de entrada que deba ser clasificado. El hiperplano es encontrado al maximizar el margen (distancia entre el hiperplano y los vectores de soporte).

Tabla 4. Mejores resultados para los modelos vectorizados por *word embeddings* medidos por la métrica *accuracy*.

	LR		**LSVC**		**RF**	
Características del texto	EN	ES	EN	ES	EN	ES
Texto en crudo	0.57	**0.61**	0.57	**0.61**	0.57	**0.61**
Texto con preprocesamiento	**0.59**	0.52	**0.59**	0.52	**0.61**	0.52

Bosque aleatorio (RF). Un bosque aleatorio es un algoritmo de clasificación que consiste en acoplar un conjunto de clasificadores estructurados en forma de árbol $\{h(x, \Theta_k), k = 1, \ldots\}$ en el que $\{\Theta_k\}$ son vectores aleatorios independientes, distribuidos de forma idéntica y, en el que cada árbol obtiene un voto unitario para la clase más popular para la entrada x [2]. La librería `scikit-learn` [12] implementa este bosque de árboles aleatorios como un ensamble de árboles construido de una muestra del conjunto de entrenamiento, que divide los nodos de cada árbol de un subconjunto aleatorio de tamaño igual a las características que se estén extrayendo. En el que la predicción probabilística de cada árbol se promedia para obtener la clase final.

3.5. Métricas de evaluación

En [15], se detalla la forma en la que se va a evaluar a los participantes de la tarea. La métrica de desempeño propuesta por los organizadores, que además se utilizó en este artículo para comparar con los resultados del estado del arte, es la exactitud (*accuracy*) que se obtuvo para cada uno de los lenguajes contenidos en el corpus. El cálculo de esta métrica se describe matemáticamente en la ecuación 2:

$$accuracy = \frac{\text{Numero de predicciones correctas}}{\text{Total de predicciones}} = \frac{V_P + V_N}{V_P + V_N + F_P + F_N}, \quad (2)$$

donde V y F corresponden a Verdadero y Falso y los subíndices P y N corresponden a positivo y negativo. Por lo tanto, V_N se lee como verdaderos negativos.

4. Experimentos y resultados

En esta sección se describen los experimentos realizados durante el desarrollo de la investigación, la configuración de cada uno de estos, los resultados obtenidos, así como una breve discusión de los mismos.

4.1. Experimentos

Para realizar nuestras aproximaciones, hicimos uso de los 3 modelos descritos en la sección 3. Para implementar y entrenar los modelos se utilizó la librería de `scikit-learn` [12] para Python (versión 3.9.5). Con la finalidad de que nuestros experimentos puedan ser replicables, asignamos la semilla 42 para la generación de números aleatorios.

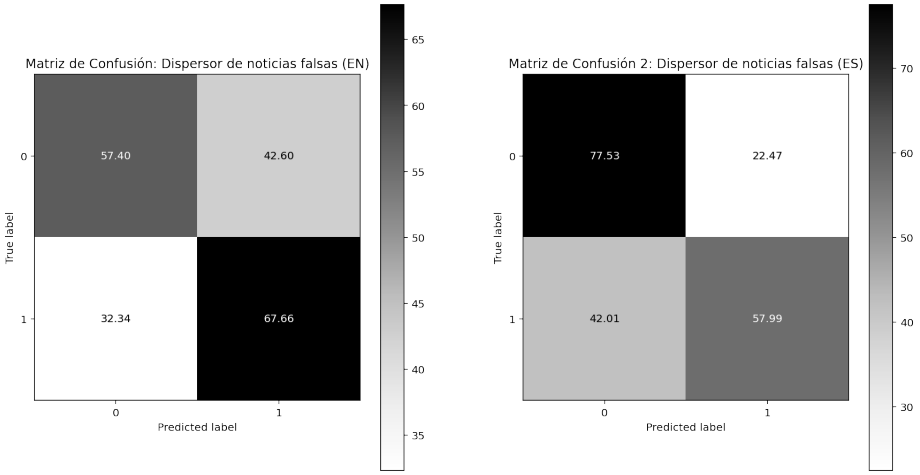


Fig. 1. Matrices de confusión del mejor modelo (RL), obtenido para ambos casos: inglés (izquierda) y español (derecha).

Se llevaron a cabo pruebas con varias configuraciones para cada modelo, utilizando diferentes secuencias de palabras en n-gramas, distintos hiperparámetros y dos opciones de *word embeddings*. Asimismo, se realizaron experimentos con el texto sin procesar y con el preprocesamiento propuesto.

Los modelos propuestos para realizar la clasificación, así como las combinaciones de los métodos de extracción de características (TF·IDF con secuencias de n-gramas y *embeddings*) se muestran en la tabla 2.

4.2. Resultados

Durante el proceso de experimentación, realizamos múltiples iteraciones variando los distintos hiperparámetros en cada modelo. Estos experimentos nos permitieron observar como los cambios en los parámetros afectan su desempeño.

A pesar de que obtuvimos resultados interesantes y valiosos en nuestro estudio del problema, nos enfocaremos únicamente en los mejores resultados, ya que nuestra intención es comparar este trabajo con el estado del arte.

En este sentido, presentaremos los modelos que lograron obtener un mayor *accuracy* en cada tarea, así como sus configuraciones respectivas. Estos resultados se pueden observar en las tablas 3 y 4, dónde, las mejores configuraciones fueron las siguientes:

En el caso de los modelos para los que se hizo uso de la vectorización TF·IDF, el rango de n-gramas que se utilizó, y el que mejor desempeño obtuvo, fue con secuencias $(1, 2)$, es decir, unigramas y bigramas con el modelo LR, con una exactitud promedio de ambos idiomas de 66 %.

Por otro lado, los mejores resultados que obtuvimos con los *word embeddings* se dieron con el modelo de RF, sin embargo, para lograr el mejor ensamble, se tuvieron

Tabla 5. Comparación contra los mejores resultados del estado del arte.

Autor	Inglés	Español	Promedio
Buda y Bolonyai [3]	**0.750**	0.805	0.7775
Pizarro [14]	0.735	**0.820**	0.7775
Nuestra propuesta	0.630	0.680	0.6550

que unificar los modelos con texto preprocesado y con texto en crudo para el idioma inglés y español respectivamente, obteniendo una exactitud promedio de 66 %.

De esta manera, nuestra mejor aproximación de todos los experimentos que se realizaron fue el modelo de RL con la combinación de unigramas, bigramas y frecuencia mínima de aparición de palabras de 1. Lo cual nos indica que para el modelo hay palabras o secuencias de palabras que, a pesar de que aparezcan solo una vez en el conjunto de entrenamiento, pueden ser de gran importancia cuando se prueban sobre el conjunto de pruebas.

En la Figura 1, se pueden visualizar las matrices de confusión del mejor modelo para cada uno de los idiomas. Para el caso del inglés, se logra observar que la clasificación de la etiqueta 0 es muy ambigua, ya que tiende a clasificar 57.40 % de los datos bien; para la etiqueta 1, esta clasificación mejora significativamente, otorgando un 67.66 % de clasificación correcta. Para el español, obtenemos una mejor clasificación de la etiqueta 0 con una exactitud de 77.53 % y una peor clasificación para la etiqueta 1 con un 57.99 % de exactitud con respecto al inglés.

4.3. Comparación con el estado del arte

En la tabla 5 se muestra una comparación entre el mejor modelo que obtuvimos durante el proceso de experimentación de nuestra propuesta contra los mejores resultados obtenidos de esta competencia [15].

Al darle un enfoque general a los clasificadores, los modelos no logran identificar por completo las características específicas que se pudieron haber obtenido del conjunto de datos. Es decir, al utilizar los hiperparámetros descritos en la tabla 2 no logramos captar estas características que pudieron influir de manera positiva en nuestra fase de experimentación.

5. Conclusiones y trabajo futuro

Los modelos propuestos durante el desarrollo de la investigación, obtuvieron resultados prometedores. A pesar de que el desempeño obtenido por nuestra mejor aproximación no sobrepasa al mejor de los concursantes durante el CLEF 2020, las características que estamos extrayendo son únicamente del texto, además de que a pesar de que se utilizaron los embeddings de palabras, la clasificación no mejora significativamente en comparación con el método TF·IDF.

En español, la identificación de autores que no propagan noticias falsas se hace mejor que en inglés, lo que indica que existen características propias de quienes hacen periodismo de calidad en español que son más fáciles de identificar para estos algoritmos.

Dado que en esta primera iteración de experimentación se obtuvieron resultados prometedores y que existe la posibilidad de mejorar el estado del arte, nuestras ideas para nuestro trabajo futuro son las siguientes: probar con otros métodos de obtención de características, como una bolsa de palabras por conteo, en la que se identifiquen el número de apariciones de ciertos patrones como los URLs, o las menciones de usuarios, utilizar algoritmos de aprendizaje profundo como redes neuronales, o transformadores con modelos de lenguaje a gran escala.

Aumentar el vector de características con el etiquetado gramatical (POS), la complejidad de las oraciones y el tipo de lenguaje utilizado en los textos.

Referencias

1. Bastick, Z.: Would you notice if fake news changed your behavior? An experiment on the unconscious effects of disinformation. Computers in Human Behavior, vol. 116 (2021) doi: 10.1016/j.chb.2020.106633
2. Breiman, L.: Random forests. Machine Learning, vol. 45, pp. 5–32 (2001) doi: 10.1023/A:1010933404324
3. Buda, J., Bolonyai, F.: An ensemble model using N-grams and statistical features to identify fake news spreaders on twitter. In: Conference and Labs of the Evaluation Forum (Working Notes) (2020)
4. Cervero, R., Rosso, P., Pasi, G.: Profiling fake news spreaders: Personality and visual information matter. In: Natural Language Processing and Information Systems: 26th International Conference on Applications of Natural Language to Information Systems, NLDB 2021, pp. 355–363 (2021) doi: 10.1007/978-3-030-80599-9_31
5. Davoudi, M., Moosavi, M. R., Sadreddini, M. H.: DSS: A hybrid deep model for fake news detection using propagation tree and stance network. Expert Systems with Applications, vol. 198 (2022) doi: 10.1016/j.eswa.2022.116635
6. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation, vol. 9, no. 8, pp. 1735–80 (1997) doi: 10.1162/neco.1997.9.8.1735
7. Honnibal, M., Montani, I., Van Landeghem, S., Boyd, A.: Industrial-strength natural language processing in python: spaCy. (2020)
8. Ireton, C., Posseti, J.: Journalism ”fake news”& disinformation: Handbook for journalism education and training. Organización de las Naciones Unidas para la Educación, la Ciencia y la Cultura (2018)
9. Kumar, S., Shah, N.: False information on web and social media: A survey. Computer Science, vol. 1, no. 1 (2018)
10. Manna, R., Pascucci, A., Monti, J.: Profiling fake news spreaders through stylometry and lexical features. UniOR NLP@ PAN2020. In: Conference and Labs of the Evaluation Forum (Working Notes) (2020)
11. Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., Joulin, A.: Advances in pre-training distributed word representations. In: Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018) (2018)
12. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, vol. 12, pp. 2825–2830 (2011)
13. Pizarro, J.: Profiling bots and fake news spreaders at PAN'19 and PAN'20: Bots and gender profiling 2019, profiling fake news spreaders on twitter 2020. In: 2020 IEEE 7th International

Conference on Data Science and Advanced Analytics (DSAA), pp. 626–630 (2020) doi: 10.1109/DSAA49011.2020.00088
14. Pizarro, J.: Using N-grams to detect fake news spreaders on twitter. In: Conference and Labs of the Evaluation Forum (working notes) (2020)
15. Rangel, F., Giachanou, A., Ghanem, B., Rosso, P.: Overview of the 8th author profiling task at PAN 2020: Profiling fake news spreaders on twitter. In: CEUR Workshop Proceedings, vol. 2696, pp. 1–18 (2020)
16. Rangel, F., Rosso, P., Ghanem, B., Giachanou, A.: Profiling fake news spreaders on twitter. Zenodo, (2020) doi: 10.5281/zenodo.4039435
17. Rathod, S.: Exploring author profiling for fake news detection. In: 2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 1614–1619 (2022) doi: 10.1109/COMPSAC54236.2022.00256
18. Raza, S., Ding, C.: Fake news detection based on news content and social contexts: A transformer-based approach. International Journal of Data Science and Analytics, vol. 13, no. 4, pp. 335–362 (2022) doi: 10.1007/s41060-021-00302-z
19. Sahoo, S. R., Gupta, B.: Multiple features based approach for automatic fake news detection on social networks using deep learning. Applied Soft Computing, vol. 100 (2021) doi: 10.1016/j.asoc.2020.106983
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
21. Zhang, X., Cao, J., Li, X., Sheng, Q., Zhong, L., Shu, K.: Mining dual emotion for fake news detection. In: Proceedings of the Web Conference 2021, pp. 3465–3476 (2021) doi: 10.1145/3442381.3450004

ISSN 1870-4069

Modelos 3D de reconstrucción facial: Una breve revisión

Victor Hernández-Manrique[1], Miguel González-Mendoza[1], Carlos Vilchis[1], Mauricio Méndez-Ruiz[2], Carmina Pérez-Guerrero[2]

[1] Tecnológico de Monterrey, Escuela de Ingeniería y Ciencias, México

[2] Eugenia Virtual Humans S.A. de C.V., Laboratorio de Investigación, México

{A01731594,carlos.vilchis, mgonza}@tec.mx, {mauricio,carmina}@eugenia.tech

Resumen. Los algoritmos de reconstrucción facial 3D han demostrado ser una herramienta poderosa para diversas aplicaciones, incluido el reconocimiento facial, la realidad virtual y las imágenes médicas. Sin embargo, la complejidad y los recursos computacionales requeridos para estos algoritmos, así como la restringida disponibilidad de los conjuntos de datos (*datasets*), han limitado su accesibilidad a investigadores y profesionistas. Esto es desafortunado porque la reconstrucción facial en 3D tiene el potencial de beneficiar a una gama más amplia de individuos y comunidades, incluidos aquellos que forman parte de sectores como la atención médica, entretenimiento y seguridad. La siguiente investigación indaga en la evolución de esta tecnología, así como en su accesibilidad para las masas. Se presenta una tabla comparativa de tres modelos 3D de reconstrucción facial con el fin de proporcionar las características básicas de cada método, lo que permite a cualquiera seleccionar un algoritmo que se adapte a las propiedades deseadas a través de una comparación directa de los atributos clave.

Palabras clave: Reconstrucción facial, modelos 3D, MICA, 3DDFA-V2, SynergyNet, dimensiones algorítmicas, generación de avatares.

A Brief Review on 3D Models for Facial Reconstruction

Abstract. 3D face reconstruction algorithms have proven to be a powerful tool for various applications, including facial recognition, virtual reality, and medical imaging. However, the complexity and computational resources required for these algorithms, as well as the restricted availability of datasets, have limited their accessibility to a select group of researchers and professionals. This is unfortunate because 3D face reconstruction has the potential to benefit a broader range of individuals and communities, including those in healthcare,

entertainment, and security. The following research dives into the evolution of this technology, as well as its accessibility to the masses. A comparative table of three 3D face reconstruction models is presented in order to provide the base characteristics of each method, allowing anyone to select an algorithm that suits the features they are looking for through a straightforward comparison of key attributes.

Keywords: 3D face reconstruction, MICA, 3DDFA-V2, SynergyNet, algorithmic dimensions, avatar generation.

1. Introducción

Durante el último siglo, la reconstrucción facial en 3D se ha convertido en un campo de investigación en rápida evolución, con el objetivo de crear modelos 3D de humanos realistas y precisos [7]. Este desarrollo no sería posible sin avances en áreas como la visión computacional o el aprendizaje automático, impulsados por su diversa gama de aplicaciones.

El sector del entretenimiento ofrece un nuevo y fascinante uso para la reconstrucción facial en 3D [6]. La capacidad de producir modelos 3D realistas de rostros humanos es cada vez más crucial para el desarrollo de tecnologías como la realidad virtual y aumentada. Esta tecnología permite el mapeo en tiempo real del propio rostro del usuario en un avatar digital, la cual sirve para experiencias virtuales inmersivas como videojuegos y conciertos de realidad virtual [12].

Además, la industria cinematográfica puede beneficiarse de la reconstrucción facial en 3D para duplicar a los artistas digitalmente, ya que podría ser usado para acrobacias, filmar situaciones que serían demasiado arriesgadas o muy costosas de realizar.

La atención médica es otra industria donde la reconstrucción facial en 3D se ve muy prometedora [13]. Los médicos e investigadores pueden examinar una variedad de deformidades faciales y problemas médicos con esta tecnología. Por ejemplo, los procedimientos difíciles como la cirugía de labio leporino y paladar hundido se pueden planificar y practicar utilizando modelos 3D de anatomía facial.

En los últimos años, las técnicas de aprendizaje profundo, siendo las redes neuronales convolucionales (*CNN*, por sus siglas en inglés) uno de los conceptos más revolucionadores en este campo, presentaron nuevos desafíos en este tema. El estudio y desarrollo de modelos ligeros para la creación de rostros humanos en 3D con bajo costo computacional ha llegado a llamar la atención de todos. La investigación en esta área digital abriría nuevas vías de conexión, teniendo como punto de partida el Metaverso [17].

2. Planteamiento del problema

Muchas dificultades persisten a pesar de los avances realizados en el campo de la reconstrucción facial 3D [1]. La ausencia de datos de entrenamiento es uno de los principales problemas.

Para lograr aprender las diferencias en la geometría facial y la textura, los sistemas de reconstrucción facial 3D se basan principalmente en los datos de entrenamiento. El proceso de obtener escaneos faciales 3D precisos a partir de un conjunto de datos extensos y variados sigue siendo desafiante y requiere mucho tiempo.

La precisión de los modelos reconstruidos puede verse afectada por el sobreajuste y la mala generalización debido a esta escasez de datos.

La complejidad computacional de los algoritmos es otra dificultad [20]. Las técnicas modernas a veces necesitan mucha potencia de procesamiento, lo que las hace inadecuadas para aplicaciones en tiempo real o dispositivos con capacidad de procesamiento limitada.

El requerimiento de métodos avanzados de registro y optimización, así como el manejo de estructuras de datos grandes y complicadas, lleva a esta complejidad computacional. Por lo tanto, todavía es difícil crear algoritmos ligeros y efectivos que puedan producir modelos 3D precisos en tiempo real.

La robustez a los cambios en la iluminación y las expresiones faciales es un requisito indispensable [14]. Incluso al capturar rostros con expresiones faciales extremas, los algoritmos de reconstrucción facial 3D deben ser capaces de manejar una variedad de situaciones de iluminación y registrar con precisión la geometría y textura facial. Sin embargo, la precisión de los modelos reconstruidos puede verse considerablemente afectada por estos cambios. Por lo tanto, para que la reconstrucción tenga éxito en aplicaciones prácticas, los enfoques que pueden manejar estas variaciones de manera robusta son esenciales.

Asimismo, la reconstrucción facial en 3D se ve desafiada por consideraciones de ética y privacidad [16]. Las preocupaciones sobre la seguridad y la privacidad pueden surgir cuando es posible producir representaciones 3D de alta calidad de las caras de las personas, especialmente cuando se utiliza un software de realidad virtual o reconocimiento facial. Dicho uso indebido de la tecnología puede resultar en posibles violaciones de privacidad, robo de identidad u otros fallos de seguridad.

3. Trabajo relacionado

3.1. Dimensiones algorítmicas

Los parámetros específicos, configuraciones y las decisiones de diseño que tienen un impacto en el comportamiento y la efectividad de un algoritmo se conocen como dimensiones algorítmicas [8]. La selección de la técnica de optimización, el tamaño y la estructura de la red neuronal, la selección de la función de pérdida, el número de capas y las funciones de activación utilizadas en cada capa son algunos ejemplos de estas dimensiones.

Los investigadores pueden afinar el comportamiento del algoritmo para ajustarse mejor al dominio del problema en particular y mejorar su rendimiento cambiando estas dimensiones. En general, un algoritmo de reconstrucción facial 3D se compone de un modelo de detección de rostros que marca el área donde se encuentra un rostro en la imagen de entrada. Los puntos de referencia faciales se obtienen para señalar objetos como labios, cejas, ojos, etc.

La etapa de alineación permite que el algoritmo tenga una base de referencia para el enmascaramiento. Se extraen las características de apariencia y una reducción de dimensionalidad obtiene los elementos más importantes para la escultura de un rostro humano en un software [15].

La precisión, la velocidad y el costo de computación de un modelo, características comunes de estos métodos, pueden verse fuertemente afectados por las dimensiones algorítmicas que se utilizan. Por ejemplo, una optimización en el costo computacional permitiría la operación de un modelo desde un hardware especializado a uno común.

3.2. Modelos

El *NoW Challenge* recopiló métodos de reconstrucción facial 3D de última generación basados en un conjunto de datos de referencia de escaneos faciales 3D e imágenes 2D correspondientes con diversos grados de dificultad [10].

El objetivo era establecer una medida de rendimiento consistente para la precisión y robustez de los modelos. Las evaluaciones de esta competencia consistieron en un rendimiento no métrico, donde se tuvo en cuenta la suavidad de la superficie o el nivel de detalle. El rendimiento métrico proporcionó un conjunto de resultados numéricos en los que los métodos se compararon entre sí.

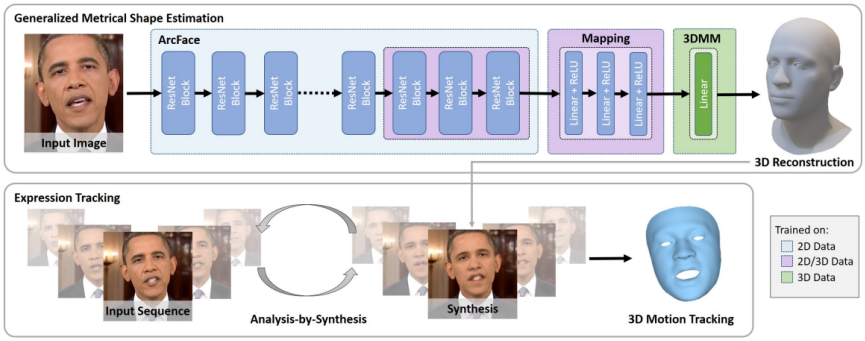
MICA. El modelo *Metric Face*, también conocido como MICA, es un método propuesto por Zielonka et al. en el 2022 [19]. Utilizaron una red de reconocimiento facial previamente entrenada en un conjunto de datos de imágenes 2D a gran escala para entrenar su estimador de forma facial de manera supervisada, lo que resulta en una mejor precisión que los métodos actuales de última generación.

A pesar de que este método tiene los mejores resultados en el NoW Challenge, no proporciona los puntos de referencia o la textura UV de una imagen de entrada. Su arquitectura, mostrada en la figura 1a, utiliza una versión modificada de ArcFace, un modelo de reconocimiento facial, como su codificador de identidad. Se realiza un mapeo 2D/3D después del proceso de ArcFace, entrenando datos 3D en un modelo lineal para su aplicación en los modelos transformables en 3D (3DMM, por sus siglas en inglés).

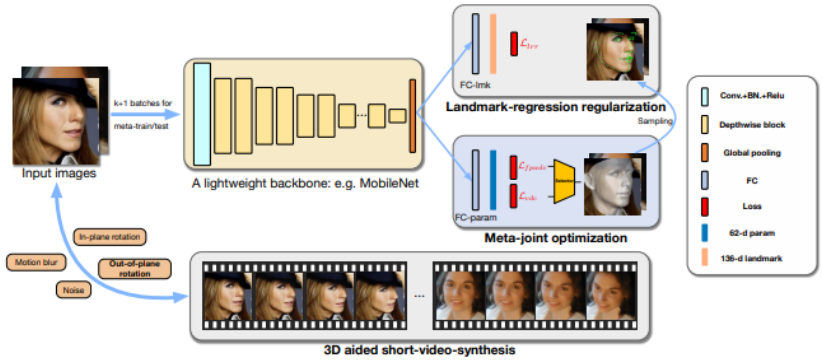
3DDFA-V2. La segunda versión de *3D Dense Face Alignment* (Alineación Densa de Rostros 3D), también conocida como 3DDFA-V2, equilibra la velocidad, la precisión y la estabilidad en comparación con su predecesora [3]. Utiliza MobileNet como un backbone ligero para predecir parámetros 3DMM (vistos en las figura 1b).

La regularización de regresión de puntos de referencia (*Landmark Regression Regularization* - LRR) proporciona parámetros que no se consideraron para regresar los puntos de referencia. La optimización del meta-conjunto (*Meta-joint Optimization*) combina el paso *LRR* con el rápido costo de distancia ponderado del parámetro (*fast Weighted Parameter Distance Cost* - fWPDC) y el costo de distancia de vértice (*Vertex Distance Cost* - VDC), el cual selecciona y actualiza los estados de los parámetros dentro del modelo.

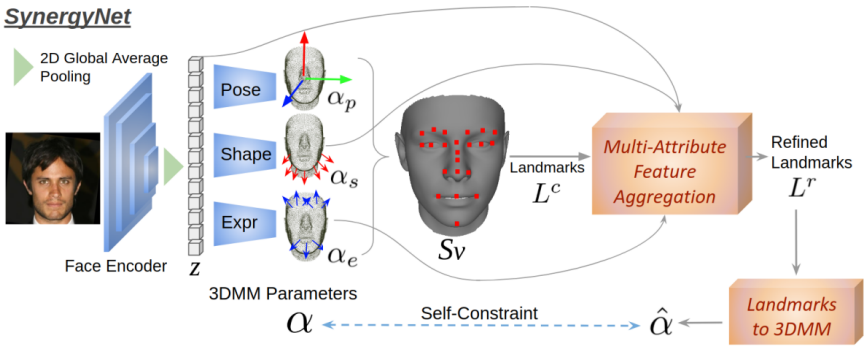
Finalmente, el análisis de video corto asistido por 3D (*3D aided short-video analysis*) simplemente expande una imagen para formar varios fragmentos.



