

EDUCACIÓN

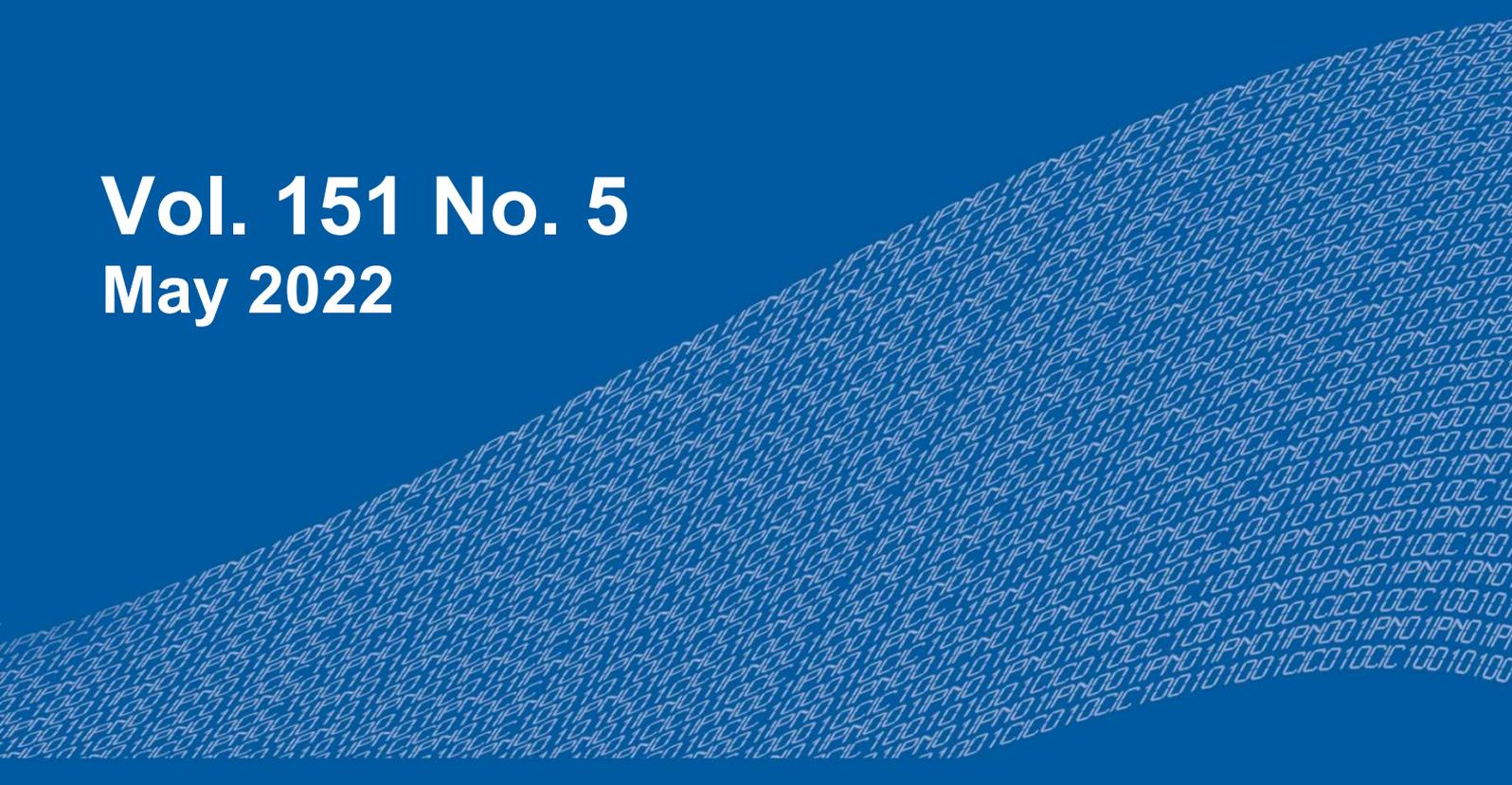
SECRETARÍA DE EDUCACIÓN PÚBLICA



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

Research in Computing Science

Vol. 151 No. 5
May 2022



Research in Computing Science

Series Editorial Board

Editors-in-Chief:

Grigori Sidorov, CIC-IPN, Mexico
Gerhard X. Ritter, University of Florida, USA
Jean Serra, Ecole des Mines de Paris, France
Ulises Cortés, UPC, Barcelona, Spain

Associate Editors:

Jesús Angulo, Ecole des Mines de Paris, France
Jihad El-Sana, Ben-Gurion Univ. of the Negev, Israel
Alexander Gelbukh, CIC-IPN, Mexico
Ioannis Kakadiaris, University of Houston, USA
Petros Maragos, Nat. Tech. Univ. of Athens, Greece
Julian Padget, University of Bath, UK
Mateo Valero, UPC, Barcelona, Spain
Olga Kolesnikova, ESCOM-IPN, Mexico
Rafael Guzmán, Univ. of Guanajuato, Mexico
Juan Manuel Torres Moreno, U. of Avignon, France

Editorial Coordination:

Alejandra Ramos Porras

Research in Computing Science, Año 21, Volumen 151, No. 5, mayo de 2022, 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 mayo de 2022.

Las opiniones expresadas por los autores no necesariamente reflejan la postura del editor de la publicación.

Queda estrictamente prohibida la reproducción total o parcial de los contenidos e imágenes de la publicación sin previa autorización del Instituto Politécnico Nacional.

Research in Computing Science, year 21, Volume 151, No. 5, May 2022, is published monthly by the Center for Computing Research of IPN.

The opinions expressed by the authors does not necessarily reflect the editor's posture.

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording or otherwise, without prior permission of Centre for Computing Research of the IPN.

Advances in Artificial Intelligence

Hiram Ponce (ed.)



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



Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2022

ISSN: in process

Copyright © Instituto Politécnico Nacional 2022
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>

The editors and the publisher of this journal have made their best effort in preparing this special issue, but make no warranty of any kind, expressed or implied, with regard to the information contained in this volume.

All rights reserved. No part of this publication may be reproduced, stored on a retrieval system or transmitted, in any form or by any means, including electronic, mechanical, photocopying, recording, or otherwise, without prior permission of the Instituto Politécnico Nacional, except for personal or classroom use provided that copies bear the full citation notice provided on the first page of each paper.

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

Table of Contents

	Page
Generador automático de resúmenes científicos en investigación turística <i>Miguel Á. Álvarez-Carmona, Ramón Aranda, Angel Diaz-Pacheco, José de Jesús Ceballos-Mejía</i>	5
Construcción de una red neuronal replicadora de datos y una aplicación a clonación de voz <i>Anel Ramírez Álvarez, Mari Anzures García, José Alejandro Rangel Huerta</i>	15
Clasificación bi-clase de canciones infantiles aplicando inteligencia artificial y procesamiento de lenguaje natural <i>David Soto Osorio, Jesús Jaime Moreno Escobar, Liliana Chanona- Hernández, Grigori Sidorov, César Jesús Núñez-Prado</i>	31
Asistente virtual de respuestas referentes a las becas ofertadas en ESIME Zacatenco <i>Sahian D Garcia Camacho, Brian J. Figueroa Acevedo, Liliana Chanona-Hernández, César Jesús Núñez-Prado, A. J. Salazar Cisneros, Grigori Sidorov</i>	39
Predicción de victoria en videojuego competitivo con Deep Learning <i>Juan Alberto González Gutiérrez, Cristian Ramón Hernández Ordoñez, Marco Antonio Vidal Flores, Melina Alexia Zárate Zenteno</i>	47
Segmentación probabilística de carriles mediante parametrización lineal de baja dimensión <i>Carlos Acuña, Mario Castelán, Gustavo Arechavaleta</i>	59
Diseño automático de filtros analógicos pasivos mediante un algoritmo genético <i>Aurora Torres Soto, María Dolores Torres Soto, Eunice Ponce de León, Braulio Jesús Montoya Padilla</i>	75
Desarrollo de un módulo para la prevención de la hipertensión usando el paradigma IoT y aprendizaje automático <i>Juan Antonio Alor-Hernandez, Nancy Aracely Cruz Ramos, Giner Alor-Hernández, José Luis Sánchez-Cervantes, Lisbeth Rodríguez-Mazahua</i>	91