(a) Arquitectura de MICA [19].



(b) Arquitectura de 3DDFA-V2 [3].



(c) Arquitectura de SynergyNet [18].

Fig. 1. Modelos 3D de reconstrucción facial.

SynergyNet. SynergyNet, desarrollado por Wu et al. en el 2021 [18], propone un proceso de sinergia entre los modelos transformables 3D (3DMM) y los puntos de referencia faciales 3D para predecir la geometría, la textura UV y, lógicamente, los puntos de referencia de cualquier entrada.

Tabla 1. Comparativa de dimensiones algorítmicas entre métodos.

Dimensiones Algorítmicas	**MICA**	**3DDFA-V2**	**SynergyNet**
Método de construcción	Reconstrucción métrica	Regresión Profunda + Refinamiento de Malla	Combinación entre 3DMM y puntos de referencia 3D
Complejidad del modelo	Alta	Baja	Moderado
Complejidad computacional	Alta	Baja - Media	Media - Alta
Robustez	Sensible a la orientación de la cara y la iluminación	Sensible a la orientación de la cara y la iluminación	Estable frente al cambio en la orientación de la cara y la iluminación
Precisión	Alto	Alto	Alto
Rendimiento	Rápido	En tiempo real	Rápido

Los puntos de referencia 3D se extraen y son refinados a partir de mallas faciales que se construyeron con parámetros 3DMM, las cuales son pose, forma y expresión. En comparación con 3DDFA-V2, el backbone en esta arquitectura, mostrado en la figura 1c, usa MobileNetV2.

3.3. Comparación de dimensiones algorítmicas entre los modelos 3D de reconstrucción facial

En el *NoW Challenge* [10], MICA obtuvo los mejores resultados tanto en la evaluación métrica como no métrica, con una mediana (en milímetros) de 1,08 y 0,90, respectivamente. 3DDFA-V2 y SynergyNet mantuvieron sus posiciones, siendo la última mencionada con los peores resultados en comparación con las otras dos.

Vale la pena mencionar que estos métodos tienen los procesos de instalación más sencillos en comparación con los otros algoritmos que se encuentran en el NoW Challenge. La sencillez de instalación y la compatibilidad de sus librerías fueron la razón principal por la que se eligieron estos algoritmos. Otros modelos no eran posibles de ejecutar ya que sus dependencias eran obsoletas o las versiones de librerías entraban en conflicto.

4. Discusión

Los aspectos algorítmicos de las tres propuestas demuestran que cada método tiene distintas ventajas y desventajas, como fue visto en la tabla 1. Debido a su complejidad de modelo relativamente alta, MICA requiere mucha potencia computacional para ejecutarse. Sin embargo, para lograr su gran precisión en la reconstrucción de geometría facial 3D, se requiere una complejidad considerable.

Por el contrario, el modelo 3DDFA-V2 es ligero y tiene un bajo nivel de complejidad, lo que permite velocidades de procesamiento más altas, pero posiblemente comprometiendo la precisión del resultado final.

SynergyNet logra una mezcla sólida entre las otras dos técnicas gracias a su moderada complejidad del modelo y gran precisión en la reconstrucción de la geometría facial 3D. Las principales diferencias de cada método se encuentran en sus limitaciones. MICA no presenta un análisis de los puntos de referencia o la textura UV de una imagen de entrada.

3DDFA-V2 lo hace, pero falla al no predecir la forma total de la cabeza correctamente. Finalmente, SynergyNet no puede producir una reconstrucción facial 3D basada en una imagen propia. Sus resultados son mallas 3D de datos internos. El trabajo a futuro se basará en el estudio de modelos ligeros que podrían reemplazar el backbone actual de alguno de los métodos para reducir el costo computacional sin comprometer la precisión del algoritmo.

Los modelos fueron entrenados y probados en diversos procesadores NVIDIA, por lo que se requiere un GPU de esta marca para ejecutar los algoritmos. Esto es importante de destacar y plantea la necesidad de reducir el costo computacional de los modelos, ya que no todos tienen la posibilidad de comprar este tipo de GPUs exigentes.

Brindar acceso a herramientas como los algoritmos de reconstrucción facial en 3D es esencial para garantizar que todos tengan la misma oportunidad de participar en diversas actividades que requieren dicha tecnología. Un área donde la reconstrucción facial en 3D se ha vuelto cada vez más relevante es en el campo de la medicina.

En procedimientos médicos como la cirugía de reconstrucción facial o la ortodoncia, la reconstrucción facial 3D puede proporcionar información valiosa y ayudar a los médicos a tomar decisiones vitales [13]. Brindar acceso a herramientas de reconstrucción facial 3D a todos puede ayudar a las personas de diferentes orígenes a recibir un mejor tratamiento, independientemente de su estatus financiero o social.

Utilizando la reconstrucción facial 3D en la educación, se pueden crear módulos de aprendizaje interactivos que pueden ayudar a los estudiantes a comprender mejor temas difíciles [2]. Por último, la reconstrucción facial en 3D en la aplicación de la ley puede ayudar con las investigaciones forenses y acelerar el proceso de búsqueda de personas desaparecidas o sospechosas [9].

Pero para que las técnicas de reconstrucción facial en 3D sean accesibles para todos, es necesario resolver una serie de problemas, incluido el alto costo computacional y la escasez de datos de entrenamiento. El desarrollo de algoritmos más precisos y eficientes que puedan ejecutarse en muchas plataformas de hardware y software debe seguir siendo el objetivo de los investigadores y desarrolladores. Adicionalmente, se deben hacer esfuerzos para aumentar la facilidad de uso de estas tecnologías para las personas que podrían carecer de los medios o la experiencia para hacerlo.

5. Conclusión

Esta investigación presentó una comparación de dimensiones algorítmicas de tres modelos de reconstrucción facial 3D de última generación. Su principal similitud se encuentra en la precisión de estos métodos, pero los diversos procesos que realiza cada algoritmo o sus complejidades son características importantes que un usuario debe tener en cuenta al momento de elegir el procedimiento que mejor se ajuste a sus necesidades, así como las limitaciones que se discutieron.

Como se mencionó anteriormente, la disponibilidad de recursos informáticos y técnicas de aprendizaje automático ha llevado a un gran avance en los algoritmos de reconstrucción facial 3D en los últimos años.

Pero también ha habido más dificultades en la creación de modelos 3D precisos y realistas de rostros humanos a un bajo costo computacional [5]. La ausencia de conjuntos de datos extensos y diversos, la complejidad de las expresiones faciales y la necesidad de lograr un equilibrio entre precisión y eficiencia computacional son solo algunas de estas dificultades.

Por lo tanto, la creación de modelos ligeros para la reconstrucción de rostros en 3D es crucial para superar estas dificultades. Al lograr esto, sería posible poner estos modelos en uso en hardware estándar, como teléfonos inteligentes o tabletas, abriendo la tecnología a un público más amplio. Esta accesibilidad sería ventajosa en una variedad de industrias donde la reconstrucción facial 3D tiene la capacidad de aumentar y actualizar las aplicaciones actuales [4, 11].

Garantizar que las personas de diversos orígenes y estatus socioeconómicos puedan beneficiarse de los avances de estas aplicaciones por igual requiere otorgar acceso a estas herramientas y tecnología a todos.

Fomentar la alfabetización digital y proporcionar a los usuarios las herramientas que necesitan para generar y utilizar adecuadamente los modelos de reconstrucción facial en 3D implica poner a disposición el hardware y el software necesarios, así como proporcionar recursos de capacitación e instrucción.

La creación de modelos portátiles de reconstrucción facial 3D y la expansión de su comerciabilidad pueden resultar en avances sustanciales en una serie de industrias y dar a todos la misma oportunidad de beneficiarse de estas tecnologías de vanguardia.

Agradecimientos. Los autores quieren agradecer el apoyo financiero del Tecnológico de Monterrey a través del programa “Challenge-Based Research Funding Program 2022”. Project ID # E120 - EIC-GI06 - B-T3 - D.

Referencias

1. Feng, M., Gilani, S. Z., Wang, Y., Mian, A.: 3D face reconstruction from light field images: A model-free approach. In: Proceedings of the European Conference on Computer Vision (ECCV), vol. 11214, pp. 501–518 (2018) doi: 10.1007/978-3-030-01249-6_31
2. Goeser, P. T., Hamza-Lup, F. G., Johnson, W. M., Scharfer, D.: VIEW: A virtual interactive web-based learning environment for engineering. Advances in Engineering Education, vol. 2, no. 3, pp. 1–24 (2011) doi: 10.48550/arXiv.1811.07463
3. Guo, J., Zhu, X., Yang, Y., Yang, F., Lei, Z., Li, S. Z.: Towards fast, accurate and stable 3D dense face alignment. In: Computer Vision- European Conference on Computer Vision (ECCV 2020), pp. 152–168 (2020) doi: 10.1007/978-3-030-58529-7_10
4. Jin, H., Wang, X., Zhong, Z., Hua, J.: Robust 3D face modeling and reconstruction from frontal and side images. Computer Aided Geometric Design, vol. 50, pp. 1–13 (2017) doi: 10.1016/j.cagd.2016.11.001
5. Korban, M., Li, X.: A survey on applications of digital human avatars toward virtual co-presence (2022)

6. Molina, L.: Celebrity avatars: A technical approach to creating digital avatars for social marketing strategies. Ph. D. thesis, Florida Atlantic University (2021)
7. Morales, A., Piella, G., Sukno, F. M.: Survey on 3D face reconstruction from uncalibrated images. Computer Science Review, vol. 40 (2021) doi: 10.1016/j.cosrev.2021.100400
8. Ngu, A.: Dimensional complexity and algorithmic efficiency. International Journal of Modern Nonlinear Theory and Application, vol. 11, no. 1 (2021) doi: 10.4236/ijmnta.2022.111001
9. Raneri, D.: Enhancing forensic investigation through the use of modern three-dimensional (3D) imaging technologies for crime scene reconstruction. Australian Journal of Forensic Sciences, vol. 50, no. 6, pp. 697–707 (2018) doi: 10.1080/00450618.2018.1424245
10. Sanyal, S., Bolkart, T., Feng, H., Black, M.: Learning to regress 3D face shape and expression from an image without 3D supervision. In: Proceedings IEEE Conference on Computer Vision and Pattern Recognition, pp. 7763–7772 (2019) doi: 10.1109/CVPR.2019.00795
11. Sevastopolsky, A., Ignatiev, S., Ferrer, G., Burnaev, E., Lempitsky, V.: Relightable 3D head portraits from a smartphone video (2020)
12. Slater, M., Cabriera, C., Senel, G., Banakou, D., Beacco, A., Oliva, R., Gallego, J.: The sentiment of a virtual rock concert. Virtual Reality, pp. 1–25 (2022) doi: 10.1007/s10055-022-00685-9
13. Stern, G., Fu, Z., Ardabilian, M.: 3D face analysis for healthcare. Biometrics under Biomedical Considerations, pp. 147–160 (2019) doi: 10.1007/978-981-13-1144-4_6
14. Tran, A. T., Hassner, T., Masi, I., Paz, E., Nirkin, Y., Medioni, G.: Extreme 3D face reconstruction: Seeing through occlusions. In: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 3935–3944 (2018) doi: 10.1109/CVPR.2018.00414
15. Vilchis, C., Perez-Guerrero, C., Mendez-Ruiz, M., Gonzalez-Mendoza, M.: A survey on the pipeline evolution of facial capture and tracking for digital humans. Multimedia Systems, pp. 1–24 (2023) doi: 10.1007/s00530-023-01081-2
16. Vladimirov, I., Nenova, M., Nikolova, D., Terneva, Z.: Security and privacy protection obstacles with 3D reconstructed models of people in applications and the metaverse: A survey. In: 2022 57th International Scientific Conference on Information, Communication and Energy Systems and Technologies (ICEST), pp. 1–4 (2022) doi: 10.1109/ICEST55168.2022.9828791
17. Wang, Y., Su, Z., Zhang, N., Xing, R., Liu, D., Luan, T. H., Shen, X.: A survey on metaverse: Fundamentals, security, and privacy. IEEE Communications Surveys & Tutorials, vol. 25, no. 1, pp. 319–352 (2022) doi: 10.1109/COMST.2022.3202047
18. Wu, C. Y., Xu, Q., Neumann, U.: Synergy between 3DMM and 3D landmarks for accurate 3D facial geometry. In: 2021 International Conference on 3D Vision (3DV), pp. 453–463 (2021) doi: 10.1109/3DV53792.2021.00055
19. Zielonka, W., Bolkart, T., Thies, J.: Towards metrical reconstruction of human faces. In: Computer Vision- European Conference on Computer Vision, vol. 13673, pp. 250–269 (2022) doi: 10.1007/978-3-031-19778-9_15
20. Zollhöfer, M., Thies, J., Garrido, P., Bradley, D., Beeler, T., Pérez, P., Stamminger, M., Nießner, M., Theobalt, C.: State of the art on monocular 3D face reconstruction, tracking, and applications. In: Computer Graphics Forum, vol. 37, pp. 523–550 (2018) doi: 10.1111/cgf.13382

ISSN 1870-4069

Derecho comparado asistido por computadora: Procesamiento de lenguaje natural legal

Héctor Alejandro Vargas-Gutiérrez[1], Gerardo Rodriguez-Hernandez[2]

[1] Universidad de Guadalajara,
Centro Universitario de Ciencias Económico Administrativas,
México

[2] Tecnológico de Monterrey,
Escuela de Ingeniería y Ciencias,
México

hecvagu@hotmail.com, gerardo.rodriguez@tec.mx

Resumen. El sistema legal Mexicano cuenta con un amplio conjunto de leyes y regulaciones. Sin embargo, el crecimiento constante del marco normativo ha generado dificultades en el análisis y control de las normas debido a la redacción diversa realizada por diferentes autoridades legislativas, dando lugar a la existencia de legislaciones semánticamente similares pero lexicamente distintas. Este artículo propone un método basado en procesamiento de lenguaje natural o PLN (*Natural Language Processing* NLP por sus siglas en ingles) para ayudar al sistema jurídico mexicano, en particular al derecho comparado, mediante la identificación de expresiones con significados semánticamente similares en las constituciones de los 32 estados mexicanos. Se utilizaron cuatro diferentes transformaciones de texto en espacios vectoriales (*embedding*) (count vectorizer, tf idf, word2vec, BERT) y tres enfoques de agrupamiento (*clustering*) (k medias, GMM, agrupamiento aglomerativo) para el procesamiento de los artículos constitucionales. Se evaluaron diferentes casos de prueba y se observó que la representación en el espacio vectorial Word2Vec ofreció los mejores resultados en general, independientemente del método de agrupamiento utilizado. Sin embargo la combinación puntual que obtuvo el mejor resultado fue BERT con k-medias.

Palabras clave: PLN, procesamiento lenguaje natural legal, agrupamiento, derecho comparado.

Computer-Assisted Comparative LAW: Legal Natural Language Processing

Abstract. The Mexican legal system has a broad set of laws and regulations. However, the growth of the regulatory framework in recent years has generated difficulties in the analysis and control of norms due to the diverse wording used by different authorities, which can generate confusion and legal conflicts. This article proposes a procedure based on natural language processing (NLP) to assist the Mexican legal system, particularly comparative law, by identifying expressions

with semantically similar meanings in the constitutions of the 32 Mexican states. Text transformations in vector spaces and clustering approaches were suggested for the processing of constitutional articles. Laws were collected and preprocessed, and representation techniques in vector spaces, such as Word2Vec, were applied for distance-based clustering. Different test cases were evaluated, and it was observed that the Word2Vec embedding offered the best overall results, regardless of the clustering method used. The best-performing combination was BERT with k-means to 22 groups.

Keywords: NLP, legal natural lenguage processing, clustering, comparative law.

1. Introducción

El derecho regula múltiples sectores de la sociedad, lo cual implica el análisis y creación de grandes cantidades de texto, así como la interpretación del lenguaje legal. En el caso del sistema legal mexicano, mantenerse actualizado en la interpretación de textos complejos supone un gran esfuerzo debido a las numerosas leyes vigentes. El procesamiento de lenguaje natural (PLN) se aplica en diferentes contextos legales para mejorar la eficiencia y automatizar tareas rutinarias [10, 11, 6].

El Derecho Comparado, se enfoca en el estudio de los sistemas legales de distintos ordenamientos jurídicos, de manera comparativa, a fin de identificar las mejores prácticas y soluciones, y mejorar el propio sistema legal. Los abogados y juristas pueden aplicar estos conocimientos para mejorar la calidad y eficacia de la justicia [5].

El sistema legal de México se caracteriza por una amplia colección de leyes y regulaciones, lo que dificulta su análisis y control, generando posibles conflictos legales [4, 9]. Proponemos un modelo que tiene el potencial de ayudar en la labor del derecho comparado a identificar los artículos entre las constituciones de todos los estados más la Constitución Política de los Estados Unidos Mexicanos (CPEUM), que se refieren a un mismo tema, aún cuando sean escritos con diferentes términos.

Este problema de similitud de textos se puede resolver como un problema geométrico de distancias en espacios vectoriales como lo hizo Shahmirzadi, Lugowski y Younge, en su trabajo: Modelos de similitud de textos en espacios vectoriales: un estudio comparativo, donde miden la similitud de patentes con variaciones del modelo de representación en espacios vectorialesTF IDF [12].

Además otros autores como Arnarsson, Frost, Gustavsson, Jirstrand y Malmqvist proponen en su trabajo: Métodos de PLN para la gestión del conocimiento: aplicación agrupamiento de documentos para una búsqueda y agrupación rápidas de documentos de ingeniería, otras transformaciones de texto a espacios vectoriales como doc2vec y para agrupación el modelo asignación latente de Dirichlet (*Latent Dirichlet allocation* LDA) para identificar documentos relacionados en una base de datos de solicitudes de cambios de ingeniería, donde con ayuda de un experto corporativo examinan si los informes pertenecen o no a los grupos generados [8].

En este trabajo se comparan modelos con cuatro métodos de representación en espacios vectoriales y tres métodos de agrupamiento, los cuales son evaluados con

Fig. 1. Distribución de la cantidad de palabras por artículo en el corpus antes y después de limitar la cantidad máxima a 500 palabras.

una métrica que integra cuatro casos de prueba que son proporcionados por un experto en el área legal y otros obtenidos de un tesauro de la suprema corte de justicia de la nación (SCJN).

El objetivo principal de este estudio es explorar los métodos de PLN que mejor describan la similaridad semántica de los artículos de la ley mexicana con respecto a los casos de prueba y compararlos a través de una métrica especializada sencilla para descubrir el modelo que mejor desempeño tiene en esta tarea.

2. Metodología

Como una posible solución al problema del derecho comparado a través del PLN se propone la agrupación de los artículos por sus significado semántico, para lograrlo se transforma el problema de trabajar con textos y palabras a trabajar en espacios vectoriales, agrupando estos artículos de las constituciones con base en su cercanía semántica a través de estas representaciones en espacios vectoriales y su cercanía geométrica, esto quiere decir que dentro de estos grupos se pueden encontrar los artículos que se refieren a temas similares, a pesar de que estos estén expresados con diferentes palabras. El proceso detallado de la solución se describe a continuación:

2.1. Elaboración del corpus de datos

Para el presente estudio de PLN aplicado al problema legal planteado, se recopilaron las 32 constituciones políticas de los estados de México además de la CPEUM, formando un corpus con los artículos que integran estas 33 constituciones las cuales fueron descargadas de las páginas oficiales de los congresos de cada estado incluyendo la CPEUM que fue descargada desde la página oficial de la SCJN.

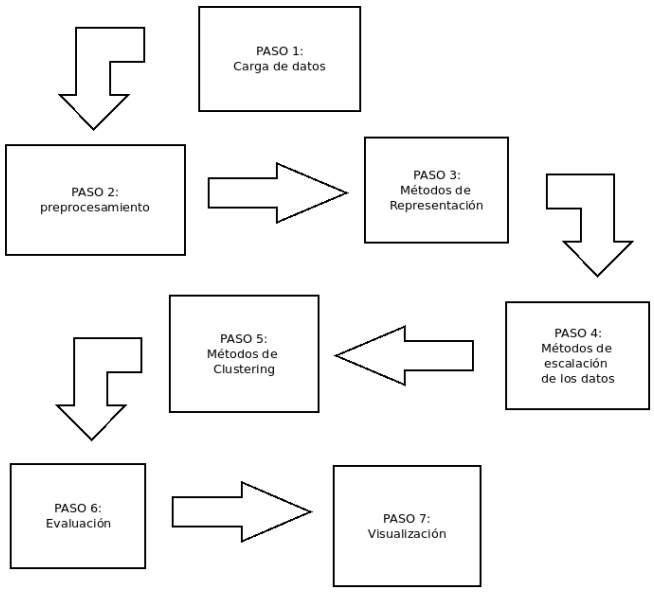


Fig. 2. Clases del modelo de implementación.

En la figura 1 se muestra la longitud de cada artículo medida por el número de palabras contenida en cada uno; La gráfica derecha muestra una amplia variación en el tamaño de los artículos originales, llegando algunos a exceder el tamaño 2300 palabras. En el corpus original el recuento de artículos es de 5051 mientras que el corpus de trabajo con el limite máximo de 500 palabras representado en la gráfica izquierda de la figura 1 es de 14047.

El umbral de 500 palabras se definió como longitud máxima, debido a que es el límite de entrada que acepta uno de los modelos de representación utilizados en este estudio. Por este motivo, se realizó un primer pre-procesamiento o curado de los documentos utilizando una combinación de expresiones regulares y separación semi automática, debido a que no existe un estándar o una norma que regule la forma de construir los artículos entre las constituciones de los estados de la república mexicana.

La separación de estos artículos se hizo tomando en cuenta la jerarquía en las numeraciones de cada constitución, en la mayor parte de estas constituciones la jerarquía más alta la tienen los apartados usualmente en letras del alfabeto romano en mayúsculas, en seguida los listados con números romanos, y al final los listados con letras del alfabeto romano en minúsculas o en números arábigos. Todos estos con variantes de punto, punto y guión, punto y coma, paréntesis, espacio o nada.

2.2. El sistema

Se eligió Python como lenguaje de programación para los experimentos realizados en este estudio porque es ampliamente utilizado por la comunidad científica en campos como PLN, IA y Macro datos (*Big Data*), y cuenta con una amplia comunidad de soporte.

Se utilizaron herramientas como numpy, scikit-learn, pandas y matplotlib. Para garantizar la escalabilidad y el uso constante, se utilizó Apache Spark versión 3.0.1 por su capacidad de procesar grandes volúmenes de datos de manera distribuida y escalable.

En la Figura 2 se muestra el proceso del flujo de trabajo que se siguió para generar un modelo en el que sus partes más importantes en el proceso son el Paso 3 que se refiere a los métodos en el cual se transforman los textos a espacios vectoriales y el Paso 5 donde se forman los grupos con técnicas de agrupamiento midiendo las distancias geométricas de los vectores producidos desde el Paso 3.

El proceso ilustrado en la Figura 2 abarca la combinación de todas las técnicas que se incluyen en cada paso del proceso. Se produjeron 4320 modelos que se ejecutaron una sola vez cada uno y de los cuales es desprenden los resultados.

2.3. Carga de los datos y de los casos de prueba

Ya con los artículos pre-procesados, para cumplir con los criterios de tamaño máximo se procede a cargarlos en la memoria junto con los casos de prueba para que sean procesados, transformados y agrupados al mismo tiempo.

2.4. Pre-procesamiento

El siguiente paso es el pre-procesamiento, aquí se intercambian todas las letras mayúsculas por minúsculas seguido por una etapa de limpieza en la que se eliminan las palabras de paro (*stop words*), que son palabras que se denomina que no agregan valor al sentido del texto, esto facilita la transformación de texto en vectores numéricos y su manipulación en pasos subsecuentes.

Después de la limpieza de los artículos, sus textos se transforman en listas de palabras, a esto se le conoce como simbolización (*tokenization*) pues a cada palabra en la lista se le denomina símbolo (*token*).

A continuación se genera una serie de N-gramas de 2 y 3 palabras, esto con la intención de que las secuencias de palabras que sean más frecuentes y que usualmente representan conceptos de 2 o tres palabras, puedan ser capturadas también como un solo término.

2.5. Representación de textos en espacios vectoriales

Después que los artículos se convierten en símbolos, pueden ser procesados en la siguiente fase del modelo, en la cual incluimos cuatro aproximaciones recientemente utilizadas en la literatura. Estos métodos transforman las listas de símbolos contenidas en cada artículo en una representación de vectores numéricos que se mencionan a continuación:

CountVectorizer. El modelo *count vectorizer* (también llamado vectorizador de la frecuencia de los términos (*term frequency vectorizer*) o TF), es un modelo basado en la técnica de representación bolsa de palabras (*bag-of-words*), lo que implica que no se tiene en cuenta la información sobre la posición de los símbolos ni su contexto. En lugar de ello, se enfoca únicamente en la presencia y frecuencia de cada palabra en la colección de documentos.

En otras palabras registra el número de veces que aparece cada palabra de un vocabulario en un documento. Por esta razón, *countVectorizer* es especialmente útil en tareas de PLN donde el contexto y la posición de los símbolos no son esenciales para la tarea en cuestión, como la detección de similitudes entre documentos o la agrupación de documentos por temas [1].

En este modelo, el parámetro utilizado es el ”tamaño del vocabulario”(*vocab size*), que se refiere al número máximo de palabras que el modelo *CountVectorizer* crea en su vocabulario. Este vocabulario se forma a partir de los términos más importantes, ordenados por frecuencia de aparición en todo el corpus de texto, y se limita al tamaño especificado por el parámetro *vocab_size*. En este estudio se limita la búsqueda de modelos a la variación del parámetro *vocab_size* de tamaño 200, 400 y 800.

TF_IDF. TFIDF es un acrónimo que hace referencia a “frecuencia del término por su inverso en la frecuencia de los documentos” por sus siglas en inglés (*Term Frequency-Inverse Document Frequency*).