Análisis del estado de ánimo en personas con tendencia suicida mediante testores típicos	105
<i>Daniel Alejandro Barajas Aranda, Aurora Torres Soto, María Dolores Torres Soto</i>	
Lógica tradicional para razonamiento no tradicional.....	115
<i>J.-Martín Castro-Manzano</i>	
Implementación de modelos de red neuronal profundos en GPUs embebidas para la clasificación de secuencias de fibrilación auricular	129
<i>Christian García-Aquino, Dante Mújica-Vargas, Juan Gabriel González-Serna, Manuel Matuz-Cruz</i>	
Uso de técnicas de Web Scraping para obtención automática de bases de datos en la Web.....	143
<i>Rogelio Mijangos-Espinosa, Alicia Martínez-Rebollar, Hugo Estrada-Esquivel, Yasmín Hernández-Pérez</i>	
Servicio para consumir datos de transporte público en tiempo real utilizando la plataforma FIWARE	159
<i>Luis Reyes-González, Alicia Martínez-Rebollar, Hugo Estrada- Esquivel, Yasmin Hernández-Pérez</i>	
Clasificación de haces de luz para el uso de energía por medio de redes neuronales	173
<i>Alexi Vasquez Rodriguez, Leticia Flores Pulido, Perfecto Malaquias Quintero, Carlos Perez Corona, Francisco Javier Albores Velazco</i>	
Técnicas metaheurísticas aplicadas a un mecanismo de 4 barras implementado en dispositivos de rehabilitación	183
<i>Francisco Javier Espinosa-Garcia, Rocío Ambrocio-Delgado, Esther Lugo-González, Alfredo Enrique Feria-De la Rosa, Alejandro Barragán-Bernardo, Lizeth Enriquez-Bravo</i>	
Detección del alcance de las negaciones en español usando Conditional Random Fields	199
<i>Antonio Tamayo, Jason Angel, Alexander Gelbukh</i>	

Generador automático de resúmenes científicos en investigación turística

Miguel Á. Álvarez-Carmona^{1,2}, Ramón Aranda^{1,2}, Angel Diaz-Pacheco¹,
José de Jesús Ceballos-Mejía³

¹Unidad de Transferencia Tecnológica Tepic,
Centro de Investigación Científica y de Educación Superior de Ensenada,
México

² Consejo Nacional de Ciencia y Tecnología,
México

³Instituto Tecnológico de Tepic,
México

malvarez@cicese.edu.mx

Resumen. Con la nueva oleada de las redes neuronales, arquitecturas basadas en aprendizaje profundo han tenido mucho éxito en la última década. Una de las principales aplicaciones que han tenido estas arquitecturas es la generación automática de texto, la cual ha ganado popularidad recientemente. Distintas tareas han tratado de ser resueltas como generador de canciones, *chatbots*, resúmenes automáticos, traductores, entre otros. Sin embargo pocos trabajos se han enfocado en generar textos científicos. En este trabajo se presenta un análisis de generación de *abstracts* de artículos científicos explorando distintos tipos de arquitecturas. Para este trabajo se recolectaron de 227 artículos de Procesamiento de lenguaje natural aplicado al sector turístico. Con esta colección se proponen diferentes tipos de *fine tuning* donde el mejor resultado es de 0.21 obtenido por GPT-3 según el coeficiente de Jaccard.

Palabras clave: Procesamiento de lenguaje natural, generación de textos científicos, turismo, RNN, LSTM, GPT-3.

Automatic Generator of Scientific Summaries in Tourism Research

Abstract. With the new wave of neural networks, architectures based on deep learning have been very successful in the last decade. One of the main applications that these architectures have had is the automatic generation of text, which has recently gained popularity. Different tasks have tried to be solved such as song generator, chatbots, automatic summaries, translators, among others. However, few works have focused on generating scientific texts. This paper

presents an analysis of the generation of abstracts of scientific articles exploring different types of architectures. For this work, 227 articles of Natural Language Processing applied to the tourism sector were collected. With this collection different types of fine tuning are proposed where the best result is 0.21 obtained by GPT-3 according to the Jaccard coefficient.

Keywords: Natural language processing, generation of scientific texts, tourism, RNN, LSTM, GPT-3.

1. Introducción

El Procesamiento de Lenguaje Natural (PLN) ha sido una herramienta que se ha implementado en diversas soluciones en los últimos años [2]. Áreas como medicina [19], turismo [3], psicología [5] o detección de agresividad [4], han sido algunos campos del conocimiento que han sacado ventaja del PLN. Dentro del PLN, una de las tareas que han ganado popularidad es la de generación automática de texto [6]. La tarea de generación automática de texto es el proceso de construir lenguaje natural a través de entradas no necesariamente lingüísticas [8]. La idea de estos sistemas es que dada una cadena de tokens, automáticamente se genere un texto que sea coherente y continúe con la secuencia que se introdujo.

En la figura 1 se muestra un demo de la api de Open-AI para generación de texto¹. Este demo es un chatbot capaz de interactuar con humanos. En la imagen se puede ver que la cadena de tokens inicial es una oración que pone un humano, y el texto resultante es una oración que responde a dicha cadena. El modelo general de generación de texto se puede expresar como $GT(T_i, M) = T_f$, donde GT es un sistema de generación de texto, T_i es un conjunto de tokens iniciales, M es un modelo entrenado para generar texto y T_f es el texto final que se genera a partir de los parámetros de la función GT . La tabla 1 muestra algunos ejemplos de T_i y T_f para distintas tareas.

Curiosamente, diversas tareas que utilizan métodos de generación automática de texto ven sus principales aplicaciones en el sector turístico. Por ejemplo, los *chatbots* se suelen utilizar para interactuar con turistas en agencias de viajes, hoteles o restaurantes o generando publicidad para destinos turísticos, o un traductor automático que ayuda a turistas en recorridos y vistas de museo. Por otro lado, pocos trabajos se han centrado en generar textos científicos de manera automática, normalmente, los trabajos se centran en textos comerciales, de interacción o artísticos.

Generar textos científicos se vuelve un reto puesto que existen muchos factores para poder crear este tipo de documentos como la hipótesis, variables, los experimentos y resultados, conclusiones entre otros, por lo que el modelo de solo utilizar un texto inicial parece insuficiente para atacar esta tarea. Por esta razón, en este trabajo se propone atacar el problema de generar resúmenes científicos de manera automática explorando diversas maneras de representar el texto inicial (T_i). Para llevar a cabo los experimentos se trabajó con todos los artículos publicados acerca de procesamiento de lenguaje natural en el sector turístico en la última década.

¹ <https://gpt3demo.com/>

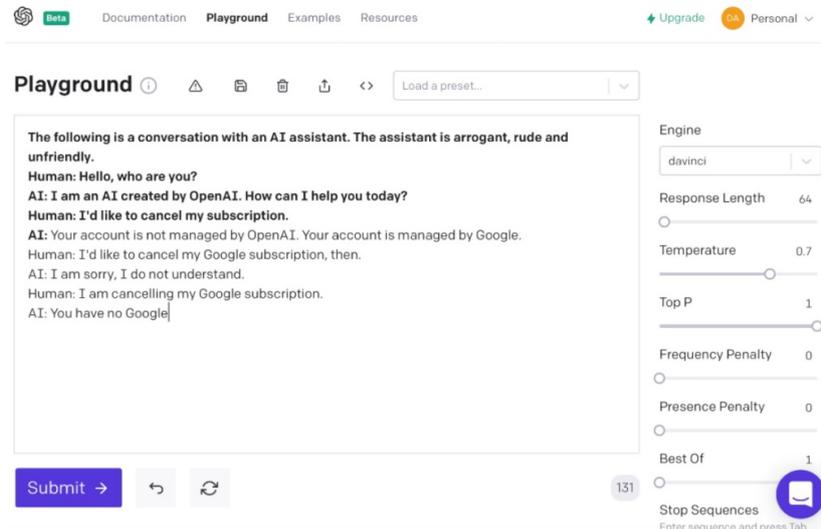


Fig. 1. Demo de Open-AI.