Esta técnica se utiliza para establecer una proporción entre la frecuencia de un término en un documento específico y su frecuencia en todo el corpus de documentos. De esta manera, si un término aparece con frecuencia en un documento pero no en los demás, su valor TFIDF será alto, lo que indica que es un término relevante y distintivo para ese documento.

El objetivo de la técnica TFIDF es identificar los términos clave de un documento en relación con el corpus completo. Se espera que los términos clave aparezcan con una frecuencia mayor en el documento en cuestión y con una frecuencia menor en el corpus. De esta forma, los términos que aparecen con mayor frecuencia en el documento tienen un valor TFIDF más alto y se consideran más importantes que los términos que aparecen con frecuencia similar en todos los documentos [13].

La técnica TFIDF es muy útil en el procesamiento de lenguaje natural en la clasificación de textos, ya que ayuda a identificar los términos más relevantes en un conjunto de documentos.

El parámetro *numFeatures* en este modelo determina la dimensión del vector de características. Un valor más alto para *numFeatures* dará como resultado un vector de características de mayor dimensión, lo que potencialmente capturará información más detallada de los datos de texto, pero también aumentará la complejidad computacional y el uso de la memoria.

Por otro lado, un valor más bajo para *numFeatures* dará como resultado un vector de características de menor dimensión, lo que podría perder parte de la información de los datos de texto pero también reducir la sobrecarga computacional. En este estudio se limita la búsqueda de modelos a la variación del parámetro *numFeatures* de tamaño 200, 400 y 800.

Word2vec. Word2vec es un conjunto de modelos de redes neuronales que se caracterizan por ser poco profundos (bolsa de palabras continua (*Continuous Bag of Words* (CBOW) y salta grama *Skip-Gram*). Las palabras que comparten contextos comunes en el corpus están ubicadas cerca unas de otras en el espacio vectorial. Esto significa que las palabras que tienen significados similares se encuentran en la misma zona del espacio vectorial [7].

Es importante destacar que los modelos de Word2vec no solo tienen en cuenta la frecuencia de las palabras en el corpus, sino que también consideran su contexto. De esta manera, las palabras que aparecen juntas con frecuencia en el corpus son codificadas en vectores cercanos en el espacio vectorial, lo que permite que las similitudes semánticas se reflejen en la ubicación de las palabras en el espacio.

El parámetro *vector_size* en el modelo Word2Vec determina el número de dimensiones en los vectores de palabras aprendidos. Un valor más alto para *vector_size* dará como resultado vectores de palabras de mayor dimensión, capturando potencialmente relaciones semánticas más matizadas entre palabras, pero también aumentando la complejidad computacional y el uso de memoria.

Por otro lado, un valor más bajo para *vector_size* dará como resultado vectores de palabras de dimensiones más bajas, lo que podría reducir la capacidad de capturar información semántica detallada pero también reducir la sobrecarga computacional.En este estudio se limita la búsqueda de modelos a la variación del parámetro *vector_size* de tamaño 200, 400 y 800.

BERT Se usó el modelo BETO, el cual es una variante del modelo BERT (*Bidirectional Encoder Representations from Transformers*) o Representación de Codificador Bidireccional desde Transformadores entrenado con un gran corpus en español.

BETO es de un tamaño similar al modelo BERT-base y fue entrenado con la técnica de enmascaramiento de todas las palabras (*Whole Word Masking technique* WWM) que toma en cuenta todas las sub palabras como prefijos o sufijos o variantes de la misma raíz como una sola palabra para que el significado original se mantenga durante todo el entrenamiento [3, 2].

En este estudio se limita la búsqueda de modelos a la variación del parámetro *max_seq_length* de tamaño 500, el limite máximo para este parámetro es de 516. El parámetro *max_seq_length* establece un límite superior en el número de símbolos que pueden estar presentes en una secuencia de entrada. Si una secuencia excede este límite, debe truncarse u omitirse para ajustarse a la longitud especificada.

Las secuencias que son más cortas que la longitud máxima especificada generalmente se rellenan con símbolos especiales para garantizar la uniformidad en los datos de entrada. Secuencias más largas pueden contener más contexto y proporcionar información más rica para el modelo, pero también aumentan la complejidad computacional y el uso de la memoria. Por otro lado, las secuencias más cortas pueden provocar la pérdida de información contextual importante.

2.6. Escalación y normalización

Una vez que los artículos fueron transformados a sus representaciones en espacios vectoriales, mediante los métodos ya mencionados, el siguiente paso consistió en seleccionar un método de escalado o normalización de entre los cuales se probaron escala mínima-máxima, estandarización y ninguno.

2.7. Agrupamiento

El siguiente paso consistió en formar los grupos y para cada uno de ellos se realizó la búsqueda dentro de los siguientes parámetros: $numb\_clusters\_s = [20, 22, 24, 26, 28, 30, 32, 34, 36]$. Se eligió este rango en función de la cantidad de temas presentes en la CPEUM.

La CPEUM consta de 136 artículos distribuidos en nueve títulos, estos títulos a su vez se dividen en capítulos, sumando 16 secciones en total, más aquellos que deben ser clasificados como derogados y otras categorías que pudieran estar presentes se eligieron 20 grupos como mínimo para empezar la búsqueda de los modelos.

Para realizar el agrupamiento de los artículos en su representación vectorial, se utilizaron los siguientes métodos:

Distribuciones gaussianas mixtas. El método de Distribuciones gaussianas mixtas (*Gaussian Mixture Models* GMM) en cuestión no es compatible con el proceso de estandarización llevado a cabo en el paso anterior, ya que está diseñado para trabajar con las distribuciones de los espacios vectoriales de los artículos, y la estandarización provoca una pérdida de la información fundamental con la que opera este algoritmo, por esta razón solo se corrieron modelos con escalación de datos máxima y mínima y modelos sin ningún tipo de escalación o normalización.

El parámetro "tipos de covarianza" o *cov_types* en un modelo GMM especifica el tipo de matriz de covarianza que se utiliza para modelar la estructura de covarianza de las distribuciones gaussianas en la mezcla. La matriz de covarianza representa la covarianza o correlación entre diferentes características o dimensiones de los datos. para este método, se utilizó el parámetro *cov_types* de tipo *full*, *diag* y *spherical*.

El *cov_types* de tipo completo (*full*) asume que cada componente gaussiano en la mezcla tiene su propia matriz de covarianza completa, que puede capturar correlaciones arbitrarias entre diferentes características o dimensiones de los datos. Sin embargo, esto puede ser computacional-mente costoso y puede requerir una gran cantidad de parámetros para estimar.

El *cov_types* de tipo *diag* asume que cada componente gaussiano en la mezcla tiene su propia matriz de covarianza diagonal, que solo modela las variaciones de las características individuales sin capturar ninguna correlación entre ellas. Esto puede ser computacional-mente eficiente y puede funcionar bien para datos con poca o ninguna correlación entre características.

El *cov_types* de tipo esférico (*spherical*) asume que cada componente gaussiano en la mezcla tiene su propia matriz de covarianza esférica, que es una matriz de identidad escalada. Esto significa que todas las características tienen la misma varianza dentro de cada componente, sin capturar ninguna correlación entre ellas. Esto puede ser útil para datos con varianzas isotrópicas o similares en todas las entidades.

Agrupamiento aglomerante. Para el modelo de agrupamiento aglomerante *Agglomerative clustering* se utilizaron los parámetros de afinidad (*affinity*) y de enlace (*linkage*); para el parámetro de afinidad se utilizó la métrica euclidiana y para el método de enlace o encadenamiento se utilizó la estrategia de aglomeración de pabellón (*ward*). El parámetro de "afinidad" *aff* en un modelo de agrupamiento aglomerante determina el método utilizado para calcular las distancias por pares o las similitudes entre los puntos de datos.

Euclidean calcula la distancia euclidiana entre pares de puntos de datos, que es la distancia en línea recta entre dos puntos en el espacio euclidiano. Es la opción predeterminada en muchos algoritmos de agrupamiento aglomerativo y es adecuada para datos con características numéricas continuas.

El parámetro de "vinculación" o enlace en un modelo de agrupamiento aglomerativo especifica el método utilizado para calcular la diferencia o similitud por pares entre los grupos al fusionarlos. El método *ward* calcula el aumento en la suma de las diferencias al cuadrado (SSD) dentro de los grupos al fusionarlos. Tiende a producir grupos más compactos y esféricos, y es adecuado para datos con características numéricas continuas.

K medias. Para este método utiliza el parámetro *numIterations_s* con un valor de 100. El algoritmo K medias comienza inicializando aleatoriamente K centroides, donde K es el número de grupos especificado. Luego, alterna entre dos pasos:

Paso 1: Asignación de cada punto de datos al centroide más cercano en función de la distancia euclidiana u otra métrica de distancia.

Paso 2: Actualización de los centroides calculando la media de todos los puntos de datos asignados a cada centroide.

Estos dos pasos se repiten iterativamente hasta que se cumple un criterio de convergencia, que es el número máximo de iteraciones o un umbral de tolerancia en el cambio d e centroides. El parámetro "número de iteraciones" o *numIteration_s* en un modelo de agrupamiento de K-medias determina el número máximo de veces que el algoritmo realizará los pasos de asignación y actualización antes de detenerse, independientemente de si los centroides han convergido o no. Si el algoritmo alcanza el número máximo de iteraciones antes de converger, se detendrá y devolverá las asignaciones de grupo y los centroides actuales como resultado final.

2.8. Evaluación

Elaboración de los casos de pruebas

Los casos de prueba se obtuvieron de un experto en la materia legal y de un Tesauro jurídico de la SCJN. Son conjuntos de palabras que se sustituyen fácilmente a lo largo de todas las constituciones y que no cambian el significado de los artículos que los contienen.

Así el caso de prueba 1 hace referencia a grupos originarios; grupo étnico; grupos étnicos; indígenas; pueblos indígenas; pueblos tribales; pueblos indios; Poblaciones Indígenas y Tribales, etc... El caso de prueba 2 a la libre determinación; autonomía; auto regulación; etc... El caso de prueba 3 a la Acta Constitucional; Carta constitucional; Carta federal; Carta magna; Código fundamental; etc... Y el caso de Prueba 4 a Derechos de la persona humana; Derechos del hombre; Derechos esenciales del hombre; Derechos implícitos; etc...

Métrica de evaluación

Para evaluar los modelos, se utilizó una métrica especializada (*ad-hoc*) a la que se denominó "*recall* integrado".

El funcionamiento de esta métrica requiere saber qué artículos contienen una o varias palabras del caso de prueba a las que definimos como palabras clave.

Se escogió una variación de la métrica "*recall*" porque solo sabemos cuales artículos contienen palabras clave, por lo que los elementos relevantes son los verdaderos positivos (*True Positives* TP) que representan el grupo que contienen el caso de prueba y los artículos que tienen por lo menos una de las palabras clave y los falsos negativos (*False Negatives* FN) que representan los grupos que contienen artículos con alguna de las palabras clave pero no el caso de prueba.

Para lograrlo se resolvieron los siguientes pasos en orden:

Paso 1: En este paso se agregó una columna con una bandera que nos ayudó a identificar los artículos que contenían por lo menos una de las palabras clave y comparar y verificar que por lo menos los textos que contenían estas palabras pudieran ser agrupados en el mismo grupo y los que no se agrupaban donde mismo poder encontrar las diferencias en el contexto del artículo.

Paso 2: Se utilizó el artículo personalizado que contenía las palabras clave del caso de prueba para determinar el agrupamiento correcto al que debían pertenecer los artículos semánticamente similares y se descartaron los demás grupos.

Paso 3: Se compararon los resultados de los pasos 1 y 2 y se obtuvieron las dos categorías que se necesitan para calcular la métrica de evaluación a partir de los resultados de los modelos de agrupación verdaderos positivos TP y falsos negativos FN.

Paso 4: Se separaron los modelos según sus características de composición similares como sigue:

1. método de representación de texto en espacios vectoriales y
2. método de agrupamiento.

Paso 5: Se utilizaron los valores de TP y FN para calcular la métrica especializada "*recall* integrado" sobre los modelos que difieren en el caso de prueba.

La función "*recall* integrado" se calculó de acuerdo a la siguiente ecuación 1:

$$recall\_integrado = \frac{\sum_i^n tp_i}{\sum_i^n tp_i + \sum_i^n fn_i}, \tag{1}$$

donde:

$\sum_i^n tp_i$ = es la sumatoria de todos los verdaderos positivos
$\sum_i^n fn_i$ = es la sumatoria de todos los falsos negativos

Ambas sumatorias se calcularon sobre los modelos similares en composición, esto es, los que se componen por el mismo algoritmo de representación y de agrupamiento, incluyendo todos los modelos que utilizan cada uno de los casos de prueba.

2.9. Visualización

Por último se graficaron los resultados de la evaluación con la métrica anterior en cada punto contra el parámetro cantidad de grupos utilizado, y se mostraron las distribuciones que se obtuvieron por cada modelo entrenado.

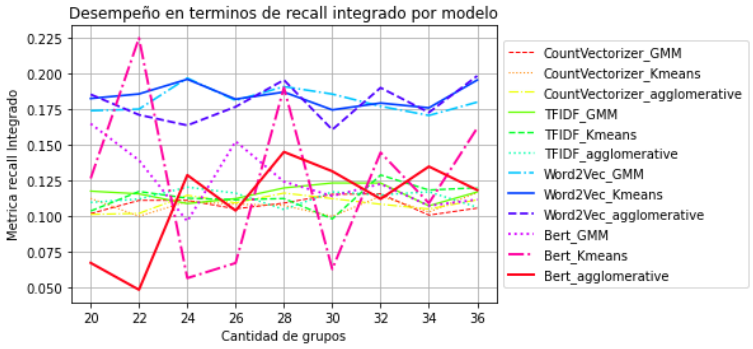


Fig. 3. Desempeño de los modelos en términos de "*recall* integrado".

3. Resultados

El desempeño de cada modelo se puede observar en la figura 3, donde los modelos con mejores comportamientos en general son los que se entrenaron con un representación en espacios vectoriales *Word2vec* sin embargo el modelo que tuvo el mejor desempeño individual fue el entrenado con un representación en espacios vectoriales BETO y con el modelo de agrupamiento de distribuciones gaussianas mixtas.

En la figura 4 se puede observar mejor las distribuciones de los comportamientos de los modelos, agrupados por el algoritmo de representación en espacios vectoriales utilizado, donde los modelos relacionados al *count vectorizer* en promedio tuvieron un desempeño de 0.110 , la representación tf idf tuvo en promedio un desempeño de 0.120, la representación BERT en promedio 0.124 y la representación word2vec 0.180 con respecto a la métrica "*recall* integrado". De éstos números se deriva que los modelos de representación que se calculan a partir de redes neuronales, es decir, BERT y *word2vec*, son los que obtienen un mejor desempeño.

A pesar que el modelo que obtuvo individualmente el mejor resultado utiliza BERT para la representación en el espacio vectorial, la dispersión de los resultados obtenidos por el resto de los modelos que también utilizan BERT es la mayor de todos los algoritmos de representación utilizados, esto significa que el desempeño obtenido no es consistente.

Se considera que los modelos que utilizan word2vec como algoritmo de representación tuvieron en general una baja dispersión y un mejor desempeño que los demás en términos de recall integrado", por esta razón se concluye que este algoritmo es más consistente que el resto de los modelos comparados en este trabajo.

4. Trabajo futuro

Se cree que los resultados obtenidos pueden ser aplicables a otros corpus de normas, reglamentos o materias legales y este trabajo busca establecer un precedente en el campo ya que, hasta donde sabemos, no existen estudios similares en la literatura en el ámbito legal.

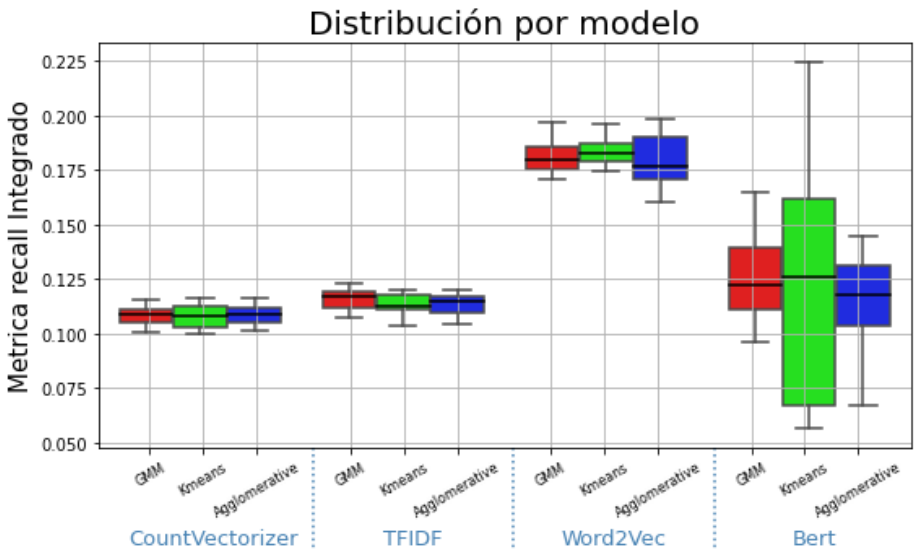


Fig. 4. Distribución de los modelos en términos de "*recall* integrado".

Es importante tener en cuenta que el alcance de este trabajo se limitó a la selección de los métodos previamente mostrados, debido a su uso demostrado en trabajos anteriores con problemáticas similares y a la complejidad y costo computacional de implementar modelos con una mayor cantidad de parámetros.

Este trabajo se puede extender en el futuro como una aplicación que realice una búsqueda por término para obtener los artículos específicos que se relacionan al mismo, también se pude extender y mejorar la cantidad de casos de prueba, además de evaluar los resultados de los métodos de agrupación con métricas como el índice de Calinski-Harabasz o el índice de Davies-Bouldin que miden la cercanía entre los grupos generados.

Se pueden explorar otras métricas de similitud para agrupamiento de los textos. Otra posibilidad es ejecutar varias veces todos los modelos y realizar los cálculos de nivel de significancia de los resultados de los agrupamientos.

Una limitación de la aplicación de estos modelos es la dependencia de un experto en el área legal para la generación de casos de prueba y para la interpretación de los resultados debido a la gran cantidad de casos posibles de los cuales solo se utilizaron cuatro en este trabajo para la métrica de evaluación, por esta razón se propone el desarrollo modelos de aprendizaje de tipo no supervisado.

5. Conclusiones

En este estudio, se desarrollaron modelos que identifican textos semánticamente similares utilizando técnicas de procesamiento del lenguaje natural (PLN) para ayudar en el problema del derecho comparado.

Se probaron cuatro técnicas de representación en espacios vectoriales (count vectorizer, TF-IDF, word2vec y BERT) combinadas con tres métodos de agrupamiento (GMM, agrupamiento aglomerante y k-medias).

Se aplicaron estas técnicas a un corpus legal compuesto por las constituciones de todos los estados y la constitución federal, y se curó de forma semi-automática. El corpus resultante consta de 14047 documentos.

Para evaluar los resultados, se utilizaron casos de prueba de diferentes términos legales semánticamente similares y una métrica ad-hoc llamada recall integrado". El modelo individual obtuvo el mejor resultado fue el formado por la técnica de representación en espacios vectoriales BERT en combinación con el método de agrupamiento k-medias.

Sin embargo, debido a la dispersión que tuvieron los otros modelos que utilizaron BERT como técnica de representación no se escoge como el algoritmo con mejor desempeño por su inconsistencia.

No obstante, Los modelos formados por la técnica de representación en espacios vectoriales word2vec tuvieron una menor dispersión y en promedio los mejores resultados independientemente del método de agrupamiento utilizado.

Referencias

1. Basarkar, A.: Document classification using machine learning. Ph. D. thesis, San Jose State University, pp. 1–56 (2017) doi: 10.31979/etd.6jmu-9xdt
2. Cañete, J., Chaperon, G., Fuentes, R., Ho, J. H., Kang, H., Pérez, J.: Spanish pre-trained BERT model and evaluation data. In: PML4DC at ICLR 2020 (2020)
3. Devlin, J., Chang, M. W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 4171–4186 (2019) doi: 10.18653/v1/n19-1423
4. Fix-Fierro, H.: ¿Por qué se reforma tanto la constitución mexicana de 1917? Hacia la renovación del texto y la cultura de la constitución. Cien Ensayos para el Centenario, vol. 4, pp. 143–162 (2017)
5. Flores-Sánchez, J. A., Maglioni-Montalvo, R.: Derecho comparado en investigación y enseñanza de derecho en carreras económico administrativas. Hitos de Ciencias Económico Administrativas, vol. 25, no. 71, pp. 136–147 (2020) doi: 10.19136/hitos.a25n71.3608
6. Katz, D. M., Hartung, D., Gerlach, L., Jana, A., Bommarito, M. J.: Natural language processing in the legal domain. SSRN, pp. 1–13 (2023) doi: 10.2139/ssrn.4336224
7. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: 1st International Conference on Learning Representations, ICLR (Workshop Poster) (2013) doi: 10.48550/arXiv.1301.3781
8. Örn Arnarsson, I., Frost, O., Gustavsson, E., Jirstrand, M., Malmqvist, J.: Natural language processing methods for knowledge management—applying document clustering for fast search and grouping of engineering documents. Concurrent Engineering Research and Applications, vol. 29, no. 3, pp. 142–152 (2021) doi: 10.1177/1063293x20982973
9. Risco, J.: Los malditos vacíos legales. El financiero (2017)
10. Romero-Pérez, J. E.: Notas sobre la interpretación jurídica. Revista de Ciencias Jurídicas, vol. 1, no. 133, pp. 79–102 (2014)
11. Secretaría de Servicios Parlamentarios: Leyes federales vigentes (2020) www.diputados.gob.mx/LeyesBiblio/index.htm

12. Shahmirzadi, O., Lugowski, A., Younge, K.: Text similarity in vector space models: A comparative study. In: 2019 18th IEEE International Conference On Machine Learning And Applications (ICMLA), pp. 659–666 (2018) doi: 10.1109/ICMLA.2019.00120
13. Siregar, A. M., Faisal, S., Tukino, A. P., Simarangkir, M. S.: Comparison study of term weighting optimally with svm in sentiment analysis. In: Proceedings of The 2nd International Conference On Advance And Scientific Innovation (2019) doi: 10.4108/eai.18-7-2019.2288508

ISSN 1870-4069

Análisis y clasificación de tweets contextuales de desastres

Tania Alcántara, Omar García-Vázquez,
Hiram Calvo, Marco A. Cardoso-Moreno

Instituto Politécnico Nacional,
Centro de Investigación en Computación,
Laboratorio de Ciencias Cognitivas Computacionales,
México

{talcantaram2020, hcalvo,
mcardosom2021}@cic.ipn.mx, omar.gava@hotmail.com

Resumen. El despliegue de información de los desastres ha ido cambiando, hoy en día esta divulgación de información se hace de manera más rápida a través de las redes sociales, especialmente por Twitter, pero, ya que Twitter es completamente abierto, hace que el posteo de información lo haga cualquier usuario y la confianza se pierda. Este trabajo se centra en el análisis y clasificación de un conjunto de textos etiquetados como desastre y no desastre, donde los etiquetados como no desastre incluyen contexto metafórico. La clasificación va centrada en modelos clásicos, como Random Forest, Support Vector Machine, Decision Tree y XGBOOST, esto con los extractores de características SentenceBert y n-gramas y así, determinar la importancia de la selección de características para encontrar relaciones entre los textos y la importancia de las palabras.

Palabras clave: Aprendizaje automático, clasificación, procesamiento de lenguaje natural, N-gramas, SentenceBERT, desastres

Analysis and Classification of Contextual Disaster Tweets

Abstract. The shipping of information on disasters has changed over time, nowadays, social networks allow for faster information disclosure about these topics, specially Twitter; since this platform is completely open, i.e., there is little to none restriction on what a user can post, hence, creating a lack of confidence and trust on the information available. This paper focuses on the analysis and classification of a set of texts labeled as disaster and non-disaster, where those labeled as non-disaster include metaphorical context. The classification is focused on classical models, such as Random Forest, SVM, Decision Trees and XGBOOST with different extractors, with the help of Sentence Bert feature extraction and n-grams, and thus determine the importance of feature selection to find relationships between texts and the importance of words.

Keywords: Machine learning, classification, natural language processing, N-grams, SentenceBERT, disasters.

1. Introducción

La huella del cambio climático ha impactado en la forma en la que nuestro planeta se comporta y esto ha provocado un aumento de las catástrofes. De acuerdo a la Organización Meteorológica Mundial (OMM), los desastres naturales han aumentado en los últimos 50 años, además el Atlas de Mortalidad y Pérdidas Económicas por Fenómenos Meteorológicos, Climáticos e Hídricos de los organismos, reportan que entre 1970 y 2019, los desastres naturales representan el 50 % de todos los desastres, el 74 % de las perdidas económicas y el 45 % de las muertes [1].

Así mismo, y derivado de la forma en la que vivimos, no solo encontramos desastres naturales, sino humanos, como choques automovilísticos, incendios provocados, entre algunos otros.

Hace 30 años, cuando un evento de esa magnitud ocurría, la información era desplegada principalmente a través de los medios tradicionales, como la televisión, la radio, y en algunos casos, debido a los plazos para la impresión de papel, era necesario esperar hasta el siguiente día para conocer la información a través de periódicos y revistas.

Además, los medios oficiales del gobierno lanzaban comunicados donde se indicaban las acciones a seguir, la ayuda y los datos de más relevancia. La única manera en que la gente cercana al problema pudiera dar información, era a través de la radio por medio de los radioaficionados, los cuales proporcionaban servicios de comunicación críticos en momentos de crisis [2].

Hoy en día, derivado de los alcances tecnológicos, los radioaficionados han evolucionado en el entorno de las redes sociales, las cuales se han convertido en una nueva herramienta de comunicación. Una red social muy usada en este tipo de casos es Twitter, en donde con tal solo crear una cuenta un usuario puede compartir todo lo que pasa en su alrededor, dar información detallada en tiempo real e inclusive actualizando la información minuto a minuto, a través de *tweets*, los cuales son mensajes cortos que pueden ser incluso hilados por medio de un *hasthag* (un identificador de palabra que lleva antepuesto el símbolo numérico #).