En este trabajo de investigación, se propone implementar diversos modelos de generación textual, planteando dar respuestas a las siguientes preguntas de investigación:

- ¿Cuál es la mejor manera de representar T_i de tal manera que sea posible generar textos científicos?
- ¿Cómo se comportan los modelos de generación textual con diferentes entradas de T_i para generar textos científicos?

El resto del documento está organizado de la siguiente manera: en la sección 2 se describen los principales modelos de generación automática de textos. En la sección 3 se detalla la metodología que se llevó a cabo para recopilar los artículos para este estudio y se proponen las distintas variantes para generar T_i . En la sección 4 se muestran los experimentos y resultados obtenidos. Finalmente, en la sección 5 se presentan las conclusiones y el trabajo a futuro derivado de esta investigación.

2. Generadores automáticos de texto

En los últimos años, se han propuesto diversos generadores de secuencias. En particular, los que más éxito han tenido son los basados en arquitecturas de aprendizaje profundo [14]. Un ejemplo de estas arquitecturas son las **Redes Neuronales Recurrentes (RNN)** por sus siglas en inglés). Estas redes son modelos dinámicos que se han utilizado para generar secuencias y se han aplicado a diversas tareas como la generación de texto [13], de música [16] y datos de movimiento [15].

Tabla 1. Ejemplos de sistemas de generación de texto.

Tarea	T_i	T_f	Ejemplo T_i	Ejemplo T_f
Chatbot	Petición de un usuario	Respuesta	Cancela mi suscripción	Subscripción cancelada
Traducción	Texto en el idioma original	Texto traducido	Hello i'm Gaby	Hola, soy Gaby
Textos artísticos	El inicio de un poema, canción, partitura, etc	El resto de la pieza	En un lugar de la mancha de	Cuyo nombre no quiero acordarme

Las RNN se entrenan con secuencias con el fin de predecir el elemento siguiente a dichas secuencias [20]. Después surgieron las arquitecturas **Memoria a Corto-Largo Plazo (LSTM)** por sus siglas en inglés [10]. LSTM es una arquitectura es un tipo de RNN, sin embargo, las LSTM tienen conexiones de retroalimentación. De esta manera, pueden procesar no solo datos individuales, sino también secuencias completas de datos como texto. Las redes LSTM se han aplicado en tareas como reconocimiento de escritura, reconocimiento de voz y generación de texto [18].

Una unidad LSTM común se compone de una celda, una puerta de entrada, una puerta de salida y una puerta de olvido. La celda recuerda valores en intervalos de tiempo arbitrarios y las tres puertas regulan el flujo de información dentro y fuera de la celda. Las redes LSTM son adecuadas para clasificar, procesar y hacer predicciones basadas en datos de series temporales, ya que puede haber retrasos de duración desconocida entre eventos importantes en una serie temporal, por lo que son ideales para la generación de texto a partir un una entrada textual.

En años más recientes, una de las mejores plataformas para generar texto ha sido **GPT-3** (Generative Pre-trained Transformer 3) [11]. Esta arquitectura fue construida sin pensar en alguna tarea en específico. El laboratorio OpenAI lo liberó para pruebas en agosto de 2020 y ha tenido resultados muy importantes comparado con modelos anteriores ajustados para tareas específicas. Esta arquitectura, está pre-entrenada en un conjunto de datos diez veces más grande que Wikipedia [9].

OpenAI desarrolló una interfaz de programación de aplicaciones donde los usuarios acceden a GPT-3 ingresando palabras en lenguaje humano, dando instrucciones sobre qué hacer y, opcionalmente, algunos ejemplos del resultado deseado. GPT-3 ha demostrado ser capaz de generar poesía o artículos indistinguibles de los autores humanos, código de computadora para interfaces web y descripciones de productos o trabajos [7]. Para este trabajo, utilizamos el modelo Ada de GPT-3.

3. Metodología

La principal idea detrás de este trabajo es generar textos científicos automáticamente, por esta razón, la metodología esta dividida en dos partes: i) la selección de una colección de artículos científicos con los cuales se puedan entrenar modelos de generación de textos, para este trabajo, se necesita una temática concreta para poder experimentar, por lo que se seleccionaron los trabajos publicados de Procesamiento de Lenguaje Natural aplicado al sector turístico en la última década, y ii) la generación de representaciones de T_i para poder aprender y generar textos como los de la base de datos recolectada. A continuación se presentan ambas partes.

3.1. Trabajos recolectados

Para esta revisión sistemática, nos basamos en la metodología *Preferred Reporting Items for Systematic Reviews and Meta-Analyses* (PRISMA) [17]. En este sentido, para comprender el estado actual de la investigación de PLN en turismo, todos los documentos fueron buscados en bases de datos en línea. Solo se consideró la investigación en inglés y se consultaron las bases de datos de documentos indexados desde el 1 de enero de 2010 hasta hasta el 31 de diciembre de 2021.

La búsqueda se realizó en las siguientes bases de datos académicas: ScienceDirect, Wiley Online Library, Sage Journals, the Web de Ciencias, Esmeralda y Scopus. La consulta de búsqueda consideró el uso de varias palabras clave² de *Turismo y Procesamiento del lenguaje natural* como *viaje, visita, hospitalidad, hotel, y destino* para el campo del turismo. Con respecto al PLN, las palabras clave empleadas fueron: *minería de texto, análisis de sentimientos, chatbot, UGC* (Contenido generado por el usuario), clasificación textual y *NLP*.

La concatenación de palabras clave se estableció con el operador *OR* y se empleó el operador *AND* para la intersección de las palabras clave de las áreas consideradas. La búsqueda retornó más de 500 trabajos. Se eliminaron 52 registros duplicados para tener 448 documentos almacenados para elegibilidad. Los criterios de exclusión rechazaron 98 registros. Se descartaron 112 registros por no parecer relevantes; 9 no fueron evaluados porque no estaban escritos en inglés (estaban en chino).

Finalmente, se incluyeron un total de 227 estudios en el proceso de revisión. Para la etapa de selección de estudios, dos revisores evaluaron cada uno de los documentos según los criterios de exclusión. Luego, todos los estudios no descartados fueron evaluados por separado por dos revisores diferentes. La inclusión se basó en una revisión de texto completo. Los desacuerdos se resolvieron por consenso.

Los datos extraídos de cada estudio son los siguientes: Palabras clave, año de publicación, país del primer autor, temas de turismo abordados, base de datos, idioma o idiomas de los datos, resultados obtenidos, número de registros en la base de datos, técnicas de pre-procesamiento de datos, algoritmos de selección de características y de aprendizaje, representaciones de datos, técnicas para la solución propuesta y métricas de evaluación.

3.2. Diferentes representaciones

Para poder generar un texto T_f a través de un Modelo M se necesita como un entrada un texto T_i . El objetivo de este trabajo es la generaciones de resúmenes científicos, específicamente en la temática de Procesamiento de Lenguaje Natural aplicado al sector turístico. En este sentido se propone aprovechar la revisión que se hizo en la sección 3.1 para entrenar modelos M_k y generar resúmenes con el estilo de la colección. Para poder responder las preguntas de investigación propuestas en la sección 1, se proponen tres maneras de construir T_i :

² Las palabras se usaron en inglés

$$T_{i_0} = \{T\}, \quad (1)$$

$$T_{i_1} = \{T, K\}, \quad (2)$$

$$T_{i_2} = \{T, K, R\}. \quad (3)$$

En las ecuaciones 1, 2 y 3 se representa las relaciones de diferentes T_i basadas en conjuntos. En la ecuación 1 el elemento T se refiere al Título de los artículos, es decir, los modelos M_k son entrenados únicamente con el título de un artículo. Por otro lado, la ecuación 2 agrega en el conjunto al elemento K , el cual se refiere a las *Keywords* que los propios autores de los artículos.

De esta manera se pretende que estas palabras clave agreguen información importante a los modelos M_k para aprender a escribir estos resúmenes. Finalmente, en la ecuación 3 se agrega al elemento R , el cual se refiere a los comentarios que agregaron los revisores que participaron para construir la base de datos de la sección 3.1.