Esta facilidad para que cualquier usuario pueda publicar ha hecho que exista un aumento en la difusión de noticias falsas o *"fake news"*.

En algunas ocasiones, los usuarios no solo difunden noticias falsas, sino información engañosa, esto con el uso de lenguaje figurado[1] como metáforas para expresar sus ideas.

Las metáforas, por si solas, no deberían afectar, pero, por otro lado, si un usuario de Twitter utiliza una metáfora que trivializa un desastre, podría generar una mayor atención en este tipo de *tweets*.

Un ejemplo de una metáfora, *Un #tsunami de problemas*, donde no nos referimos a un tsunami real, sino a una gran cantidad de situaciones que para alguien podrían sentirse como un tsunami. En este trabajo se exploró el análisis de tuits etiquetados como desastres y no desastres, donde aquellos que no son desastres, son en su mayoría textos con metáfora.

[1] Figurado: Dicho de un sentido: Que no corresponde al literal de una palabra o expresión, pero está relacionado con él por una asociación de ideas. https://dle.rae.es/figurado

Tabla 1. Ejemplo de uni-gramas, bi-gramas y tri-gramas con la frase *"La niña juega con la pelota"*.

Frase: La niña juega con la pelota		
Uni-gramas	**Bi-Gramas**	**Tri-gramas**
La	La-niña	La-niña-juega
Niña	niña-juega	niña-juega-con
juega	juega-con	juega-con-la
con	con-la	con-la-pelota
la	la-pelota	
pelota		

Una vez que hayan sido analizados, se pasó a una clasificación automática con diferentes modelos de aprendizaje máquina automático, combinado con los extractores de características SBERT y n-gramas.

2. Marco teórico

2.1. n-gramas

Los n-gramas son subsecuencias de n elementos consecutivos, los cuales pueden contener palabras, números, símbolos y puntuación. El valor de "n" puede ser cualquiera, aunque los más utilizados son los uni-gramas (n=1), bi-gramas (n=2) y tri-gramas (n=3) [3].

Los n-gramas pueden ser utilizados como extractores de características, que sirven como entrada a un modelo, especialmente clasificadores. Este proceso de extracción de características se realiza mediante el conteo de ocurrencia de cada n-grama, donde se calcula la probabilidad condicional y se asigna a una categoría [13].

En el cuadro 1 muestra un ejemplo de uni-gramas, bi-gramas y tri-gramas, para la frase *La niña juega con la pelota.*

2.2. SentenceBERT

SentenceBERT o SBERT es una técnica de procesamiento de lenguaje Natural (PLN), utilizado para codificar oraciones en vectores numéricos, los cuales pueden ser comparados de manera sintáctica y semántica [5]. SBERT utiliza una red neuronal para la codificación de frases, en donde se realiza un mapeo de una oración.

Ya que SBERT utiliza técnicas de transferencia para su adaptación en tareas específicas, puede ser utilizado en modelos tradicionales de aprendizaje automático, es decir, solo extrayendo las características de los datos y convirtiéndolas en vectores. El número de dimensiones base de SBERT, es de 768 dimensiones [5], el cual implicara el número de características extraídas por oración.

2.3. Aprendizaje automático tradicional

El aprendizaje automático o *Machine Learning* (ML), es una rama de la inteligencia artificial o *Artificial Intelligence* (AI), para tareas de clasificación, regresión, reducción dimensional y agrupamiento.

Este tipo de algoritmos son entrenados a partir de un conjunto de datos y generando una salida [6]. Existen un sin fin de algoritmos de aprendizaje máquina tradicional. Para fines de esta investigación nos centramos en los siguientes algoritmos de clasificación:

- **XGBOOST *(Extreme Gradient Boosting)*:** Utiliza la técnica de *boosting*, para mejorar el rendimiento en la predicción. Este algoritmo está enfocado en los llamados "modelos débiles" (árboles de decisión) y realizar una combinación predictiva. XGBOOST, también tiene una técnica de regularización, para evitar un sobre ajuste y mejorar la generalización del modelo[7].
- **KNN *(K-Nearest Neighbors)*:** Este algoritmo se basa en "k" muestras más cercanas a una nueva, todo esto dentro de un espacio de características. Después, el valor objetivo de la nueva muestra se predice en función del promedio de los valores de "k" [8].
- **RF *(Random Forest)* :** Es un algoritmo de aprendizaje supervisado, utilizado para la clasificación y regresión. Este funciona basado la creación de múltiples árboles de decisión y la combinación de sus predicciones individuales. Todo lo anterior, ayuda a evitar el sobre ajuste y mejora la generalización [9].
- ***SVM (Support Vector Machine*):** Este algoritmo funciona mediante la búsqueda de hiperplanos óptimos, que maximicen el margen de separación entre clases [8].
- ***DT (Decision tree)*:** Funcionan dividiendo los datos en subconjuntos cada vez más pequeños, todo esto a partir de las características de entrada, a. Los nodos internos del árbol, representan las características de entrada y las hojas las decisiones finales [8].

2.4. Metáfora

De acuerdo a la Real Academia Española [10], la metáfora es una figura retórica que consiste en la utilización de una palabra o expresión en un sentido distinto al habitual, estableciendo relación de semejanza entre dos elementos que no son idénticos.

Por ejemplo: *"El sol caía sobre la ciudad como una lluvia de fuego"*, en esta metáfora, se compara la sensación del calor intenso con una lluvia de fuego, lo que da la idea de una fuerza destructiva y peligrosa.

3. Estado del arte

Hoy en día existen diferentes trabajos relacionados con eventos de crisis, a continuación se presentan algunos:

Parrilla-Ferrer et al. [11], hacen uso de la clasificación binaria automática, esto a través de Naive Bayes y SVM. Este trabajo utiliza un conjunto de *tweets* etiquetados a partir de la inundación del 2012 en Habagat, en donde se pueden encontrar *tweets* informativos y no informativos.

Se ha encontrado que el uso de n-gramas ha ayudado a conocer el contexto en algunas tareas de PLN. Thewall Mike et al. [12] encontraron que la combinación de uni-gramas, bi-gramas y tri-gramas proporcionan resultados de hasta un 86.40 % en la métrica de precisión.

Tabla 2. Extracto del conjunto de datos de entrenamiento de Disaster Tweets [18].

id	keyword	location	text	target
48	ablaze	Birmingham	@bbcmtd Wholesale Markets ablaze http://t.co/lHYXEOHY6C	1
49	ablaze	Est. September 2012 - Bristol	We always try to bring the heavy. #metal #RT http://t.co/YAo1e0xngw	0
52	ablaze	Philadelphia, PA	Crying out for more! Set me ablaze	0
363	annihilation	United States	Are souls punished annihilation? http://t.co/c1QXJWeQQU http://t.co/Zhp0SOwXRy	0
364	annihilation		@CalFreedomMom @steph93065 not to mention a major contributor to the annihilation of Israel	1
365	annihilation		@willienelson We need help! Horses will die!Please RT &amp	1
394	annihilation	Chandler, AZ	U.S National Park Services Tonto National Forest: Stop the Annihilation of the Salt River Wild Horse... http://t.co/SB5R7ShcCJ via @Change	1
396	apocalypse	ColoRADo	I'm gonna fight Taylor as soon as I get there.	0
397	apocalypse	sindria	ohH NO FUKURODANI DIDN'T SURVIVE THE APOCALYPSE BOKUTO FEELS HORRIBLE my poor boy my ppor child	1

Así mismo, en esta investigación se encontró que esta técnica de extracción de características tiene buenos resultados con clasificadores como Naive Bayes, SVM, Regresión Logística y Árboles de Decisión.

Los enfoques clásicos no han quedado excluidos, Stowe, K., et al. [13] trabajaron con el conjunto de datos del huracán Sandy del 2012, en el cual recolectaron al rededor de 22.2M de *tweets* con las fechas de 23 de octubre del 2012 al 5 de abril del 2013.

Para el preprocesamiento de datos, utilizaron los enfoques clásicos y para la extracción de características utilizaron uni-gramas y bi-gramas, poniendo especial atención en algunos elementos que generalmente eliminan como los *retweet* y URL. Los resultados más relevantes encontrados con una selección de características fueron de 0.75 %.

Enfocándonos de manera específica en el conjunto de datos *Disaster Tweets*, se observan tres trabajos: El realizado Saji, B., et al. [14] hacen un análisis en el conjunto de datos para la remoción de urls, arrobas, y menciones. Para el preprocesamiento, lematizan, convierten a minúsculas, eliminan palabras de parada y utilizan la codificación *one-hot*.

Para el modelo de clasificación utilizan LSTM*(Long Short Term Memory)*, con el cual obtienen el *accuracy* del 95 % en el conjunto de entrenamiento.

El autor autodenominado **wisdomml** en [15] realizan un preprocesamiento similar a Saji, B., et al., a excepción de incluir la remoción de caracteres no alfabéticos y una tokenización, la extracción de características la realzian por meido de TF-IDF *(Term Frecuency-Inverse Document Frecuencuency)* combnado con el uso de bi-gramas y tri-gramas.

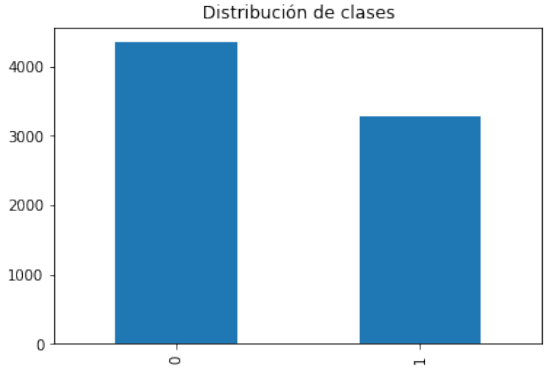


Fig. 1. Distribución de las clases del conjunto de datos *DisasteerTweets*.

Los clasificadores son Naive-Bayes y un *Passive Aggressive Classifier*, estos con validación cruzada. Los mejores resultados obtenidos fueron: Accuracy, bi-gramas y Naive-Bayes con 80.03 %; Precisión, tri-gramas y Naive-Bayes con 86.68 %; F1, tri-gramas con *Passive Aggresive* con 75.18 %.

Por último, en el tercer trabajo realizado por Krishnakumar, M., [16], realizan un preprocesamiento que consiste en eliminar las urls, conversión del texto a minúsculas y la tokenización. Para la clasificación utilizan un *embedding* de tamaño de vocabulario de 10,000 con una longitud máxima de 280, un *max-pooling* de 1 dimensión con tamaño 16, 1 capa densa de 16 y una capa densa final de 1. La métrica reportada es *accuracy*, el cual les da un 77.38 %.

4. Análisis de los datos

4.1. Conjunto de datos

Disaster Tweets [18] es un conjunto de datos de la plataforma Kaggle, donde es resaltado el uso de Twitter como un importante canal de comunicación en tiempos de emergencia.

El conjunto de datos está compuesto por dos archivos de información, entrenamiento y pruebas: Entrenamiento, incluye 5 columnas (id, keyword, location, text y target); Pruebas, incluye 4 columnas (id, keyword, location, text). A continuación, se realizará una descripción del contenido por columna:

- **id:** Identificador de la fila.
- **keyword:** Palabra clave con la que fue etiquetada el desastre natural, el conjunto cuenta con alrededor de 223, por ejemplo: *ablaze*, ardiendo; *sunk*, hundido; *survive*, sobreviviendo; *panic*, pánico; *police*, policía; entre algunos otros. Puede estar vacío.
- **location**: Ubicación del desastre, puede ser una ciudad, país, calle o la localidad (por ejemplo, playa o bosque). Puede estar vacío.
- **text:** Es el texto denominado *tweet*, en donde esta escrita la información.

Fig. 2. Palabras más relevantes con *hashtag* positivos en *DisasteerTweets*.

Fig. 3. Palabras más relevantes con *hashtag* negativos en *DisasteerTweets*.

– **target**: Indica si el texto es un desastre o no: **0** indica que no es un desastre; **1** indica que es un desastre.

En la tabla 2 se puede visualizar un extracto del conjunto de datos.

Distribución de los datos: El conjunto de datos de entrenamiento cuenta con 7613 tweets: 4342 (57 %), con la etiqueta 0 y 3271 (43 %), con la etiqueta 1. En la Figura 1 se muestra esta distribución de la información.

Distribución de palabras: En Twitter, es bastante común poder hilar o conjuntar los datos, inclusive si estos no están estrictamente relacionados. En la figura 2 se muestra la nube de los *tweets* etiquetados como desastre y en la figura 3 se muestra la nube de los *tweets* etiquetados como no desastre.

5. Clasificación

La clasificación se realizó de la siguiente manera:

– Preprocesamiento.
– Extracción de Características.

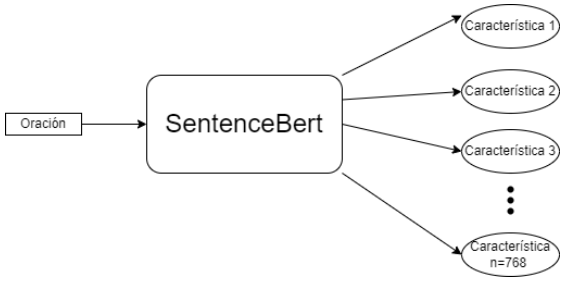


Fig. 4. Ilustración de extracción de características por oración con SBERT.

1. N-gramas,
2. Sentence Bert.

– Partición de los datos.
– Clasificadores.
1. KNN,
2. Random Forest,
3. Árboles de decisión,
4. SVM,
5. XGBOOST.

5.1. Preprocesamiento

Para obtener los mejores resultados, se realizó un preprocesamiento aprovechando un conjunto de funciones disponibles en un repositorio de GitHub [2], además de añadir otros métodos clásicos de preprocesamiento de textos. El tratamiento previo a los datos quedo compuesto de la siguiente manera:

– **Paso de minúsculas a mayúsculas:** Todas las letras pasan a minúsculas.
– **Remplazo:**
1. Cantidades numéricas.
2. Contracciones.
3. Símbolos matemáticos por descripciones.

– **Remoción:**
1. Hipervínculos.
2. Palabras de parada (*stop words*).
3. Apóstrofes.
4. Slashes y slashes invertidos.
5. Alfanuméricos.
6. Saltos de línea.
7. ASCII.
8. Signos de puntuación.

[2] Macias, C., Soto, M. (2023). Text preprocessing, disponible en https://github.com/CCogS-Mx/text-preprocessing

Tabla 3. Resultados de F1, Recall y Precisión con n-gramas.

	KNN			Árboles de Decisión			Random Forest		
Método	**F1**	**Presicion**	**Recall**	**F1**	**Presicion**	**Recall**	**F1**	**Presicion**	**Recall**
1g	0.574	0.788	0.611	0.700	0.759	0.699	0.698	**0.816**	0.694
1g + 2g	0.523	0.784	0.581	**0.704**	0.763	**0.703**	0.608	0.813	0.636
2g	0.501	0.791	0.569	0.541	0.795	0.591	0.503	0.804	0.571
2g + 3g	0.501	0.798	0.569	0.547	0.795	0.595	0.507	0.805	0.573
3g	0.514	0.797	0.576	0.493	0.794	0.575	0.488	0.802	0.562

9. Entidades de Twitter.
10. Paréntesis.
11. Espacios en blanco.

– **Tokenización.**
– **Lematización.**
– ***Steamming*.**

5.2. Extracción de características:

Se realizaron dos diferentes extracciones de características:

– **Extracción de características mediante N-gramas**
Se utilizó el conteo de vectorización para la extracción de las características basadas en n-gramas. Estas fueron hechas en orden y convertidas en una lista por cada tipo. Se eligieron las siguientes configuraciones de n-gramas: uni-gramas; bi-gramas; tri-gramas; uni-gramas, bi-gramas; bi-gramas, tri-gramas.
– **Extracción de características mediante SentenceBERT**
Para la extracción de características mediante SentenceBERT, se introdujo cada oración dentro del modelo para obtener un vector. Como espacio vectorial se eligieron 768 dimensiones. En la figura 4 se puede visualizar el funcionamiento general del modelo.

5.3. Partición de los datos

El conjunto de datos fue dividido en 80 % para el entrenamiento y en 20 % para validación.

5.4. Modelos

Después de extraer las características, se seleccionaron 5 modelos para la clasificación con la siguiente configuración:

– **XGBOOST**: Fue el clasificador utilizado con la extracción de características de SentenceBERT, incluye una profundidad de 14. Como *booster* utiliza un árbol de gradiente descendiente.
– **KNN:** Fue utilizado con la extracción de características de n-gramas, se utilizaron 3 diferentes parámetros de vecinos: 9, 7 y 17.

Tabla 4. Mejores Resultados de F1, Recall y Presicion con n-gramas, para extracción de características de SentenceBERT.

	Métrica		
Vector	**F1**	**Presicion**	**Recall**
Random Forest	0.633	0.651	0.616
XGBOOST	0.625	0.632	0.617
SVC	0.710	0.720	0.710

- **RF:** Este algoritmo fue utilizado en los dos tipos de extracción de características: n-gramas, se utilizaron 200 estimadores y una profundidad de máxima de 32; SBERT, se utilizaron 200 estimadores y profundidad máxima de 50. Esta discordancia en profundidad se debe a la búsqueda de los mejores parámetros en ambos experimentos.
- **SVM:** Este algoritmo fue utilizado para SBERT, incluye un máximo de 1000 iteraciones y un parámetro de mezcla de la red elástica de 0.15.
- **DT:** Este algoritmo solo fue utilizado para la extracción de características de n-gramas, en el cual se implementaron 3 diferentes parámetros de profundidad máxima, 7, 8 y 30.

En algunos casos, se muestra que no se usó la misma extracción de características para todos los clasificadores. Para el caso de los clasificadores que solo usaron la extracción de características de SBERT, como ya se mencionó en la sección 2, esta es una técnica más avanzada que convierte lo extraído a vectores numéricos, en cambio, los n-gramas hacen un conteo de palabras y los vectores numéricos, lo que hace que los vectores numéricos de alta dimensión sean mejor procesados por modelos como XGBoost.

En el caso del clasificador SVM, al ser un clasificador lineal, permite una mejor separación de las clases [19], lo que hace que sea adecuado para trabajar con características extraídas con SBERT.

Por otro lado, las KNN tienen una gran capacidad de manejar características basadas en n-gramas, las cuales son altamente dimensionales [20]. Para el caso de los árboles de decisión, además de tener la misma capacidad de manejar características de esas dimensiones y son altamente interpretables. En adición, fue relevante hacer una revisión del comportamiento de XGBOOST con SBERT y los DT con n-gramas, ya que XGBOOST utiliza de fondo los árboles de decisión.

Para el caso de RF, se estudió una comparación directa del funcionamiento de un algoritmo que es menos propenso al sobre ajuste.

6. Resultados

Se realizaron 15 experimentos para la extracción de características con N-gramas y 4 experimentos para la extracción de características con SBERT. Para la evaluación de estos modelos, se consideró apropiado la utilización de las métricas de F1, precisión y *recall*.

Tabla 5. Comparación respecto al estado del arte.

	Métricas			
Comparación	**Accuracy**	**F1**	**Presición**	**Recall**
Saji, B., et al. [14]	95 % * [4]	-	-	-
wisdomml [15]	80.03 %	75.18 %	86.68 %	-
Krishnakumar, M. [16]	77.38 %	-	-	-
Nuestros resultados	-	70.04 %	81.60 %	70.30 %

La tabla 3 muestra los resultados (resaltados con letras negritas) con la extracción de características n-gramas[3].
donde se pueden visualizar las mejores combinaciones y resultados para las métricas propuestas, los cuales son: F1, 1g + 2g con árboles de decisión; Precisión, 1g con Random Forest; Recall 1g + 2g, Árboles de Decisión.

La tabla 4 muestra los resultados con la extracción de características de SBERT. Se puede notar que el mejor clasificador con estas características fue SVC.

Los resultados obtenidos en este análisis pueden ser comparados directamente con los trabajos [14, 15, 16], ya que utilizan el mismo conjunto de datos. Pero, aunque el conjunto sea el mismo, los autores de cada uno de los artículos no se centran en el análisis de los datos contextuales, únicamente en la clasificación de ellos. En la tabla 5 se puede observar esta comparación.

7. Conclusiones y trabajo futuro

En este artículo se presentó el análisis y clasificación automática de *tweets* sobre el conjunto de datos de *DisasterTweets*, a través de algoritmos clásicos de aprendizaje automático y extractores de características como n-gramas y SBERT.

A pesar de que existen múltiples trabajos de clasificación automática con este conjunto de datos e inclusive, resultados superiores, este no fue el único objetivo alcanzado.

Como se observó, los textos contextuales basados en metáforas son extremadamente abstractos y la utilización de una comparación explícita en algunas ocasiones no es clara.

Se observó que el uso de n-gramas combinados es importante para la clasificación, ya que la relación de un texto puede ser capturada por medio de la coocurrencia de los n-gramas, esto se puede verificar al observar que los mejores resultados son cuando existen combinaciones de uni-gramas y bi-gramas.

Por otro lado, aunque SBERT es efectivo y es un extractor muy avanzado, no se toma específicamente para la extracción de los patrones gramaticales o conocer la relación de palabras específicas, y se puede observar en los resultados obtenidos.

Este trabajo nos demuestra que es importante hacer un estudio más profundo en las características principales del texto, es decir, una exploración del conjunto de datos, las uniones, los diferentes patrones, y poner especial “atención”.

[3] 1g, unigramas; 2g, bigramas; 3g, trigramas. El símbolo de “+” denota la unión de N-gramas

Como trabajo a futuro se propone aplicar mecanismos de atención, que den mayor peso a las relaciones de las palabras. Específicamente en explorar las capas de atención de BERT *(Bidirectional Encoder Representations from Transformers)* [21], ya que usa un enfoque bidireccional, es decir, va de izquierda a derecha para crear la representación vectorial de alta precisión.

Referencias

1. Naciones Unidas: Las catástrofes relacionadas con el clima se quintuplican en 50 años, pero la mejora de los sistemas de alerta salva más vidas. Noticias ONU (2021) news.un.org/es/story/2021/09/1496142
2. Rodríguez, H., Trainor, J. E.: Emergency response and information exchange during natural disasters: The role of the amateur radio service. Disasters, vol. 22, no. 3, pp. 238–250 (1998)
3. Jurafsky, D., Martin, J. H.: Speech and language processing: An introduction to natural language processing, computational linguistics, and speech recognition. Pearson Education (2019)
4. Manning, C., Schütze, H.: Foundations of statistical natural language processing. The MIT Press (1999)
5. Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using siamese BERT-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), pp. 3982–3992 (2019) doi: 10.18653/v1/D19-1410
6. Russell, S., Norvig, P.: Artificial intelligence: A modern approach. Pearson (2010)
7. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794 (2016) doi: 10.1145/2939672.2939785
8. Hasan, M. M., Hossain, M. S., Rahman, M. S.: A comprehensive survey on machine learning techniques. Applied Sciences, vol. 11, no. 4 (2021)
9. Nielsen, M.: Neural networks and deep learning. Determination Press (2015)
10. Real Academia Española: Metáfora. Diccionario de la lengua española (22.ª ed.) (2001) https://www.rae.es/drae2001/metáfora
11. Parilla-Ferrer, B. E., Fernández, P. L., Ballena, J. T.: Automatic classification of disaster-related tweets. In: International conference on Innovative Engineering Technologies, pp. 62–69 (2015) doi: 10.15242/IIE.E1214072
12. Thelwall, M., Buckley, K., Paltoglou, G.: Sentiment in twitter events. Journal of the American Society for Information Science and Technology, vol. 62, no. 2, pp. 406–418 (2011) doi: 10.1002/asi.21462
13. Stowe, K., Paul, M. J., Palmer, M., Palen, L., Anderson, K. M.: Identifying and categorizing disaster-related tweets. In: Proceedings of the Fourth International Workshop on Natural Language Processing for Social Media, pp. 1214–1224 (2016) doi: 10.18653/v1/W16-6201
14. Saji, B.: Disaster tweet classification using LSTM - NLP. Analytics Vidhya (2022) https://www.analyticsvidhya.com/blog/2022/05/disaster-tweet-classification-using-lstm-nlp/
15. Wisdom ML: Disaster tweets classification using machine learning & NLP approach. NLP Project (2022) https://wisdomml.in/disaster-tweets-classification-using-machine-learning-nlp-approach/
16. Krishnakumar, M.: Natural language processing with disaster tweets : Part 1. Medium (2021) https://medium.com/@mukilankrishnakumar2002/natural-language-processing-with-disaster-tweets-part-1-db31c9ad07

17. Liu, J., Singhal, T., Blessing, L., Wood, K., Hui-Lim, K.: CrisisBERT: A robust transformer for crisis classification and contextual crisis embedding. In: Proceedings of the 32nd ACM Conference on Hypertext and Social Media, pp. 133–141 (2021) doi: 10.1145/3465336.3475117
18. Stepanenko, V.: Disaster tweets. Real or not? NLP with disaster tweets challenge add-on. Kaggle (2021) www.kaggle.com/datasets/vstepanenko/disaster-tweets
19. Sun, C., Qui, X., Xu, Y., Huang, X.: How to fine-tune BERT for text classification? (2020) doi.org/10.48550/arXiv.1905.05583
20. Zhang, H., Yu, L., Zhou, M.: A review on recent advances in N-gram language modeling. Computer Science Review, vol. 29, pp. 21–39 (2018)
21. Devlin, J., Chang, M. W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 4171–4186 (2019) doi: 10.18653/v1/N19-1423

ISSN 1870-4069

Análisis de sentimientos de textos de Twitter utilizando aprendizaje profundo

Jessica Olivares L., Abraham Sánchez L., Rogelio González V., Abraham Maldonado G.

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

{olivares.lopez.jessica3p, rogelio.gzzvzz}@gmail.com,
{abraham.sanchez, abraham.maldonadoga}@correo.buap.mx

Resumen. En este trabajo se propone una serie de estrategias provenientes del área de aprendizaje máquina, incluyendo el aprendizaje profundo. El conjunto de datos utilizados está integrado con texto extraído de Twitter, específicamente con tweets relacionados a la inteligencia artificial, lo cual adicionalmente proporciona un pequeño panorama respecto a la opinión pública de esta área de la ciencia, a partir del análisis y la exploración de datos. Se realizaron experimentos para análisis de sentimientos desde distintos puntos de referencia, de manera supervisada y no supervisada. El desempeño o resultado de cada una de estas estrategias está variado por la integridad de datos, el problema o tema principal de los datos, la variación de parámetros, entre otros más factores.

Palabras clave: Análisis de sentimientos, Twitter, aprendizaje profundo, PLN.

Sentiment Analysis of Twitter Texts Using Deep Learning

Abstract. In this paper, a series of strategies from the machine learning area, including deep learning, are proposed. The data set used is integrated with text extracted from Twitter, specifically with tweets related to artificial intelligence, which additionally provides a small overview of public opinion in this area of science, based on data analysis and exploration. Experiments for sentiment analysis were carried out from different points of reference, in a supervised and unsupervised manner. The performance or result of each of these strategies is varied by data integrity, the main data problem or issue, parameter variation, among other factors.

Keywords: Sentiment analysis, Twitter, Deep learning, NLP.

1. Introducción

El análisis de sentimientos es una de las aplicaciones de la clasificación de textos del Procesamiento del lenguaje natural (PLN), básicamente asigna una categoría apropiada al contenido de una oración, texto o documento, a partir del procesamiento

de texto (previamente no estructurado). Esta clasificación se hace mediante la asignación de una polaridad de sentimiento: positivo, negativo o neutro a una oración o un documento, o a partir de la asignación de una emoción que se identifique en la oración [1].

El análisis de sentimientos es una de las técnicas de mayor relevancia de esta área, generalmente para la evaluación de todo el contenido en texto generado en la web [2]. Sin embargo, el análisis de sentimientos es un área multidisciplinar, pues además del procesamiento de lenguaje natural intervienen disciplinas cómo lingüística y psicología.

El hecho de definir las categorías en las que será clasificado un texto basado en la premisa de "sentimiento" es una cuestión psicológica muy ambigua y diversa, pues existe una variedad de posturas al respecto, hay cientos de estados emocionales que forman parte de la condición humana.

Una clasificación o representación de las emociones humanas popular, es la rueda de Plutchik. El psicólogo Robert Plutchik sugiere que hay 8 emociones evolutivas humanas, es decir, emociones que han formado parte de la supervivencia humana y que han sido transferidas de generación en generación. Estas emociones son las siguientes: Ira, Miedo, Tristeza, Repugnancia, Sorpresa, Expectación, Confianza y Alegría.

En la representación de Plutchik se muestra que cada una de estas emociones centrales puede intensificarse, atenuarse o incluso combinarse para producir cualquier estado emocional. Sin embargo, muchos de los trabajos realizados para análisis de sentimientos, están basados en la clasificación del texto de acuerdo con la polaridad del sentimiento, bueno, malo o neutral, en su mayoría.

Lo cual facilita considerablemente el problema de análisis de sentimientos, además, la polaridad puede ser más precisa porque solo hay dos o tres clases distintas, que hace más fácil de distinguirlas entre sí, mientras que una clasificación por emociones puede ser más ambigua, pues una oración puede involucrar más de una emoción. Generalmente el análisis de sentimientos hace la categorización de texto no procesado en polaridades de acuerdo con las siguientes categorías: Positivo, Negativo y Neutro.

El conjunto de estrategias que se proponen en este trabajo, puede ser aplicable o adaptable a cualquier conjunto de datos en formato de texto para problemas de análisis de sentimientos sin importar el dominio del texto, como se verá en el apartado de resultados obtenidos, el desempeño de cada una de las técnicas propuestas varía de acuerdo con la integridad de los datos, el problema o tema principal de los datos, la variación de los parámetros, entre otros factores.

2. Trabajos relacionados y enfoques

El análisis de sentimientos es una de las áreas de investigación más vigorosas en el campo del procesamiento de lenguaje natural (PLN) que se centra en analizar las opiniones, sentimientos, actitudes y emociones de las personas hacia varias entidades, como productos, servicios, organizaciones, problemas, eventos y temas [1]. Recientemente se han publicado una gran cantidad de trabajos de investigación sobre análisis de sentimientos en diferentes idiomas.

Por lo tanto, para lograr esto, primero deberemos revisar las bases para la investigación sobre el análisis de sentimientos mediante la revisión de la literatura relevante sobre estudios anteriores realizados en este campo. En uno de estos trabajos, los autores proponen que el análisis de sentimientos se podría dividir en tres enfoques principales, a saber, un enfoque basado en el aprendizaje máquina, un enfoque basado en el conocimiento y un enfoque híbrido [2]. En este trabajo adoptamos el enfoque de aprendizaje máquina.

Un trabajo interesante y relacionado a nuestra propuesta, se propuso en [2] para el análisis automático de los comentarios de los usuarios de Internet que se publican en las páginas oficiales de los supermercados en Túnez en las redes sociales de Facebook. En su propuesta utilizaron CNN (Convolution Neural Networks), LSTM (Long Short Term Memory) y Bi-LSTM (Bi-directional Long Short Term Memory) indicando que los resultados obtenidos son satisfactorios, especialmente para los algoritmos LSTM y Bi-LSTM.

En [3] los autores señalan que el uso de la técnica de análisis de características juega un papel importante en el desarrollo y la mejora de un modelo de análisis de sentimientos. La propuesta de los autores coincide con esta propuesta en el hecho de poder utilizar datos ruidosos, palabras ajenas al vocabulario, etc.

Su propuesta es interesante pues utilizan un modelo BERT previamente entrenado que permite obtener características semánticas y contextuales a nivel de oración y con ello generar integridades. En sus resultados experimentales, destacan el uso de un algoritmo CNN dilatado para poder extraer información local y global.

Los autores en [4] presentan un estudio de varias arquitecturas de aprendizaje profundo y sus aplicaciones en el análisis de sentimientos. En el trabajo, ellos detallan cómo muchas de estas técnicas de aprendizaje profundo han demostrado su utilidad en muchas tareas del análisis de sentimientos y vislumbran en los años venideros, un aumento importante en el uso del aprendizaje profundo en investigaciones.

La investigación existente en el área ha producido muchísimas técnicas para diversas tareas del análisis de sentimientos, que incluyen algoritmos supervisados y no supervisados. Es claro que dentro de los enfoques supervisados se utilizan ampliamente las máquinas de vectores de soporte (SVM), entropía máxima, naïve Bayes, entre otros; incluso combinaciones de estos algoritmos han dado buenos resultados.

Por su parte, los algoritmos no supervisados incluyen métodos que explotan los léxicos de sentimiento, el análisis gramatical y los patrones sintácticos. Sería poco práctico detallar los libros y artículos, pero el lector interesado puede revisar la literatura existente en el área [1, 7, 8].

3. Extracción y exploración de datos

Uno de los objetivos de la propuesta, es recuperar datos de Twitter, obteniendo así datos no estructurados, a los cuales es necesario aplicar estrategias de preprocesamiento para poder realizar el análisis de sentimientos. El primer paso, es consolidar un conjunto de datos que contenga la información y estructura necesaria para realizar la tarea propuesta. El proceso de creación de un conjunto de datos consta de tres procesos: adquisición de datos, limpieza de datos y etiquetado de datos.

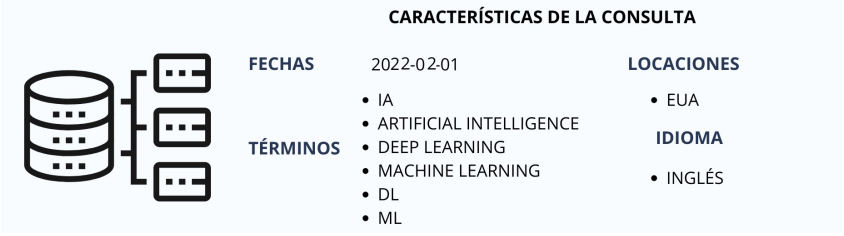


Fig. 1. Estructura de consulta de extracción de texto de Twitter.

3.1. Adquisición de datos

Para la adquisición de datos se hizo uso de la API de Twitter, bajo una cuenta de desarrollador. La consulta de extracción de datos se describe en los parámetros de la Fig 1.

Se realizó un filtro de los Tweets que en su contenido se encontrarán los términos de la Fig 1, inteligencia artificial, deep learning y machine learning esencialmente, al ser considerados unos de los temas más relacionados y populares de la inteligencia artificial en ese periodo de tiempo. Obteniendo así, un total de 12,000 observaciones, que en este contexto son Tweets.

3.2. Limpieza de datos

El proceso de limpieza de datos es uno de los procesos que define el éxito de un modelo de aprendizaje máquina (AM). Para tareas o problemas de PLN se realizan algunas estrategias básicas que facilitan la manipulación del texto. El proceso de limpieza de esta propuesta está compuesto de dos partes tal y cómo se ilustra en la siguiente Fig. 2.

La limpieza básica consiste en la eliminación de algunos elementos, pues al ser extraídas las observaciones de un medio social como Twitter, en el cual el texto lo integran elementos adicionales como: usuarios, menciones, hashtags, enlaces o URL y emoticones.

Algunos de estos elementos deben ser eliminados, pues pueden introducir ruido y reducir el desempeño o aprendizaje de un modelo. La segunda parte de la limpieza de datos es más general para cualesquiera tareas de Lenguaje de Procesamiento Natural y consiste en la realización de las siguientes tareas [9].

- **Convertir el texto a minúsculas**. Esto facilita algunos procesos de exploración que se muestran más adelante.
- **Eliminar espacios dobles.** Se hace la sustracción de espacios dobles, porque no aportan nada al contenido del texto.
- **Eliminar números.** La eliminación de números previene la errónea interpretación de números en los modelos.

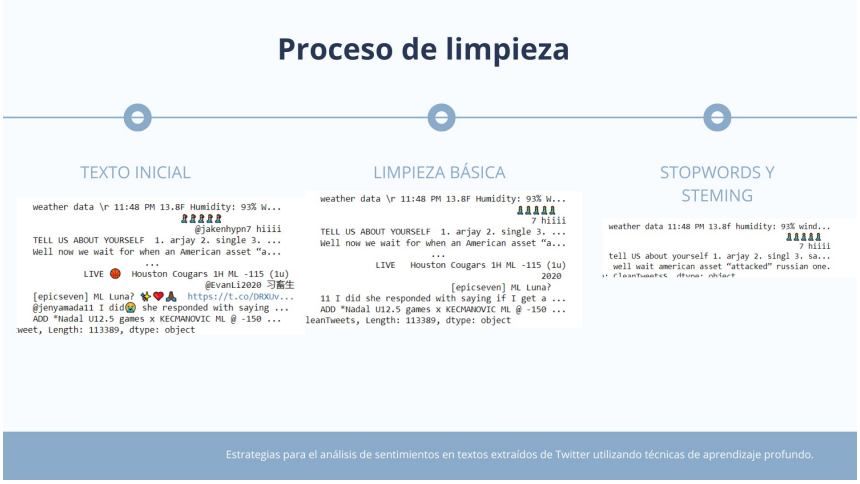


Fig. 2. Proceso de limpieza del conjunto de datos.

- **Eliminar StopWords**. Consiste en la eliminación de palabras gramaticales que regularmente se ocupan con mayor frecuencia en el idioma y que podrían no añadir valor al contexto, ejemplos de estas palabras son: conjunciones, artículos, preposiciones, etcétera.
- **Stemming.** Este proceso consiste en convertir una palabra a su palabra base (obtener la raíz de una palabra).

3.3. Etiquetamiento de datos

Finalmente, una vez recolectados y estructurados los datos, se procede a la asignación de una etiqueta, en particular, una polaridad de sentimiento. Esta asignación se realizó de manera semi-supervisada. Mediante un flujo de trabajo en Python que solicitó respuesta del servicio de Text Analysis de Azure, el cual, entre otras tareas, puede asignar una polaridad de sentimiento a un conjunto de documentos o textos.

Además de una supervisión manual sobre la etiqueta asignada por Text Analysis. Text Analysis, asigna una de las siguientes categorías o polaridad de texto: neutral, positive, negative y mixed.

También devuelve puntuaciones de confianza entre 0 y 1 para cada documento y oraciones dentro del documento. Una vez realizada la solicitud de las 12,000 observaciones recolectadas se obtiene la siguiente cantidad de observaciones para cada categoría de sentimiento: neutral; 6162, positive; 2965, negative; 2290 y mixed; 583, relación que se muestra en la tabla 1.

3.4 Exploración de datos

La exploración de datos es el proceso en el cual se realiza la examinación de los datos para entender la composición o integración de estos y obtener las primeras observaciones e inferencias sobre ellos. Para el procesamiento del lenguaje natural

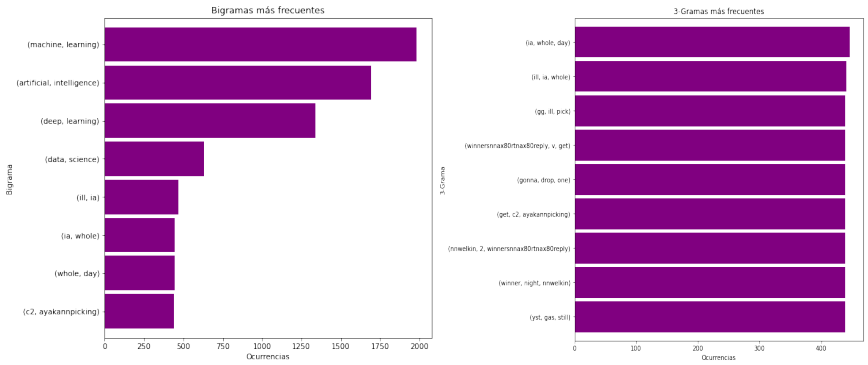


Fig. 3. Histogramas de bigramas y trigramas más frecuentes.

Tabla 1. Número de observaciones por sentimiento.

Sentimiento	Observaciones
Neutral	6162
Positivo	2965
Negativo	2290
Mezclado	583

existen varias técnicas que permiten visualizar y deducir conclusiones eficientemente. Se realizaron algunas de estas técnicas, con las que se obtuvieron peculiares observaciones que se detallan ampliamente en [10].

En primer lugar, se obtuvieron dos histogramas de la frecuencia de palabras a partir de la composición de n-gramas. Los n–gramas son una subsecuencia, donde *n* es el número de elementos de la subsecuencia; esta técnica es usualmente utilizada en el procesamiento del lenguaje natural para el tratamiento de textos.

Los histogramas de las siguientes figuras corresponden a la división del texto en 2 y 3 gramas, bigramas y trigramas, respectivamente. Para obtener este gráfico es necesario obtener la lista de bigramas y trigramas con su respectivo número de ocurrencias a partir del corpus.

Al comparar ambos histogramas, la implementación de bigramas favorece al comportamiento de los datos, pues retomando el contexto, se tiene mayor coherencia del contenido con respecto de la división en trigramas donde, los subconjuntos de palabras no tienen mucha relevancia al contexto del problema. Entre los bigramas más relevantes del histograma de la Fig. 3 se encuentran, machine-learning, artificial-intelligence, deep-learning y data-science, todos estos términos que son directamente relacionados con la “Inteligencia artificial”.

A partir de la lista de bigramas obtenidos, también se puede obtener una representación semántica, la cual muestra la relación que hay entre distintos bigramas establecidos en el paso anterior. La visualización de bigramas es de ayuda para confirmar que los datos si están constituidos en base al contenido deseado, es decir, con opiniones respecto a la Inteligencia artificial.

Se utilizaron representaciones de texto muy populares en aprendizaje profundo, entre las que se encuentran Word embeddings o incrustaciones de palabras. Sin

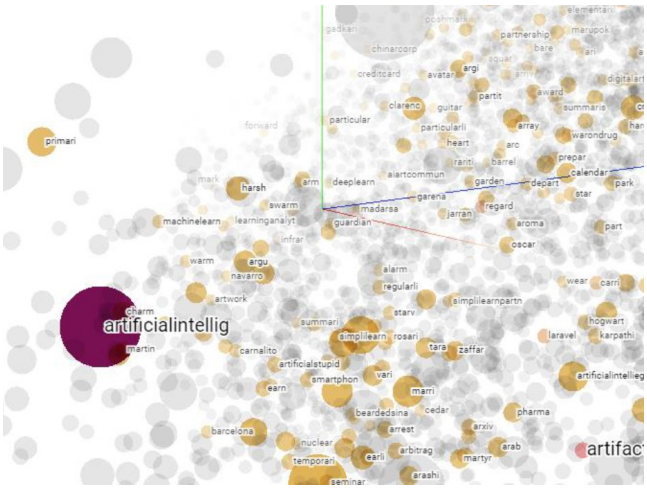


Fig. 4. Visualización de espacio vectorial de Word Embeddings cercanas a "artificialIntelling".

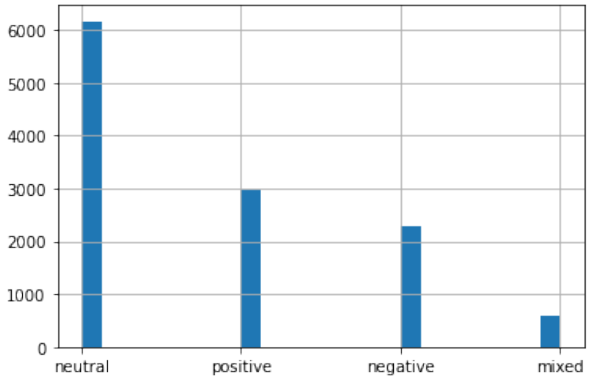


Fig. 5. Histograma de la distribución de datos respecto a la variable objetivo (sentiment).

embargo, esta misma representación también puede ser eficiente desde el proceso de exploración de datos, las incrustaciones de palabras son representaciones vectoriales que permiten entre otras cosas conocer su relación vectorial entre estas.

Mediante la ayuda del proyector de Embeddings de TensorBoard [11] una herramienta que permite visualizar gráficamente los vectores de incrustaciones, para facilitar la interpretación de las relaciones al visualizar Word embeddings, se muestra la visualización de las Word Embeddings, dónde cada punto es el vector correspondiente a la representación de la palabra que contiene la etiqueta.

Para poder visualizar las incrustaciones específicas del conjunto de datos, cómo se muestra en la Fig.4, es necesario hacer un entrenamiento previo, que genera los vectores de características de las dimensiones correspondientes a las representaciones del vocabulario de las observaciones de los datos.

Esta herramienta permite a partir de una palabra, encontrar su representación vectorial y las palabras más cercanas, es decir, las palabras relacionadas a partir del cálculo de la distancia (coseno o euclidiana).

Otras deducciones interesantes abordadas en el proceso de exploración de datos fue la identificación del número de observaciones para cada clase, en el histograma de la Fig. 5 se muestran las observaciones que se tienen para cada una de las categorías de sentimiento.

De la cual a primera vista se puede ver una desproporción de observaciones. La distribución de datos está desbalanceada, y este problema podría afectar el desempeño de los modelos, al introducir un sesgo de información sobre las clases mayoritarias y minoritarias; neutral y mezclada, respectivamente. Para evitar problemas en fases futuras, se aplican técnicas de balanceo de clases [12].

Para no reducir considerablemente el número de instancias de cada clase se eliminaron las muestras de la clase mixed, reduciendo el número de observaciones totales a 11,417. La eliminación de observaciones o instancias para el balanceo de clases es lo que se conoce como undersampling, y ha sido aplicado a la distribución de datos, de forma aleatoria para al final tener 2,290 observaciones de cada clase.

Para reducir la dimensionalidad del conjunto de datos, solo se seleccionaron algunas columnas sobre las cuales trabajaremos para realizar la tarea de análisis de sentimientos con diversos algoritmos, como se verá a continuación.

4. Implementación de las estrategias

Una característica importante para la realización de alguna tarea no supervisada es que las observaciones no están etiquetadas previamente, como es el caso del conjunto de datos propuesto. Sin embargo, se debía implementar una forma de tener datos etiquetados que ayude a evaluar el comportamiento de los modelos, y aunque no sean entrenados con estas etiquetas en los modelos no supervisados, permita comparar su desempeño y además generar modelos de manera supervisada.

Para lo anterior, se realizó el etiquetado del conjunto de datos mediante el apoyo de una herramienta en la nube, que permite categorizar texto, mediante el análisis de sentimientos, presentada en el punto anterior. Esta función de análisis de sentimientos proporciona etiquetas de polaridad de sentimiento (como "negativo", "neutral" y "positivo") basadas en la puntuación de confianza más alta encontrada por el servicio a nivel de oración y documento.

Es importante recordar que los procesos de limpieza y preproceso de datos es parte esencial para la realización de los algoritmos que se estarán valorando y/o evaluando. Entre los algoritmos seleccionados de clustering se eligieron los siguientes: K-means y algunas de sus variaciones.

Los algoritmos de clustering son fundamentalmente métodos de aprendizaje no supervisados, por lo que, las métricas de evaluación del modelo resultante se deben adaptar a este enfoque. Para las comparaciones de los desempeños se realizaron cuatros configuraciones distintas, con las siguientes especificaciones: i) K-means con TF-IDF, ii) K-means con LSA y TF-IDF, iii) K-means con LSA y vectores hashed, iv) MinibachKmeans con LSA [10].

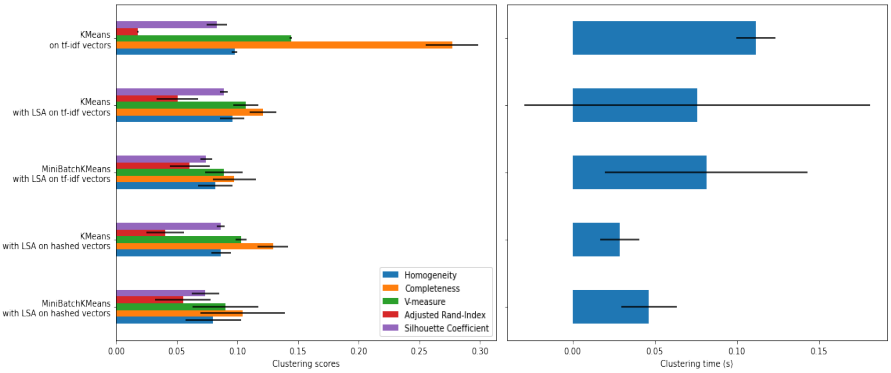


Fig. 6. Resultados de las métricas para la evaluación de los modelos de clustering con 3 clústeres.

Tabla 2. Relación de palabras clave creadas con el léxico de Sentiment Analysis VADER.

	Positive	**Neutral**	**Negative**
Num. Palabras Clave	2986	703	3813

Tabla 3. Parámetros para Lbl2Vec.

Parámetroo	**Valor**
similarity_threshold	0.43
min_num_docs	2000
epochs	200

Se puede observar que el comportamiento del clustering con K-means no es efectivo, incluso haciendo métodos de reducción de dimensionalidad. El modelo que mejor comportamiento alcanzó fue K-means con TF-IDF, sin embargo, los valores están por debajo del valor medio óptimo.

Esto puede ser por la consistencia de los datos, donde de cierta manera existe una agrupación diferente a la que se desea, por polaridad de sentimiento y en cambio se tiene algo relacionado por los temas con los que fue recuperada la información o simplemente porque el algoritmo no es el adecuado para este problema.

Las cuatro configuraciones anteriores de clustering también fueron realizadas para 5 clústeres con el fin de ver el comportamiento de los modelos, los tiempos de ejecución son más grandes para la mayoría. En cuanto a las métricas, también hay un aumento para algunas de ellas en ciertas configuraciones, lo que podría corroborar la idea anterior, de que no necesariamente las etiquetas estas relacionadas con el comportamiento de los clústeres.

4.1. Lbl2Vec

Lbl2Vec es un algoritmo no supervisado para problemas de clasificación y recuperación de documentos [13]. Genera vectores de etiquetas, documentos y vectores de palabras automáticamente mediante la definición de palabras claves

Tabla 4. LSTM accuracy.

	Train	Test
Accuracy	0.70	0.62

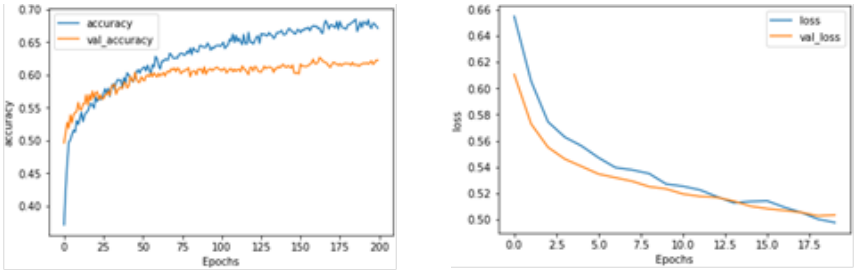


Fig. 7. Accuracy del modelo vs accuracy del conjunto de prueba y la pérdida del modelo vs la pérdida del conjunto de prueba.

(keywords) predefinidas manualmente. Está basado en la idea de que muchas keywords semánticamente similares pueden representar un tema o clase, en este caso en particular, un sentimiento.

En el primer paso, el algoritmo crea una incrustación conjunta de documentos y vectores de palabras. Una vez que los documentos y las palabras se incrustan en un espacio vectorial, el objetivo del algoritmo es aprender vectores de etiquetas a partir de palabras clave, estas previamente definidas, ocasionalmente de manera manual, donde cada conjunto de palabras clave representan un tema o clase. Finalmente, el algoritmo puede predecir la afiliación de documentos a una clase del vector de documento.

Para realizar el entrenamiento de este algoritmo se realizó una división del conjunto de datos, con el fin de evaluar el comportamiento del algoritmo mediante F1 Score. Por lo qué, se definió el 30% de observaciones del conjunto de datos de prueba (Test Set) y el 70% para el conjunto de entrenamiento (Train Set).

La lista de palabras claves fue construida a partir del Lexicón de Sentiment Analysis VADER (Valence Aware Dictionary and Sentiment Reasoner) [14], mediante el indexado de la categorización de palabras: positivas, neutras y negativas. VADER es una herramienta de análisis de sentimientos basada en reglas y léxico, específicamente para los sentimientos expresados en las redes sociales, pero que incluso puede funcionar bien en textos de otros dominios.

El algoritmo se entrenó con una lista de palabras clave que contiene la relación de la tabla 2, con un número determinado de palabras clave por clase, o polaridad de sentimiento, entre las palabras también se encuentran, conjuntos de caracteres asociados a emojis, cómo ‘:)’,’:|’ ‘:/’, ‘(-%’’,’) ':’, ‘)-':’, entre muchos otros más, de igual forma están etiquetados según la polaridad de sentimiento que representan.

Para los otros parámetros del algoritmo, se definieron los valores de la tabla 3. Los resultados de este algoritmo se evaluaron con la métrica, F1 score obteniendo un valor de 0.7403697617091208, que se puede considerar un valor aceptable tomando en cuenta, que es un algoritmo no supervisado, y qué el conjunto de datos fue recuperado de internet, es decir, que se ha tratado con los datos desde inicio a fin.

4.2. LSTM

LSTM es un algoritmo de aprendizaje supervisado, por lo que, se trabajó con el conjunto de datos etiquetado previamente con Azure. El primer paso por realizar para generar el modelo de LSTM es: cargar los datos, cómo estos ya han sido previamente procesados, están listos para ser utilizados en el modelo de LSTM. Primero es necesario hacer la división del conjunto de datos, en train y test.

Esta división también corresponde al 70% para el conjunto de train y 30% para el conjunto de test [15]. Antes de ingresar al modelo LSTM, los datos deben pasar por el proceso de padding y tokenización. Para más detalles del algoritmo, ver los detalles propuestos en [10].

Después de realizar pruebas con el desempeño de diferentes modelos de LSTM, este modelo fue el final, se realizó el entrenamiento con 200 épocas sobre el conjunto de datos de entrenamiento, y así mismo se realizó la validación con el conjunto de prueba, obteniendo cómo resultado los valores de la tabla 4, usando como métrica a Accuracy.

Se puede apreciar que el desempeño es mejor para el conjunto de entrenamiento, la diferencia no es mucha entre ambos valores, por lo que no se podría considerar como un problema de overfitting.

En la siguiente Fig.7 se muestra el comportamiento del accuracy y la pérdida a través del transcurso de las épocas en el entrenamiento del modelo. La curva de aprendizaje del conjunto de entrenamiento muestra que tan bien el modelo aprendió, mientras que la curva de aprendizaje del conjunto de validación representa que tan bien el modelo se comporta con nuevas observaciones.

Este comportamiento del modelo se puede deber a distintos factores, desde la integridad de los datos hasta la configuración del mismo modelo. Se realizaron pruebas incorporando más capas al modelo, pero se presentaron problemas de overfitting, lo mismo sucedió al aumentar el número de épocas.

El proceso de análisis y exploración de datos es uno de los procesos más relevantes de cualquier trabajo de aprendizaje máquina, la consistencia e integridad de los datos, es lo que define el funcionamiento de los modelos.

Para el enfoque supervisado, solo se exploró el uso de LSTM, El uso de un modelo pre-entrenado para la capa de incrustaciones de palabras o embedding layer proporcionó mayor estabilidad al modelo, esta representación permite obtener mejores relaciones semánticas de las palabras durante el entrenamiento.

Al hacer este pre-entranemiento, se obtuvieron los vectores de incrustaciones resultantes del vocabulario del conjunto de datos, junto con las relaciones semánticas que GloVe proporciona, lo que adiciona mayor robustez al modelo.

Además, la integración del modelo en la capa oculta, con la capa bidireccional de LSTM y las dos capas densas lograron obtener un Acuracy aceptable, es cierto que los valores óptimos podrían ser más altos, pero también es interesante destacar que este conjunto de datos es completamente nuevo y no hay trabajos realizados sobre el mismo tópico, como pudiera ser para cualquier otro conjunto de datos obtenido de algún repositorio dedicado a la investigación.

5. Conclusiones y trabajo futuro

El análisis de sentimientos es un problema del PLN, que como muchos otros más problemas retoman el estado del arte de otras disciplinas, lo que hace que sea una de las áreas desafiantes de la inteligencia artificial.

El principal propósito del desarrollo de este trabajo fue la exploración de distintas alternativas de aprendizaje máquina para el análisis de sentimientos en textos no etiquetados. Esto porque la mayoría de los datos disponibles se encuentran no procesados, sin etiquetas o clasificación, generalmente distribuidos en grandes bases de datos, o cómo es el caso de este trabajo, información recuperada desde medios digitales, como redes sociales.

Por lo que, es necesario realizar estos procesos manualmente, el etiquetamiento muchas de las veces con ayuda de expertos, siendo una tarea ardua y costosa. Con el fin de buscar alternativas para realizar este proceso de una manera automatizada, se propusieron algunas soluciones basadas en aprendizaje máquina para realizar el proceso de análisis de sentimientos de manera no supervisada.

Sin embargo, para tener un punto de evaluación mediante métricas más precisas fue necesario hacer una clasificación mediante el etiquetamiento de una polaridad de sentimiento con el uso de una herramienta de la nube, que además de permitir hacer una comparación con el comportamiento de los algoritmos no supervisados, también abriera la posibilidad de emplear técnicas supervisadas, con el fin de evaluar el comportamiento de los datos en ambos escenarios.

Los algoritmos no supervisados explorados es este trabajo de tesis fueron K-means y Lbl2Vec, los cuales permitieron explorar dos tipos de técnicas no supervisadas, Clustering analysis y Self-supervised learning, respectivamente. Las métricas empleadas para la evaluación de k means y MiniBachKmeans fueron al igual que los algoritmos, no supervisadas, para obtener comparaciones justas de los resultados.

Métricas de las cuales apenas se alcanzó un resultado por debajo de la media de los resultados óptimos, de lo que se puede concluir que el clustering no es una solución óptima para el análisis de sentimientos en concreto, puede ser una estrategia interesante para descubrir conocimiento de los datos, en la etapa de análisis y exploración de datos, e incluso en el proceso de data mining, o para otros problemas semejantes como la categorización de documentos, en casos donde las etiquetas no han sido asignadas, es decir, a partir de los clústeres obtenidos, se puede definir que etiquetas se les asigna a los datos.

Después de realizar pruebas con distintas configuraciones de parámetros para Lbl2Vec, se obtuvieron buenos resultados, según el valor alcanzado con F1 Score, obteniendo un valor aproximado de 0.74, considerando que el valor máximo es 1, los resultados fueron parcialmente buenos.

Adicionalmente a esta métrica también se realizó una comparación de los resultados obtenidos con este modelo respecto de los resultados de Azure, con lo que sorprendentemente se descubrió que las etiquetas de sentimiento si cambian en un 35% de las observaciones del conjunto de datos, pero esto es claramente comprensible, pues para empezar Lbl2Vec hace la clasificación con 3 polaridades de sentimiento, mientras que Azure lo realiza con 4. Además, muy posiblemente el algoritmo de Azure este entrenado con un vocabulario considerablemente más grande al creado en la implementación de Lbl2Vec.

Una LSTM, proporciona buenos resultados para un análisis de sentimientos basado en polaridad de sentimiento, y se puede decir que también seria efectivo para un análisis de sentimientos basado en emociones, en ese caso, el modelo se modificaría en la capa de salida, aumentando el número de neuronas de acuerdo con la relación de clases en las que se clasificarían las emociones, pero dado la integridad de este conjunto de datos hasta el momento, este experimento aun no es posible, pues sería necesario tener los datos etiquetados de acuerdo a emociones.

El aprendizaje no supervisado tiene muchos desafíos importantes en la investigación, sin embargo, hay muchos modelos y algoritmos desarrollados que permiten modelar los datos y obtener buenos resultados. Alternativas como el uso de Transformers, para entrenar modelos de análisis de sentimientos que pudieran ser más robustos, sin necesidad de tener datos etiquetados, al ser modelos pre-entrenados, por lo que, también se podrían obtener deducciones importantes y sería una buena implementación a futuro, que enriquezca este trabajo.

Por su parte el aprendizaje supervisado, provee resultados más precisos, además de que hay mucha información e investigación disponible sobre la cual trabajar. A diferencia del aprendizaje no supervisado para tareas de PLN. Por lo que, las mejores estrategias para tareas de PLN en general hasta el momento se derivan de técnicas supervisadas o semi- supervisadas.

Referencias

1. Shah, C.: A hands-on introduction to data science. Cambridge University Press (2020)
2. Masmoudi, A., Hamdi, J., Belguith, L. H.: Deep learning for sentiment analysis of Tunisian dialect. Computación y Sistemas, vol. 25, no. 1, pp. 129–148 (2021) doi: /10.13 053/cys-25-1-3472
3. Kokab, S. T., Asghar, S., Naz S.: Transformer-based deep learning models for the sentiment analysis of social media data. Array, Elsevier, vol. 14 (2022) doi: 10.1016/ j.array.2022.100157
4. Zhang, L., Wang, B., Liu, B.: Deep learning for sentiment analysis: A survey. WIREs Data Mining and Knowledge Discovery, vol. 8, no. 4 (2018) doi: 10.1002/widm.1253
5. Gao, L. Liu X., Yin, J.: Improved deep embedded clustering with local structure preservation. IJCAI'17: In: Proceedings of the 26th International Joint Conference on Artificial Intelligence, pp. 1753–1759 (2017)
6. Moolayil, J.: Learn Keras for deep neural networks: A fast-track approach to modern deep learning with Python, Apress Berkeley, CA (2018) doi: 10.1007/978-1-4842-4240-7
7. Kantardzic, M.: Data mining: Concepts, models, methods and algorithms. Wiley-IEEE Press, Chapters 1–18, pp. 360 (2003)
8. Olson, D. L.: Data mining models. Second Edition 2nd edition. Business Expert Press (2018)
9. Kwartler, T.: Text mining in practice with R. First Ed. John Wiley & Sons (2017) doi: 10.1002/9781119282105
10. Olivares L. J.: Estrategias para el análisis de sentimientos en textos extraídos de Twitter utilizando técnicas de aprendizaje profundo. Tesis de Licenciatura, FCC-BUAP (2022)
11. Embedding projector. Visualization of high-dimensional data. https://projector.tensor flow.org/. (2022)
12. Garcia A. J.: Comparativa de técnicas de balanceo de datos.: Aplicación a un caso real para la predicción de fuga de clientes. Tesis de Maestría. Universidad de Oviedo (2021)

13. Schopf, T., Braun, D. Matthes, F.: Lvl2Vec: An embedding-based approach for unsupervised document retrieval on predefined topics. In: Proceedings of the 17th International Conference on Web Information Systems and Technologies WEBIST′21, pp. 124–132 (2021) doi: 10.5220/0010710300003058
14. Hutto, C., Gilbert, E.: VADER: A parsimonious rule-based model for sentiment analysis of social media text. In: Eighth International AAAI Conference on Weblogs and Social Media, vol. 8, no. 1, pp. 216–225 (2014) doi: 10.1609/icwsm.v8i1.14550
15. Lindemann, B., Müller, T., Vietz, H., Jazdi, N., Weyrich, M.: A survey on long short-term memory networks for time series prediction. Procedia CIRP, vol. 99, pp. 650–655 (2021) doi: 10.1016/j.procir.2021.03.088

ISSN 1870-4069

Selección de características de representaciones de texto de BETO usando un algoritmo genético

Juan José Guzmán-Landa, José Clemente Hernández-Hernández,
Guillermo de Jesús Hoyos-Rivera, Efrén Mezura-Montes

Instituto de Investigaciones en Inteligencia Artificial,
Universidad Veracruzana,
México

zS21000453@estudiantes.uv.mx,
jclementehdzhdz@gmail.com, {ghoyos, emezura}@uv.mx

Resumen. El Procesamiento del Lenguaje Natural es un área que se está volviendo sumamente importante dentro de la investigación en Inteligencia Artificial, esto incluye el análisis de sentimientos, la traducción automática, y la generación de texto, entre otros. El análisis manual sobre el texto es un desafío significativo debido a la gran cantidad de datos que se generan a través de los medios sociales en la red; el análisis asistido por computadora es una opción viable. Recientemente, han emergido múltiples tareas para el manejo automático del texto, especialmente en el idioma inglés de las tareas previamente mencionadas. Los casos más representativos se encuentran dentro de las técnicas de Aprendizaje Profundo, específicamente aquellos relacionados con el modelo BERT y otros modelos de tipo Transformer. Dicho modelo genera un vector de 768 características para representar cada palabra o fragmento de palabra de manera numérica. El número de características usualmente tiene una ausencia de justificación y de descripción. Además, la mayoría de los trabajos de investigación en la clasificación de sentimientos se encuentran fuera del idioma español, y del uso de técnicas con modelos de tipo Transformer. Basado en lo antes mencionado, este trabajo propone hacer uso de un conjunto de datos en español, a través de un Algoritmo Genético para la selección de características en un enfoque de envoltura, con el fin de reducir el número de características de los vectores generados por un modelo entrenado en conjuntos de datos en español, BETO (BERT para el español), y obtener un subconjunto de ellas, asimismo, verificar si con el nuevo conjunto de características se puede obtener un buen desempeño en la clasificación de sentimientos. Los resultados obtenidos en una serie de experimentos indican un desempeño competitivo, en la tarea de clasificación de sentimientos, incluso con una representación mucho menor con respecto al total de las características originales.

Palabras clave: BERT, algoritmo genético, representaciones vectoriales, selección de características, reducción de dimensionalidad.

Feature Selection of BETO Token Embeddings Using a Genetic Algorithm

Abstract. Natural Language Processing is an area that is becoming increasingly important in Artificial Intelligence research, including sentiment analysis,

machine translation, and language understanding, among others. Handcrafted analysis over text is very challenging due to the large amounts of data generated through digital social networks; computer-assisted analysis is a viable option. Recently, multiple text-handling task proposals have emerged to tackle various assignments automatically, especially in the English language of the aforementioned tasks. Most representative cases are found in the Deep Learning techniques, specifically those related to BERT and other Transformer models. Such a model generates a continuous vector of 768 elements representing a word or a token. The number of characteristics usually has an absence of justification and description. In addition, most of the research works on sentiment classification are outside of the Spanish language and the use of Transformer-type modeling techniques. Based on the above, this work proposes to make use of a Spanish data set through a Genetic Algorithm for feature selection in a wrapper approach in order to reduce the number of features of the embeddings generated by BETO (BERT for Spanish), and to obtain a subset of them, also, to verify if with the new set of features, a good performance in sentiment classification can be obtained. The results obtained in a series of experiments indicate a competitive performance, in the sentiment classification task, even with a much smaller representation with respect to the total of the original features.

Keywords: BERT, genetic algorithm, token embeddings, feature selection, dimensionality reduction.

1. Introducción

El texto, visto por una máquina, es una secuencia de símbolos que no tienen significado alguno, dado que interpretar este tipo de recursos es una actividad inherentemente humana.

El texto puede ser fácilmente manipulado en diferentes formas con una máquina, por ejemplo, dividir el texto por caracteres, detectar espacios en blanco y saltos de línea, entre otros; también se puede cambiar la posición de los caracteres, reemplazar y cambiar el uso de mayúsculas y minúsculas, pero nada que pueda dar un significado al texto. Todo lo anterior es meramente sintáctico, y centrado en alteraciones morfológicas.

BERT [1] es una opción que permite ir más a profundidad de las tareas mencionadas previamente. Usando este modelo, se obtiene una representación vectorial del texto, es decir, el texto cambia a una forma numérica, por lo tanto, se cuenta con un procesamiento computacional más adecuado con respecto al significado del texto.

BERT también tiene una variante para el español llamado BETO [4]. Este último obtiene una representación numérica del texto, sólo que con un contexto y significado diferente debido al idioma usado.

La representación resultante es habitualmente generada usando técnicas de Aprendizaje Profundo (AP). En este caso en especial, el modelo está basado en el mecanismo de atención [11], el cual genera buenos resultados en un amplio rango de tareas.

Aún así, tiende a requerir bastantes recursos computacionales para el procesamiento. Por ejemplo, una oración de tan solo 50 palabras genera una matriz de 50×768, donde 768 es el número de valores continuos que representan a una palabra o token (fragmento de palabra) [13], y por lo tanto, esto se convierte en una entrada de 38,400 características. Este resultado puede impactar en el número de cálculos cuando el conjunto de oraciones es mayor.

Por esta razón, reducir el número de características que representan al texto se convierte en una tarea importante, eventualmente esperando lograr un similar, o incluso mejor desempeño que la representación original, al momento de realizar tareas de aprendizaje como la clasificación.

Este trabajo de investigación propone un enfoque de envoltura usando un Algoritmo Genético (AG) como un algoritmo de búsqueda y una Red Neuronal Artificial (RNA) como un clasificador para realizar la tarea de Selección de Características (SC), todo esto utilizando el conjunto de datos de reseñas de películas IMDb[1] para el idioma en español. La tarea de evaluación se centra en realizar Análisis de Sentimientos (AS) de dicho conjunto de datos.

Este artículo tiene como contribución, además de mostrar un estudio experimental sobre la reducción de características de las representaciones generadas por BETO, demostrar, que sólo algunas características son necesarias para resolver una tarea en específico, que para el caso de estudio actual es el AS.

El resto del trabajo de investigación está organizado de la siguiente forma: en la Sección 2, se presentan los trabajos relacionados con este estudio; en la Sección 3, se presenta en detalle el enfoque propuesto. La Sección 4 incluye la descripción de los datos utilizados, mientras que la Sección 5 muestra los experimentos y sus correspondientes resultados. Finalmente, en la Sección 6, se presentan las conclusiones y el trabajo futuro.

2. Trabajos relacionados

Como parte de los antecedentes, esta Sección describe algunos de los principales trabajos de investigación relacionados en el proceso de SC sobre representaciones vectoriales del texto.

En [10], se describe un proceso de SC con una representación de texto de tipo, Frecuencia de Termino - Frecuencia Inversa de Documento, por su siglas en inglés, TF-IDF. El mecanismo de SC reduce la dimensionalidad de la representación a través de un AG y el método de Análisis de Componentes Principales, por sus siglas en inglés, PCA.

Este último obtiene una representación a nivel de documentos para una tarea en específico, lo cual no permite utilizar esa representación a nivel de palabras, y por lo tanto, no puede ser orientado para otras tareas. Para trabajar otro tipo de tarea dentro del PLN, se rquiere otro proceso de SC. Los resultados finales muestran que después de correr el mecanismo de SC, se obtiene una mejor precisión de clasificación.

En [3], se propone mejorar la precisión de clasificación de textos de diagnósticos médicos mediante una SC con un AG para la reducción de dimensionalidad.

[1] https://www.kaggle.com/datasets/luisdiegofv97/imdb-dataset-of-50k-movie-reviews-spanish

Algorithm 1: Algoritmo Genético basado en envoltura para la Selección de Características de BETO

Data: Reseñas de películas representada por BETO
Result: El subconjunto de características con la mejor precisión de clasificación
$P \leftarrow$ Inicializar una población;
Calcular la aptitud de cada solución en P;
while *MAX_GEN no es alcanzado* **do**
    Seleccionar T soluciones de P usando la Selección por Torneo;
    Aplicar cruza a la solución en T;
    Mutar al descendiente generado después de la cruza;
    Calcular la aptitud de cada descendiente;
    Aplicar reemplazo de soluciones;
end

En este trabajo de investigación, la representación del texto no se especifica con claridad; se puede intuir que se utilizó una bolsa de palabras, conocido por sus siglas en inglés, BoW. Los resultados mostrados concluyen en una buena reducción de la dimensionalidad.

El trabajo de investigación descrito en [9] propone un mecanismo de SC para reducir la dimensionalidad de los vectores generados por el modelo Word2Vec. El proceso principal consiste en filtrar las características que destacan más de una categoría y están ausentes en otras. El método producido se compara con mecanismos de SC tradicionales. Lo resultados finales muestran un desempeño similar en ambos casos.

Los autores de [12] propusieron un mecanismo de SC antes de representar el texto a través de un modelo, que en este caso es BERT. Por ejemplo, obteniendo las palabras más representativas de un texto a través de un método como TF-IDF. Dado que BERT no se desempeña muy bien cuando la longitud del texto es muy grande, el proceso de SC descrito es benéfico para la clasificación de textos grandes.

A partir de la revisión de la literatura anterior, se puede observar que los trabajos de investigación se enfocan particularmente en BERT y Word2Vec, dejando sin explorar, de acuerdo a la revisón hecha por los autores, la rama del español que usa BETO y todo lo que eso implica.

Motivado por lo antes mencionado, este trabajo de investigación introduce un método de SC para las representaciones vectoriales de BETO, un modelo entrenado para el idioma español.

3. Enfoque propuesto

Para lograr el propósito de este trabajo de investigación, se consideró una SC basada en envoltura con un AG [14]. En este sentido el proceso de SC de tipo envoltura, incluye un clasificador que servirá como función de aptitud para evaluar la calidad de las características seleccionadas, mediante la precisión del modelo. Por otra parte, el proceso de un AG está basado en la evolución natural de las especies, y su proceso se puede observar en el Algoritmo 1.

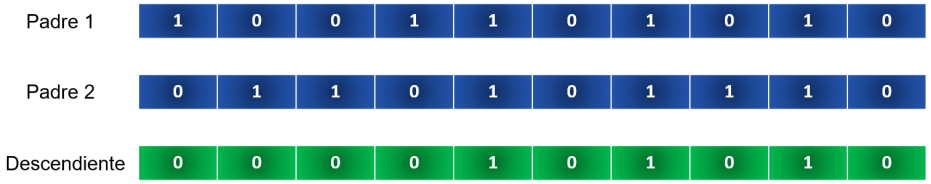


Fig. 1. Operador de cruza AND.

Fig. 2. Mutación simple modificada.

La complejidad del problema a resolver tiende a aumentar cuando el número de características disponible es grande. En el caso de estudio actual, se tienen 768 características, un número alto si una solución algorítmica de fuerza bruta trata de reducir esa dimensionalidad [6].

Por esto mismo, un AG es viable. Además, de que es posible que exista más de una solución buena para el problema en cuestión. Un AG explora el espacio de búsqueda y explota las zonas donde se encuentran mejores soluciones y, así, lograr una mejor calidad en la selección o reducción de características.

Una población inicial P de soluciones potenciales son generadas de manera aleatoria y evolucionadas durante un número específico de generaciones, usando operadores de variación como la cruza y mutación, los cuales están completamente ligados a la representación de la solución.

Después de algunas generaciones, es posible encontrar una solución competitiva con respecto a la función de aptitud, lo que representa la calidad de la solución del problema a resolver.

El AG utiliza una representación con cadenas binarias de tamaño igual al total de características de las representaciones vectoriales del conjunto de datos de interés, para definir una solución potencial.

Dicha representación indica qué características son seleccionadas o descartadas, por ejemplo, “1” indica que una característica será seleccionada, mientras que un “0” significa lo opuesto. Las características seleccionadas en esta cadena binaria vienen a partir de las representaciones vectoriales continuas generadas por BETO para cada palabra o token visto en el corpus de texto, que para el caso de BETO es 768, para el caso de Word2Vec, normalmente, es 300.

Considerando el enfoque de envoltura, un clasificador se utiliza como parte de la función de aptitud en el AG. Una solución potencial X es evaluada con la Ecuación 1:

$$aptitud(X) = w_1 * C(X) + w_2 * \frac{|X| - R(X)}{|X|}, \tag{1}$$

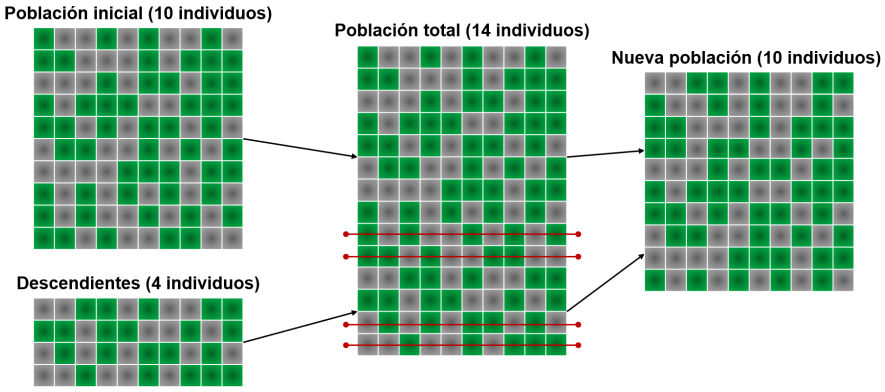


Fig. 3. Reemplazo basado en la aptitud; donde se puede ver las características seleccionadas en color verde y cada individuo representado por las filas de las matrices.

donde w_1 y w_2 son los pesos de importancia de cada factor en la función de aptitud, siendo $w_1 + w_2 = 1$. $C(X)$ es la precisión de clasificación usando las características seleccionadas en X, mientras que $R(X)$ es el número actual de las características seleccionadas en la solución X.

Es importante enfatizar que la polaridad de la clasificación es considerada a partir de las reseñas de películas en español, usando la RNA generada en el trabajo de investigación descrito en [5]. La descripción de dicha RNA se encuentra en la Sección 5.

La técnica de selección de padres adoptada en este trabajo es la selección mediante torneo. En este caso, el tamaño del torneo es definido por T, que es el número de soluciones tomadas de manera aleatoria de la población, de donde se selecciona la mejor solución, y además se considera realizar, posteriormente, la operación de cruza. El proceso de selección por torneo es llamada n veces, donde n es un número par.

Con base en la probabilidad de cruza, un par de padres son recombinados por un operador de cruza inspirado en el operador lógico AND, el cual demostró resultados competitivos en problemas de SC [3]. La Figura 1 detalla este operador.

Después de la cruza, se aplica una versión modificada de la mutación simple, con una probabilidad de mutación para cada descendiente que es definida previamente. Inspirado por el operador de mutación simple para la codificación de cadenas binarias, donde una simple posición está sujeta a la operación de bitflip, donde el valor de “1” se cambia por un “0” y vice-versa, el operador de mutación modificado usado aquí, siempre selecciona aleatoriamente una posición con un “1” y la cambia por un “0”, así como también, una posición con un “0” es aleatoriamente seleccionada y cambiada por un “1”.

De esta forma, el operador de mutación cambia las características seleccionadas, pero mantiene el número de características seleccionadas. En contraste, el operador de cruza es capaz de reducir el número de características. Un ejemplo visual del operador de mutación modificado puede ser visto en la Figura 2.

El último paso del AG se encuentra en el reemplazo (también conocido como selección de supervivencia o selección ambiental) para mantener el tamaño de la población fijo después de la creación de los descendientes.

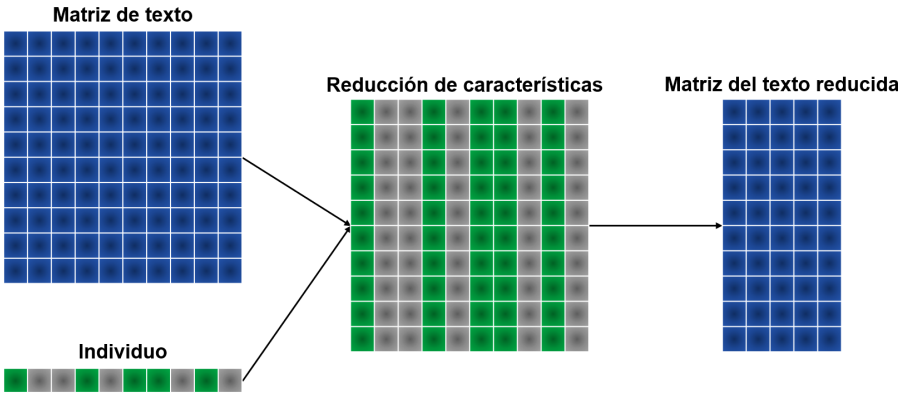


Fig. 4. Selección de Características con un individuo del AG.

A diferencia del reemplazo generacional tradicional en un AG canónico, donde la población actual es descartada, y todos los descendientes conforman la población de la siguiente generación, el proceso está inspirado por el reemplazo $(\mu + \lambda)$ de las Estrategias Evolutivas, donde la población actual y los descendientes se unen en un sólo conjunto, y los mejores $|P|$ individuos, basados en el valor de aptitud, se mantienen en la población para la siguiente generación, siendo $|P|$ el tamaño de la población original. En la Figura 3, se detalla este reemplazo.

4. Pre-procesamiento de los datos

Los datos utilizados en este estudio provienen del repositorio Kaggle, el cual contiene un conjunto de reseñas de películas en español IMDb. El corpus original está escrito en Inglés [7], y fue traducido por [2].

Dicha base de datos consiste de 50,000 reseñas de películas y es usada para la clasificación de sentimientos en dos clases; tiene 25,000 reseñas negativas y 25,000 positivas.

Un breve estudio del número de palabras en el corpus se llevó a cabo. Con ello se pudo concluir que existe una alta concentración de reseñas que tienen entre 125 y 375 palabras. Por esto mismo, el subconjunto de reseñas extraídas contienen entre 200 y 400 palabras, un porcentaje de ellas fueron utilizadas para evaluar el rendimiento del AG descrito en la sección anterior. El resultado de la extracción fue de 7,000 reseñas por cada clase, negativa y positiva.

Para tener los datos listos, fue necesario someter cada reseña a un proceso que consiste en pasar todo el texto a minúsculas y representar cada una de las palabras o tokens de las reseñas, en su respectiva representación vectorial generada por BETO. Lo anterior se realizó usando la biblioteca Bert-Tokenizer, que implementa el algoritmo de tokenización WordPiece [13], proporcionada por el lenguaje de programación Python.

Con los datos listos, el AG descrito en las secciones previas puede ser probado. En cuanto a la evaluación realizada por la función de aptitud en el enfoque basado en envoltura, el siguiente proceso se lleva a cabo para cada individuo de la población:

Tabla 1. Parámetros usados en cada experimento.

Parámetro	**Exp. 1**	**Exp. 2**	**Exp. 3**
Tamaño de la población	50	50	-
Número de reseñas (r)	40	100	100
MAX_GEN	10	10	-
Épocas de entrenamiento (e)	5	5	30
Palabras por reseña	30	50	50
w_1	0.2	0.2	-
w_2	0.8	0.8	-
Tamaño del torneo (T)	2	2	-
Número de padres (n)	20	20	-
Ejecuciones del AG	10	10	30

1. Las representaciones vectoriales generadas por BETO se van a concentrar en una matriz, siendo sus columnas, las que se van a seleccionar tomando en cuenta los valores de un individuo X, tal y como se explicó en la sección anterior.
2. Una RNA se utiliza como clasificador. Una solución X de la población del AG, con su correspondiente conjunto de características, entrena al clasificador durante un número de épocas e para un número de reseñas r, representado con BETO. Un ejemplo de la selección de características a partir de un individuo del AG se puede visualizar en la Figura 4, con una sola reseña. El proceso se realiza con todas las reseñas obtenidas después del estudio del número de palabras.
3. La RNA se entrena usando un subconjunto de las reseñas, tal y como fue descrito en la Sección 3. La red es evaluada usando otro subconjunto de las reseñas, y con ello se obtiene el rendimiento de la precisión.

5. Experimentos y resultados

Tres experimentos se llevaron a cabo para evaluar el desempeño del AG propuesto: (1) calibrar los parámetros para los operadores de cruza y mutación, (2) analizar la habilidad de la propuesta para reducir el número de características seleccionadas, y (3) comparar el desempeño de la RNA usando el número de características reducido contra el número de características original.

Todos los experimentos fueron realizados usando Python en su versión 3.10 y una computadora con Intel Xeon(R) CPU E542 de 8 núcleos, 2.80GHz, 6GB RAM, y Ubuntu 22.04.1 LTS. Es importante resaltar que todos los experimentos fueron limitados debido al recurso computacional disponible. Por lo tanto, el AG funciona con una porción de los datos seleccionados (ver Sección 4).

El número de reseñas y palabras usado en el experimento 1 es menor debido a su alto costo computacional. En la Tabla 1 se muestran los parámetros utilizados en cada experimento. La RNA adoptada en la función de evaluación del AG tiene la siguiente configuración: (1) una capa de entrada, con un número de neuronas a partir del producto entre la longitud del número de tokens usados y el número de características, e. g., 50 tokens $\times$ 768 o cualquier otro número características encontradas, y (2) dos neuronas como capa de salida, análogo con el número de clases del conjunto de datos.

Tabla 2. Experimento 1. Resultado de 10 ejecuciones del AG para cada combinación de valores de parámetros y así obtener las mejores probabilidades de cruza y mutación (%).

% de cruza	**% de mutación**	**Características (Avg, Max, Min, Med)**				**Precisión (Avg, Max, Min, Med)**				**Aptitud (Avg, Max, Min, Med)**			
0.2	0.2	5.1	6	1	6	0.78	0.83	0.75	0.75	0.951	0.965	0.943	0.945
0.2	0.4	4.3	6	1	6	0.80	0.91	0.75	0.79	0.955	0.982	0.943	0.954
0.2	0.6	8	37	1	6	0.82	0.91	0.75	0.83	0.956	0.977	0.943	0.960
0.2	0.8	8.8	37	3	6	0.79	0.91	0.75	0.75	0.949	0.960	0.943	0.944
0.4	0.2	4.4	6	1	5	0.80	0.91	0.75	0.79	0.955	0.979	0.943	0.954
0.4	**0.4**	5.8	15	1	6	0.82	0.91	0.75	0.83	**0.958**	0.977	0.945	0.960
0.4	0.6	4.5	14	1	4	0.75	0.83	0.66	0.75	0.945	0.963	0.929	0.944
0.4	0.8	7.5	15	5	6	0.80	0.91	0.75	0.79	0.952	0.967	0.943	0.948
0.6	0.2	3.9	6	1	4	0.76	0.83	0.75	0.75	0.949	0.960	0.943	0.946
0.6	0.4	4.5	6	1	6	0.80	0.83	0.75	0.83	0.955	0.964	0.943	0.960
0.6	0.6	4.8	15	1	4	0.77	0.83	0.75	0.75	0.950	0.965	0.934	0.946
0.6	0.8	6.2	15	1	5	0.79	0.83	0.75	0.79	0.951	0.963	0.943	0.950
0.8	0.2	10.3	19	1	7.5	0.84	1	0.75	0.83	0.957	0.981	0.943	0.958
0.8	0.4	7.1	19	1	3.5	0.79	1	0.75	0.75	0.950	0.980	0.934	0.948
0.8	0.6	4.5	15	1	3.5	0.74	0.83	0.66	0.75	0.943	0.960	0.929	0.945
0.8	0.8	3	10	1	2.5	0.73	0.75	0.66	0.75	0.943	0.948	0.931	0.946

Para el entrenamiento de la RNA se usa el optimizador Adam, la capa de salida utiliza como función de activación la función softmax, y la función de pérdida es, por su nombre en inglés, Negative Log Likelihood.

Considerando el primer experimento, se llevó a cabo una calibración de parámetros para obtener la configuración más adecuada para los operadores de cruza y mutación. Ambas probabilidades de cruza y mutación fueron variadas entre 0.2 y 0.8, y se ejecutaron 10 veces para cada combinación.

Para generar los resultados mostrados en la Tabla 2, se realizó lo siguiente: (1) para cada ejecución el mejor individuo fue seleccionado, y su correspondiente aptitud, número de características, y la precisión del clasificador fue recuperada; (2) usando los 10 resultados de los mejores individuos, se obtuvieron los valores de promedio, máximo y mínimo.

De acuerdo con los resultados de la función de aptitud en la Tabla 2, se puede observar que la mejor combinación de probabilidades es cuando ambos operadores de cruza y mutación tienen una probabilidad de 0.4.

De esta manera el resultado sugiere una calibración diferente a la que usualmente se encuentra en la literatura para una codificación binaria: una alta probabilidad de cruza y una baja probabilidad de mutación.

Para esta instancia de SC en particular, se requiere más exploración (i. e., valores de probabilidad de mutación más altos) en conjunto con una explotación moderada (valores de probabilidad de cruza más pequeños). El tamaño de la población y el número máximo de generaciones (condición de paro) fueron ajustados para mantener tiempos razonables de cada ejecución (alrededor de 90 minutos).

El AG fue ejecutado 10 veces para el segundo experimento con las probabilidades de cruza y mutación encontradas en el experimento previo (0.4 para ambas probabilidades de cruza y mutación). La tabla 3 muestra los resultados obtenidos. Cada fila de la tabla es una ejecución independiente.

Tabla 3. Experimento 2. Resultado de 10 ejecuciones del AG para reducir el número de características mientras se obtienen valores de clasificación competitivos.

Características (El mejor)	Precisión de clasificación (El mejor)	Aptitud (El mejor)
6	0.70	0.933
6	0.70	0.933
6	0.70	0.933
6	0.70	0.933
6	**0.76**	**0.947**
6	0.70	0.933
6	0.70	0.933
6	0.70	0.933
6	0.70	0.933
6	0.70	0.933

La solución encontrada para el AG propuesto tiene una reducción significativa del número de características (sólo 6), con una precisión de 0.76 y una aptitud de 0.947. Este resultado sugiere que existe información útil para la clasificación de reseñas en español sólo en un número reducido de características.

Finalmente, el tercer experimento compara el desempeño de la RNA usando el número total de características, 768, contra el mejor resultado, 6 en este caso, encontrado por el AG. La RNA usada para este experimento tiene la misma arquitectura que la usada previamente, además, la especificación del optimizador, la función de activación en la capa de salida y la función de pérdida, son los mismos que fueron usados en la función de aptitud del AG.

Con el objetivo de comparar ambas configuraciones de la red neuronal, 30 ejecuciones independientes fueron realizadas, respectivamente. Cada ejecución usa 70 reseñas para el entrenamiento y 30 para las pruebas de la red neuronal. Éstas generan 30 valores de precisión por cada configuración. La prueba estadística de Wilcoxon rank-sum, con un 95 % de confianza, fue ejecutada usando dichas precisiones de clasificación de las configuraciones.

Si el resultado del p-value es menor que 0.05, implica que los resultados tienen una diferencia significativa. De otra manera, las muestras no tienen una diferencia significativa. Los resultados se resumen en la Tabla 4.

El desempeño obtenido de la RNA usando el conjunto de características reducido por el AG, es mejor cuando se compara para la misma red pero con el conjunto de características original. Además, el número de operaciones de la red se reduce en consecuencia, y la reducción del tiempo de entrenamiento también es importante en comparación cuando se usan todas las características de representaciones vectoriales generadas por BETO. Finalmente, los resultados de las pruebas estadísticas indican una diferencia significativa entre los desempeños de ambas redes con un p-value de 1.2953e-06.

6. Conclusiones y trabajo futuro

En este artículo, se propuso una SC basado en envoltura con un AG para reducir el número de características de las representaciones vectoriales que provienen de la versión en español de BERT, BETO.

Tabla 4. Experimento 3. Resultados de la comparación de ambas configuraciones de la RNA con el conjunto de características original y el conjunto reducido.

Características	**Precisión (Min, Avg, Max)**			**Longitud de los datos**	**Operaciones de la red (incluye bias)**	**Tiempo de ejecución aproximado (Min, Avg, Max) en minutos**			**Wilcoxon p-value**
768	0.43	0.52	0.66	3,840,000	1,474,713,604	244	429	755	1.2953e-06
6	0.70	**0.69**	0.70	30,000	91,204	73	75	76	

El AG logró un desempeño muy competitivo, donde el número de características original fue reducido aproximadamente en un 99 %. Además, este número de características mejora la precisión de clasificación de la RNA en comparación de cuando se están usando las representaciones vectoriales con el número original de características.

El enfoque propuesto presenta una opción viable en el idioma español para generar modelos menos complejos, tratar con aquellas características de las representaciones vectoriales generadas por BETO, y de esta manera obtener una clasificación adecuada. De la misma forma, el costo computacional puede ser reducido en el proceso de entrenamiento. Sin embargo, es necesario mencionar que el costo computacional en el proceso de SC debe de ser considerado y reducido (ver el trabajo futuro al final de esta sección).

El trabajo futuro incluye: (1) considerar la implementación de la paralelización del AG para reducir el tiempo requerido en la función de aptitud, (2) aunado a lo anterior, se abre la posibilidad de utilizar un conjunto de datos nativo del español como el mencionado en el estudio [8], donde se propone una base de datos que contiene tres clases.

Si el número de clases aumenta, la complejidad de la RNA también lo hace. (3) Se puede incrementar el número de reseñas y tokens utilizados en el proceso de entrenamiento de cada individuo. (4) Por último, se propone utilizar clasificadores tradicionales de aprendizaje automático para comparar el desempeño contra la RNA presentada en este trabajo.

Agradecimientos. Los dos primeros autores agradecen el apoyo del Consejo Nacional de Ciencia y Tecnología (CONACyT), mediante una beca para realizar estudios de posgrado en la Universidad Veracruzana.

Referencias

1. Devlin, J., Chang, M. W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, vol. 1, pp. 4171–4186 (2019) doi: 10.18653/v1/n19-1423
2. Fernandez, L. D.: IMDB dataset of 50k movie reviews (spanish) https://www.kaggle.com/datasets/luisdiegofv97/imdb-dataset-of-50k-movie-reviews-spanish
3. Gnana-Singh, D. A. A., Leavline, E. J., Priyanka, R., Priya, P. P.: Dimensionality reduction using genetic algorithm for improving accuracy in medical diagnosis. International Journal

of Intelligent Systems and Applications, vol. 8, no. 1, pp. 67–73 (2016) doi: 10.5815/ijisa.2016.01.08
4. Gutiérrez-Fandiño, A., Armengol-Estapé, J., Pàmies, M., Llop-Palao, J., Silveira-Ocampo, J., Carrino, C. P., Gonzalez-Agirre, A., Armentano-Oller, C., Rodriguez-Penagos, C., Villegas, M.: MarIA: Spanish language models. Procesamiento del Lenguaje Natural, pp. 39–60 (2021) doi: 10.26342/2022-68-3
5. Hernández-Hernández, J. C., Mezura-Montes, E., Hoyos-Rivera, G. J., Rodríguez-López, O.: Neuroevolution for sentiment analysis in tweets written in mexican spanish. pp. 101–110 (2021) doi: 10.1007/978-3-030-77004-4_10
6. Khaire, U. M., Dhanalakshmi, R.: Stability of feature selection algorithm: A review. Journal of King Saud University - Computer and Information Sciences, vol. 34, no. 4, pp. 1060–1073 (2022) doi: 10.1016/j.jksuci.2019.06.012
7. Maas, A., Daly, R. E., Pham, P. T., Huang, D., Ng, A. Y., Potts, C.: Learning word vectors for sentiment analysis. pp. 142–150 (2011)
8. Pérez, J., Recart, E., Alves-Salgueiro, T., Furman, D., Fernández-Larrosa, P. N.: A spanish dataset for targeted sentiment analysis of political headlines. Electronic Journal of SADIO, vol. 22, no. 1, pp. 53–66 (2022)
9. Tian, W., Li, J., Li, H.: A method of feature selection based on Word2Vec in text categorization. In: 2018 37th Chinese Control Conference (CCC), pp. 9452–9455 (2018) doi: 10.23919/chicc.2018.8483345
10. Uğuz, H.: A two-stage feature selection method for text categorization by using information gain, principal component analysis and genetic algorithm. Knowledge-Based Systems, vol. 24, no. 7, pp. 1024–1032 (2011) doi: 10.1016/j.knosys.2011.04.014
11. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in Neural Information Processing Systems, vol. 30, pp. 5999–6009 (2017)
12. Wang, K., Huang, J., Liu, Y., Cao, B., Fan, J.: Combining feature selection methods with BERT: An in-depth experimental study of long text classification. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol. 349, pp. 567–582 (2021) doi: 10.1007/978-3-030-67537-0_34
13. Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, Ł., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., et al.: Google's neural machine translation system: Bridging the gap between human and machine translation. vol. 2, pp. 1–23 (2016) doi: 10.48550/arXiv.1609.08144
14. Xue, B., Zhang, M., Browne, W. N., Yao, X.: A survey on evolutionary computation approaches to feature selection. IEEE Transactions on Evolutionary Computation, vol. 20, no. 4, pp. 606–626 (2016) doi: 10.1109/TEVC.2015.2504420

ISSN 1870-4069

Elderly Mortality from COVID-19 in Mexico City: A Computational Intelligence Approach Based on Random Forests

Sinuhe Mazuti Osorio-Rivero, Guillermo Molero-Castillo,
Everardo Bárcenas, Rocío Aldeco-Pérez

Universidad Nacional Autónoma de México,
Facultad de Ingeniería,
Mexico

mazuti.96@gmail.com, guillermo.molero@ingenieria.unam.edu,
{ebarcenas, raldeco}@unam.mx

Abstract. Computational intelligence encompasses a wide variety of techniques and algorithms that are applied to address complex real-world problems. It is in the field of health where its use becomes significant to understand the behavior of a given disease, such as COVID-19. In this sense, there are sectors of the population that can easily develop a complication or die from such a condition, such as people over 60 years of age. As this is a growing and vulnerable population, it is important to make efforts to analyze the risks and effects that the elderly may present. This paper presents the implementation of a computational intelligence method for the prediction of mortality in older adults infected with SARS-CoV-2 in Mexico City. Open data, published and distributed by the Government of Mexico City, were used for this analysis. The results show that the variables with the greatest contribution of information for classification were: intubated, patient care, pneumonia and intensive care unit (ICU). This concludes that a hospitalized patient, who is admitted to the intensive care unit and requires intubation, has a high probability of being classified as 'dead'. In addition, the results show that variables related to the patient's age and sex are more important than variables associated with comorbidities.

Keywords: COVID-19, computational intelligence, elderly, random forests.

1 Introduction

Today, Artificial Intelligence (AI) encompasses a wide variety of subfields, ranging from general purpose areas, such as learning and perception [1], to more specific ones, such as applications based on computational intelligence, machine learning, deep learning, reinforcement, or mixed [2].

Specifically, Computational Intelligence (CI) concentrates a wide variety of techniques and algorithms that are applied to mimic human reasoning power in order to cope with complex real-world problems [3].

Today, it is evident the momentum that CI has taken in its application in different fields of human activity, such as health, security, education, biology, among others. Undoubtedly, at present, it is in the field of health where its use becomes significant to

understand the behavior of certain diseases, such as COVID-19, which is currently considered a pandemic affecting humanity.

The COVID-19 pandemic was caused by a new type of coronavirus, known as SARS-CoV-2. The first cases of infected people date back to December 2019 in the city of Wuhan, China.

Thus, from the first infections until October 2022, there are more than 615 million confirmed cases and more than 6.5 million deaths worldwide [4]. In the specific case of Mexico City, the object of study in this research, more than 1.74 million confirmed cases and more than 57 thousand deaths have been reported [5].

In this sense, there are sectors of the population that can easily develop a COVID-19 complication and even die. These sectors of the population are called vulnerable or at-risk groups. Among them are people 60 years of age or older, considered to be the elderly.

According to the Government of Mexico City, this vulnerable group of older adults is classified into two categories [6]: i) with comorbidity, which is characterized as people over 60 years of age, who have one or more diseases considered as factors of vulnerability; and ii) without comorbidity, which are identified as older adults without any disease or disorder that is considered a vulnerability.

There are certain characteristics, diseases and conditions in older adults that considerably affect their health status. For these reasons, they are considered one of the groups with the greatest vulnerability to COVID-19 disease [7]. This adult population can easily develop complications and even die from the disease.

Therefore, it is important to identify the patterns that condition their health status. The purpose of this is to provide useful information to understand and make better decisions about the management of pandemic disease. In addition, to provide a reflective analysis of the vulnerable group mentioned.

In addition to the above, the infection of older adults with the SARS-CoV-2 virus has a direct influence on society due to the way in which the epidemic risk traffic light COVID-19 operates, established by the Government of Mexico City through which the level of population risk and the increase or decrease of local activity is announced through colors, as well as the appropriate health safety measures for the reopening of work and educational activities and the use of public spaces [8].

On the other side, it is important to highlight the increase in the population of older adults in the last decade, where it went from 9.1% in 2010 to 12.0% in 2020. While the young population aged 0 to 17 years decreased from 35.4% in 2010 to 30.4% in 2020 [9]. This means that the population of older adults in Mexico is increasing.

For this reason, being a growing and vulnerable population, it is important to make efforts, from different perspectives, as is the case of computational intelligence, to analyze the risks and affectations that the elderly may present. This type of analysis is useful for identifying patterns in the form of trends in the population under analysis.

The aim of this research work was to implement a computational intelligence method, specifically random forests, for the classification of mortality in older adults infected with SARS-CoV-2 in Mexico City. For this, open data was used, published by the Government of Mexico City.

This paper is organized as follows: Section 2 presents the background of artificial and computational intelligence, COVID-19 in the adult population and the main related works; Section 3 describes the method established as a proposed solution; Section 4

presents the results obtained, based on a use case, such as the adult population; and Section 5 summarizes the main conclusions and future work.

2 Background

Artificial intelligence as an area of knowledge, proposed by John McCarthy in 1956, which refers to the science and engineering for the construction of intelligent machines, has faced multiple challenges during the last decades, due to the transition of states with emerging technologies, methods and algorithms [10, 11].

This makes traditional artificial intelligence incompatible with the increasing demands in search, optimization and resolution that problems require. The path from traditional to modern has enabled the emergence of better computational tools such as computational intelligence [11].

Through computational intelligence it is possible to build models, reasoning, machines and processes, based on structured and intelligent behaviors [11]. This type of intelligence adopts methods that tolerate incomplete, imprecise and uncertain knowledge in complex environments. In this way, they allow approximate, flexible, robust and efficient solutions [12]. Therefore, computational intelligence can be implemented to address problems that affect today's society [10].

Undoubtedly, to build inductive learning models, which base their function on the discovery of patterns from examples, one of the most used algorithms in computational intelligence are decision trees (DTs), through which prognosis and classification problems can be solved, aiming to build a hierarchical, efficient and scalable structure based on the conditions (variables) established in the data. The divide and conquer strategy are used for this purpose.

2.1 Random Forests

A tree is graphically represented by a set of nodes, leaves and branches. The main node or root is the attribute (variable) from which the classification process starts. The internal nodes correspond to each of the attribute conditions associated with a given problem. While each possible answer to the conditions is represented by a child node.

The branches coming out of each of these nodes are labeled with the possible values of the attribute. The final nodes or leaf nodes correspond to a decision, which coincides with some class (label) of the variable to be classified [13].

It is important to mention that sometimes decision trees are susceptible to overfitting, which means that they tend to learn very well from the training data, but their generalization may not be as good. One way to improve the generalization of decision trees is to combine several trees, known as random forests (RFs).

Random forests are widely used today. They aim to build an ensemble of decision trees, which when put together, what is actually happening is that they see different portions of the data. No tree uses all the training data, but each one is trained with different samples for the same problem.

By combining the results, the errors are compared with each other, and it has a prediction (forecast or classification) that generalizes better to the problem. Figure 1

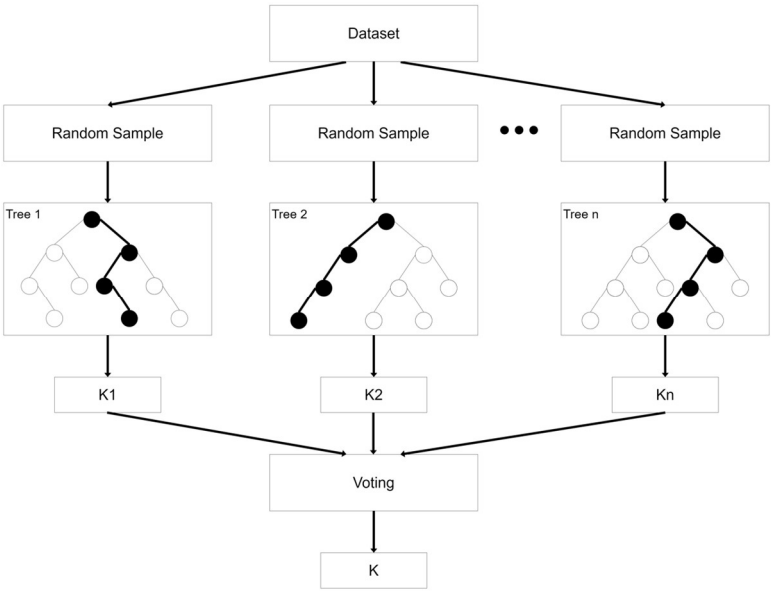


Fig. 1. Random forest general scheme.

shows the general scheme of the operation of random forests for classification, which consists of four steps:

1. Selection of random samples from the data set.
2. Construction of a decision tree for each sample and its respective result.
3. Voting (classification) based on the results obtained.
4. Selection of the result with the most votes (ranking).

2.2 Related Work

At present, one of the significant applications of random forests, due to the COVID-19 pandemic, is the classification of mortality in patients infected by the SARS-CoV-2 virus.

The objective is to classify characteristics (variables) of patients at risk of mortality due to this disease [14], as is the case of vulnerable groups, for example, the elderly.

In [15] it is stated that older people are more likely to contract COVID-19 and develop complications. These same authors mention that in the United States, through the Center for Disease Control and Prevalence (CDC) [16], it was identified that people over 65 years of age, accounted for 31% of SARS-CoV-2 infections, 45% of hospitalizations, 53% of admissions to intensive care units, and 80% of deaths caused by this infection.

Table 1. Related work.

Author	Description	Algorithm used	Limitations
Rami *et al.* (2022) [17]	Three experiments were performed using a data set of patients with COVID-19. Seven classification models were tested. The best performance was with the Bagging algorithm, with an accuracy of 83.55%.	Bagging, J48, Logistic Regression (LR), Random Forest (RF), Support Vector Machine (SVM), Naïve Bayes (NB), and Threshold Selector.	Records from 582 patients were used, of which 15 features were used for the first experiment, 6 for the second and 11 for the third.
Alves *et al.* (2021) [18]	Cases of Italian older adults hospitalized for COVID-19 were analyzed. Subsequently, the comorbidities of each group were analyzed. Dementia, diabetes, chronic kidney disease and high blood pressure were the main diseases involved in mortality.	Statistical analysis (Stata software).	The number of registered cases used ranged from 18 to 1591 patients.
Khan *et al.* (2021) [14]	The mortality rate of patients with COVID-19 was analyzed. Sociodemographic and clinical data from patients from different countries were used, and the models were evaluated for accuracy, precision, sensitivity and specificity. Deep Neural Networks model achieved a better prediction with 97% accuracy.	Deep Neural Network (DNN), Decision Tree (DT), Logistic Regression (LR), Random Forest (RF), Extreme Gradient Boosting (XGBoost), K-Nearest Neighbor (KNN).	It was used 103888 patient records from 45 countries, with the largest number from India (98632), and Philippines (4493). Nevertheless, there were less than 200 records for remaining countries
Cardoso *et al.* (2021) [19]	Algorithms were used to predict COVID-19 positive cases and find patterns in the databases of six districts (municipalities) in Argentina.	Fuzzy relationships and Artificial Neural Networks.	The artificial neural network model obtained an average error of 20%.
Akinnuwesi *et al.* (2021) [20]	Computational intelligence methods for the diagnosis of people with COVID-19 were analyzed. The performance of each algorithm was measured in terms of accuracy, precision, recall, balanced and accuracy. The methods with the best performance were MLP, FCM and DNN.	Logistic Regression (LR), Support Vector Machine (SVM), Naïve Byes (NB), Multilayer Perceptron (MLP), Fuzzy Cognitive Map (FCM) and Deep Neural Network (DNN).	Dataset limited to 600 records, of which 80% were used in training and 20% in testing.

Today, some researches have been identified that have provided knowledge about the COVID-19 disease by means of implementations of computational intelligence algorithms. These works have different approaches and objects of study. Table 1 summarizes five of these works, where the work performed, algorithms used, and limitations identified are briefly described.

Several related works use computational intelligence algorithms to address problems arising from the COVID-19 pandemic. It is important to highlight the use of the methods and tools provided by computational intelligence to understand the risks and affectations that certain vulnerable groups may suffer, as is the case of the elderly.

In relation to the related works identified, random forests were used as a classification algorithm for this research, with the purpose of taking advantage of the benefits and advantages of this computational intelligence approach, based on the divide and conquer strategy, with which explanatory rules are extracted, an advantage that other algorithms do not have by providing solutions without justification or explanation [10]. In addition, the random forest algorithm has shown high accuracy for classifying records of people with COVID-19 [17].

On the other hand, most of the identified related works perform the investigations with general population information sources. In contrast, in this research, the study focuses on the vulnerable group of older adults in Mexico City. This allows to learn about factors that condition the health status of this population, which is currently increasing and has a higher mortality rate due to COVID-19 [21].

In addition to the above, a data source containing 591352 records was used, corresponding to real cases captured during two years of the population under study. While the related works identified carried out their research with data sources of smaller period and size.

2.3 Motivation

Random forests are ideal for working with a large amount of data and multiple variables, due to the fact that it selects random samples to train classification or prognostic (regression) models, as the case may be [22].

In this sense, it is important to analyze the group of older adults, because it is one of the vulnerable groups that have been severely affected by COVID-19 disease. Increasing age conditions, a decrease in immune response and regenerative capacities, as well as a decrease in body mass index, functionality and an increase in comorbidities.

Given these situations, there is evidence of an increased risk of hospitalization and mortality compared to the general population [23]. Therefore, through this research, specialized technology is used to analyze the vulnerable group of older adults in Mexico City affected by COVID-19 disease.

3 Method

The solution method for the analysis of mortality of elderly in Mexico City, as a result of COVID-19, was divided into four stages (Figure 2): i) acquisition of data source, ii) variables selection, iii) classification, and iv) validation.

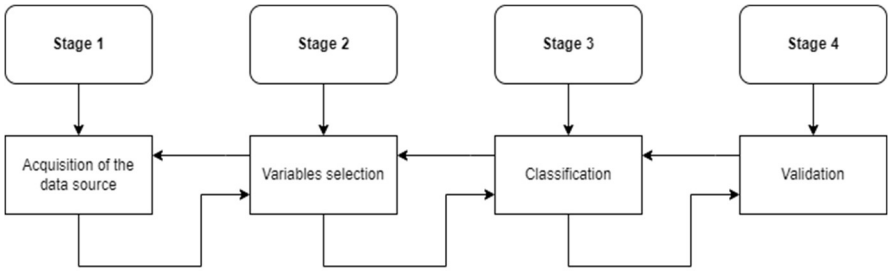


Fig. 2. Scheme of the method used as a proposed solution.

3.1 Data Source

In accordance with the decree published in the Official Journal of the Federation on February 20, 2015, which establishes the regulation on Open Data, the General Directorate of Epidemiology, made available to the general population the historical bases published since April 14, 2020 on cases associated with COVID-19 [24].

Thus, at present, there are sources of data on COVID-19 produced by different state, national and international entities, covering different populations. For example, for Mexico City alone, 24 data sources related to COVID-19 were found, such as historical hospital capacity, preliminary affluence in public transportation, inventories of contingency measures, among others.

In particular, for this research, data from the General Directorate of Epidemiology of the Ministry of Health of Mexico [24] were used, which are published periodically to facilitate all users access, use, reuse and redistribution of the same.

The purpose is to monitor possible cases of COVID-19 at the federal level, and specifically in Mexico City. Therefore, the period analyzed in this research comprises from April 14, 2020 to April 14, 2022, which represents 591352 real cases of COVID-19 in older adults in Mexico City, that is, records of two consecutive years.

3.2 Variables Selection

The original data source contains 40 variables, which provide information about the patient's case. However, not all the variables provide significant information for this research. So, an exploratory data analysis (EDA) was performed in order to make a careful selection of these variables.

Thus, from a selection of significant variables from the medical point of view and data analysis, a data source consisting of 20 variables was obtained, which are listed in Table 2. All the selected variables contain relevant information about the patient, characteristics of the medical treatment received and the evolution of the disease.

Through which patterns can be identified that allow an accurate classification of the mortality of older adults infected with SARS-CoV-2.

While other variables were discarded because they included redundant or irrelevant information, as is the case of sample taking (SAMPLING), result of the laboratory test applied to the patient to confirm the disease (LAB_RESULT), SARS-CoV-2 antigen sample (ANTINGEN_SAMPLING), to mention a few of these.

Table 2. Selected variables.

Item	Name	Description	Values
1	SEX	Identifies the sex of the patient.	1-Female, 2-Male, 99-Not Specified
2	PATIENT_CARE	Identifies the care type that patient received.	1-Ambulatory, 2-Hospitalized, 99-Not Specified
3	STATE	Identifies the situation (alive or dead) of the patient.	1-Alive 2-Dead
4	INTUBATED	Identifies whether the patient required intubation.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
5	PNEUMONIA	Identifies whether the patient was diagnosed with pneumonia.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
6	AGE	Identifies the patient's age.	Numeric
7	DIABETES	Identifies if the patient has a diagnosis of diabetes.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
8	COPD	Identifies if the patient has a diagnosis of Chronic Obstructive Pulmonary Disease (COPD).	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
9	ASTHMA	Identifies if the patient has a diagnosis of asthma.	1-Yes, 2-No, 97-Does not apply, 98-Ignore, 99-Not Specified
10	INMUSUPPR	Identifies if the patient has a diagnosis of immunosuppression.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
11	HYPERTENSION	Identifies if the patient has a diagnosis of hypertension	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
12	OTHER_DISEASES	Identifies whether the patient has a diagnosis of other diseases.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
13	CARDIOVASCULAR	Identifies whether the patient has a diagnosis of cardiovascular disease.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
14	OBESITY	Identifies if the patient has a diagnosis of obesity.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
15	CHRONIC_RENAL	Identifies if the patient has a diagnosis of chronic renal insufficiency.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
16	SMOKING	Identifies if the patient has a smoking habit.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
17	OTHER_CASE	Identifies if the patient had contact with any other case diagnosed with SARS-CoV-2.	1-Yes, 2-No, 97-Not Applicable, 98-Ignore, 99-Not Specified
18	ANTIGEN_RESULT	Identifies the result of the SARS-CoV-2 antigen sample analysis.	1-Positive SARS-CoV-2, 2-Negative SARS-CoV-2, 97- Not Applicable (Case without sample).
19	FINAL CLASSIFICATION	Identifies the classification of the Covid-19 test result: confirmed, invalid, not performed, suspect, and negative.	1-Confirmed by the Clinical Epidemiological Association, 2-Confirmed by the Ruling Committee, 3-Confirmed case, 4-Invalid by laboratory, 5-Not laboratory performed, 6- Suspect case, 7-Negative to SARS-COV-2
20	ICU	Identifies whether the patient required admission to an Intensive Care Unit (ICU).	1-Yes, 2-No, 97-Not applicable, 98-Unknown, 99-Not Specified

```
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, Y,
                                                                    test_size = 0.2,
                                                                    random_state = 0,
                                                                    shuffle = True)
```

Fig. 3. Separation of the data vectors for the training and testing of the algorithm.

```
ClassificationRF = RandomForestClassifier(random_state=0,
                                          max_depth=8,
                                          min_samples_leaf=2,
                                          min_samples_split=4)
ClassificationRF.fit(X_train, Y_train)
```

Fig. 4. Configuration of parameters for the operation of the algorithm.

Which are redundant due to the existence of another variable that indicates the final result of the antigen test for SARS-CoV-2 (ANTIGEN_RESULT).

3.3 Classification

The first criterion for classification using random forests is to calculate the entropy for all classes and attributes. Entropy is a measure of uncertainty (information) that is represented as follows:

$$Entropy(S) = l(S) = lnf(S) = \sum_{i=1}^{n} -p_i log_2 p_i, \quad (1)$$

where: S is a collection of elements (objects), and p_i is the probability of possible values. Subsequently, the best attribute is selected based on the information gain of each variable, which is represented as:

$$(S, A) = Entropy(S) - \sum_{v \in V(A)} \frac{|Sv|}{|S|} Entropy(Sv), \quad (2)$$

where: S is a collection of elements, A are the variables, Sv is a subset of elements, and $V(A)$ is the set of values that A can take.

Based on the above, for the classification of COVID-19 mortality in older adults in Mexico City, the cases were selected based on the following conditions: a) that the age was greater than 59 years, this because in Mexico City people from that age are considered older adults; b) that the medical unit and residence of the patient were Mexico City; and c) there is a class variable that allows identifying the state of life or death of people infected with SARS-CoV-2, that is, 'Alive' or 'Dead' cases, respectively.

Once the preparation and selection of variables was completed, a structure made up of 19 independent variables and one class variable (STATE), described in Table 2, was established as an input matrix for the operation of the algorithm. Subsequently, for the classification and validation process, the input matrix was divided into training and test data vectors, as shown in the code segment written in Python (Figure 3).

Finally, the maximum depth parameter that random forest estimators can reach (maximum depth of 8 levels) has been adjusted. The criteria of the minimum number of samples required before splitting a node (at least 4 elements) and the minimum number of samples in a leaf node (at least 2 elements) were also adjusted, as shown in Figure 4.

Table 3. Classification matrix.

		Classification	
		Dead	**Alive**
Real	**Dead**	4,047	2,783
	Alive	1,889	109,552

These parameters were adjusted in order to avoid overfitting the random forest estimators. In this sense, with the established configuration and the training and test data vectors, the algorithm was applied.

3.4 Validation

For the validation of the random forest, 113599 test cases were used (20% of records, new cases, which were not used in the training process), of which 4672 were misclassified. Table 3 shows the classification matrix obtained for the case study.

The classification matrix shows information about the performance of the algorithm, that it, it allows measuring the accuracy of the results obtained [25]. The classification matrix contains four types of results, which are: true positives, true negatives, false positives and false negatives.

4 Results

The variables with the greatest gain of information were identified, which are shown in Table 4. The variable with the greatest relevance for classifying cases was INTUBATED, which refers to whether it was necessary to intubate the patient.

Another important variable was PATIENT_CARE, which describes the care received by the patient (outpatient - inpatient). The third variable was ICU, whose objective is to identify whether the patient was admitted to an intensive care unit. The fourth variable was PNEUMONIA, which to identify whether the patient was diagnosed with pneumonia.

The following variables with lower percentages refer to the classification of the COVID-19 test result (FINAL_CLASSIFICATION), the result of the antigen sample analysis (ANTIGEN_RESULT) and whether the patient had contact with any other case diagnosed with the disease (OTHER_CASE). Subsequent variables refer to patient characteristics (age and sex) and comorbidities (renal failure, diabetes, cardiovascular disease and obesity).

It was observed in one of the estimators of the random forest that the main node was the variable UCI, which corroborates an important gain of information (22.53%). Other variables at the next levels (child nodes) of the estimators were the variables INTUBATED (27.38%), FINAL_CLASSIFICATION (7.1%), PNEUMONIA (13.32%) and ANTIGEN_RESULT (2.37%). These variables belong to the group that provides more information to classify the data.

Regarding validation, it was observed that the positive cases were the records correctly assigned by the algorithm in the 'Dead' category, as can be seen in Table 3, of

Table 4. Information gain.

Variable	Importance
INTUBATED	27.38 %
PATIENT_CARE	24.60 %
ICU	22.53 %
PNEUMONIA	13.32 %
FINAL_CLASSIFICATION	7.10 %
ANTIGEN_RESULT	2.37 %
OTHER_CASE	0.77 %
AGE	0.74 %
SEX	0.24 %
CHRONIC_RENAL	0.22 %
DIABETES	0.14 %
CARDIOVASCULAR	0.10 %
OBESITY	0.09 %
OTHER_DISEASES	0.09 %
COPD	0.08 %
SMOKING	0.08 %
HYPERTENSION	0.07 %
ASTHMA	0.05 %
INMUSUPPR	0.05 %

which 4047 correct classifications were obtained, considered as true positives. On the other side, true negatives were the cases classified as 'Alive', when they really belong to that category. In this case, 109552 correct classifications were obtained, considered true negatives. The true positive and negative results represent a successful classification of the analyzed cases in the category to which they belong.

The opposite happens with the results of false negatives and false positives, which are cases misclassified by the algorithm. False negatives represent cases of type 'Dead' that were classified in the 'Alive' category, in this case 2783 false negatives were obtained. On the other side, the false positive results are those cases that belong to the 'Alive' category and were classified as 'Dead'. In the classification matrix, 1889 false positives were observed.

As a result of the application and validation of the algorithm, 96.04% accuracy and 98% precision were obtained. On the other side, the average error was 3.96%, demonstrating a remarkable classification of survival and mortality of COVID-19 cases in older adults in Mexico City.

As part of the validation, the classification was also tested through a decision tree, whereby which an outstanding accuracy of 95.3% and an average precision of 97% were obtained. Nevertheless, through random forest, as observed, a better result was obtained both in accuracy (96.04%), precision (98%) and less error of misclassified cases (3.96%). Which represents that the solution through the random forest was better. Furthermore, it significantly reduces decision tree weaknesses such as overfitting.

5 Conclusions

A hospitalized-type patient is one who was admitted to the intensive care unit, required intubation, and was diagnosed with pneumonia. This type of patient has a high probability of being classified as 'Dead', since the variables with the greatest information gain for the classification were INTUBATED, PATIENT_CARE, ICU (Intensive Care Unit), and PNEUMONIA.

Variables related to the patient's age and sex have a higher degree of importance than variables associated with comorbidities, such as: obesity, hypertension, asthma, diabetes and others. For the case of persons classified as deceased, the variable AGE was an important separating condition, where deaths were mainly between 67.5 and 76.5 years old.

Based on the results obtained, the comorbidities with the highest degree of importance in the classification were chronic kidney disease, diabetes, cardiovascular disease and obesity. On the other side, the comorbidities with the lowest degree of importance were: hypertension, asthma and immunosuppression. The variables related to chronic obstructive pulmonary disease and smoking provided a low percentage of information gain.

The random forest algorithm obtained an average accuracy of 96.04% and a precision of 98%, which means that the mortality classification of older adults infected with SARS-CoV-2 in Mexico City was remarkable, whose cases were registered in two years, that is, from April 14, 2020 to April 14, 2022.

There are different vulnerable groups that are severely affected by the SARS-CoV-2 virus; in this case, efforts and attention were focused on the elderly. However, there are other sectors of society that need to be analyzed, such as indigenous groups, migrants, people with disabilities, among others.

As future work, to enrich the results obtained, it is intended to make a new analysis with updated information and new algorithms used in computational intelligence, such as support vector machines (SVM) and deep neural networks (DNN).

This may be important and of great interest due to the behavior of the pandemic due to the disease caused by COVID-19 and its impact on the population, especially on certain vulnerable groups.

References

1. Russell, S., Norvig, P.: Inteligencia artificial: Un enfoque moderno. Pearson Education, Prentice Hall (2004)
2. Kaplan, A., Haenlein, M.: Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence. Business Horizons, vol. 62, no. 1, pp. 15–25 (2019) doi: 10.1016/j.bushor.2018.08.004
3. Kumar, G., Jain, S., Singh, U.: Stock market forecasting using computational intelligence: A survey. Archives of Computational Methods in Engineering, vol. 28, no. 3, pp. 1069–1101 (2021) doi: 10.1007/s11831-020-09413-5
4. World Health Organization: Weekly epidemiological update on COVID-19 -5 October 2022. Emergency Situational Updates (2022) https://www.who.int/publications/m/item/weekly-epidemiological-update-on-covid-19---5-october-2022

5. Secretaría de Salud: Informe técnico semanal COVID-19 México, octubre 4, 2022. Subsecretaría de Prevención y Promoción de la Salud (2022) https://www.gob.mx/cms/uploads/attachment/file/765717/Informe_Tecnico_Semanal_COVID19_2022.10.04_1_.pdf
6. Gobierno de México: Criterios para las poblaciones en situación de vulnerabilidad que pueden desarrollar una complicación o morir por COVID-19 en la reapertura de actividades económicas en los centros de trabajo (2021)
7. Vega-Rivero, J. A., Ruvalcaba-Ledezma, J. C., Hernández-Pacheco, I., Acuña-Gurrola, M. F., López, L.: La salud de las personas adultas mayores durante la pandemia de COVID-19. Journal of Negative and No Positive Results, vol. 5, no. 7, pp. 726–739 (2020) doi: 10.19230/jonnpr.3772
8. Cortés, R., Dyer, D.: Lineamiento para la estimación de riesgos del semáforo por regiones COVID-19. Secretaria de Salud (2021)
9. INEGI: En México somos 126'014,024 habitantes: Censo de población y vivienda 2020 (2021)
10. Fulcher, J.: Computational intelligence: An introduction. In: Fulcher, J., Jain, L.C. (eds) Computational Intelligence: A Compendium. Studies in Computational Intelligence, Springer, Berlin, Heidelberg, vol. 115 (2008) doi: 10.1007/978-3-540-78293-3_1
11. Raj, J. S.: A comprehensive survey on the computational intelligence techniques and its applications. Journal of ISMAC, vol. 1. No. 3, pp. 147–159 (2019) doi: 10.36548/jismac.2019.3.002
12. Kruse, R., Borgelt, C., Braune, C., Mostaghim, S., Steinbrecher, M.: Computational Intelligence: A methodological introduction. Springer, London (2016)
13. Martínez, R., Ramírez, N., Mesa, H., Suárez, I., Trejo, M., León, P., Morales, S.: Árboles de decisión como herramienta en el diagnóstico médico. Revista Médica de la Universidad Veracruzana, vol. 9, no. 2, pp.19–24 (2009)
14. Khan, I., Aslam, N., Aljabri, M., Aljameel, S., Kamaleldin, M., Alshamrani, F., Chrouf, S.: Computational intelligence-based model for mortality rate prediction in COVID-19 patients. International Journal of Environmental Research and Public Health, vol. 18, no. 12, pp. 6429 (2021)
15. Leandro-Astorga, G., Calvo, I. B.: Infección por COVID-19 en población adulta mayor: recomendaciones para profesionales. Revista Médica de Costa Rica y Centroamérica, vol. 86, no. 629, pp. 44–50 (2021)
16. Centers for Disease Control and Prevention: Severe outcomes among patients with coronavirus disease 2019 (COVID-19)—United States (2020) https://www.cdc.gov/mmwr/volumes/69/wr/mm6912e2.htm
17. Mohammad, R. M. A., Aljabri, M., Aboulnour, M., Mirza, S., Alshobaik, A.: Classifying the mortality of people with underlying health conditions affected by COVID-19 using machine learning techniques. Applied Computational Intelligence and Soft Computing, vol. 2022, no. 3783058, pp. 1–12 (2022) doi: 10.1155/2022/3783058
18. Paulo-Alves, V., Golghetto-Casemiro, F., Gedeon de Araujo, B., de Souza-Lima, M. A., Silva de Oliveira, R., Tamires de Souza-Fernandes, F., Campos-Gomes, A. V., Gregori, D.: Factors associated with mortality among elderly people in the COVID-19 pandemic (SARS-CoV-2): A systematic review and meta-analysis. International Journal of Environmental Research and Public Health, vol. 18, no. 15 (2021) doi: 10.3390/ijerph18158008
19. Cardoso, A., Talame, L., Amor, M.: Aprendizaje automático aplicado a la pandemia del virus Covid-19 en Argentina. In: XXIII Workshop de Investigadores en Ciencias de la Computación, pp. 78–81(2021)
20. Akinnuwesi, B. A., Fashoto, S. G., Mbunge, E., Odumabo, A., Metfula, A. S., Mashwama, P. Uzoka, F. M., |Owolabi, O., Okpeku, M., Amusa, O. O.: Application of intelligence-based computational techniques for classification and early differential diagnosis of COVID-19

disease. Data Science and Management, vol. 4, pp. 10–18 (2021) doi: 10.1016/j.dsm.2021.12.001
21. Wang, X., Song, G., Yang, Z., Chen, R., Zheng, Y., Hu, H., Su, X., Chen, P.: Association between ageing population, median age, life expectancy and mortality in coronavirus disease (COVID-19). Aging (Albany NY), vol. 12, no. 24, pp. 24570–24578 (2020) doi: 10.18632/aging.104193
22. Merino, R. F., Chacón, C. I.: Bosques aleatorios como extensión de los árboles de clasificación con los programas R y Python. Interfases, no. 10, pp. 165–189 (2017)
23. Rodríguez, Y. L., López, L. A.: A propósito del artículo "COVID-19. De la patogenia a la elevada mortalidad en el adulto mayor y con comorbilidades". Revista Habanera de Ciencias Médicas, vol. 19, no. 4 (2020)
24. Gobierno de México: Datos abiertos de bases históricas. Dirección General de Epidemiología de la Secretaría de Salud, www.gob.mx/salud/documentos/datos-abiertos-bases-historicas-direccion-general-de-epidemiologia
25. Inca-Balseca, C. L., Paredes-Proaño, A. M., Cornejo-Reyes, P. J., Mena-Reinoso, Á. P.: Eficiencia de modelos de predicción de COVID-19 usando curvas ROC y matriz de confusión. Dominio de las Ciencias, vol. 8, no. 2, pp. 1442–1460 (2022)

Electronic edition
Available online: http://www.rcs.cic.ipn.mx

http://rcs.cic.ipn.mx

Centro de Investigación
en Computación