Dentro de estos comentarios se agrega información como: país del primer autor, temas de turismo abordados, base de datos, idioma o idiomas de los datos, resultados obtenidos, número de registros en la base de datos, técnicas de pre-procesamiento de datos, algoritmos de selección de características y de aprendizaje, representaciones de datos, técnicas para la solución propuesta y métricas de evaluación.

Este tipo de información se agrega con la intención de que un modelo M_k sea capaz de detectar cuando en un resumen científico se agrega información crucial de la investigación. La idea es probar cada una de las representaciones con las diferentes arquitecturas presentadas en la sección 2. Es decir, se van a probar los algoritmos de RNN, LSTM y GPT-3 para observar su comportamiento en esta tarea.

4. Resultados

En esta sección se presentan los resultados de los modelos descritos en las secciones anteriores. Primero se debe definir la métrica para medir la calidad de los resultados. Existen diversas maneras para evaluar sistemas de generación de texto. Para este trabajo se propone una medida basada en similitud léxica. Se propone utilizar la medida de Jaccard para poder medir la similitud entre dos textos [1]. Jaccard esta definido como se muestra en la ecuación 4:

$$\text{Jaccard}(w(\text{abstract}), w(T_f)) = \frac{w(\text{abstract}) \cap w(T_f)}{w(\text{abstract}) \cup w(T_f)}, \quad (4)$$

donde $w(x)$ es una función que devuelve el conjunto de palabras de un texto x . De esta forma, el resultado de Jaccard es cero si no se comparte ninguna palabra entre los dos textos a comparar y el resultado es uno si los textos son idénticos [12]. Con esta medida se busca premiar el modelo que utilice palabras idénticas a la de los artículos originales. Para el entrenamiento de cada uno de los modelos, se hizo una separación de validación cruzada a 10 pliegues de los resúmenes de artículos científicos.

Tabla 2. Resumen de los resultados para las noticias separadas.

T_i	RNN	Std	Min	Max	LSTM	Std	Min	Max	GPT-3	Std	Min	Max
{ T }	0.01	0.10	0.00	0.12	0.04	0.09	0.00	0.21	0.14	0.07	0.05	0.56
{ T, K }	0.01	0.09	0.01	0.13	0.04	0.09	0.02	0.21	0.15	0.06	0.05	0.79
{ T, K, R }	0.03	0.09	0.04	0.25	0.06	0.08	0.07	0.45	0.21	0.05	0.09	0.98

En la tabla 2 se muestran los resultados promedio de Jaccard obtenidos por cada uno de los modelos para cada una de las representaciones. También, para cada arquitectura empleada se presenta la desviación estándar (Std), el valor mínimo de Jaccard obtenido (Min) y el máximo (Max). Se puede observar que el algoritmo de GPT-3 obtiene resultados considerablemente mejores que los obtenidos por las otras arquitecturas.

Este resultado se debe al pre-entrenamiento con el que cuenta dicha arquitectura. Esto la hace ideal para trabajar con pocos datos como es el caso. También es interesante notar que la diferencia entre agregar las keywords no aumenta considerablemente el resultado comparado con solo usar el título del artículo.

4.1. Análisis de los resultados

En esta sección se pretende responder a las preguntas de investigación planteadas para este trabajo.

P1: ¿Cuál es la mejor manera de representar T_i de tal manera que sea posible generar textos científicos?

Claramente agregar toda la información posible y relevante de una investigación aportará a los modelos con los que se trabaje. Se recomienda para generar resúmenes de artículos científicos estructurar las partes relevantes de la investigación y con eso alimentar a los modelos de generación de texto.

El mejor resultado se obtuvo con información como país del primer autor, temas de turismo abordados, base de datos, idioma o idiomas de los datos, resultados obtenidos, número de registros en la base de datos, técnicas de pre-procesamiento de datos, algoritmos de selección de características y de aprendizaje, representaciones de datos, técnicas para la solución propuesta y métricas de evaluación.

P2: ¿Cómo se comportan los modelos de generación textual con diferentes entradas de T_i para generar textos científicos?

El mejor resultado se obtuvo con el modelo de GPT-3. Claramente GPT-3 tiene la ventaja de contar con un pre entrenamiento masivo lo que lo hace ideal para tareas con pocos datos.

Es interesante ver como únicamente con el título se puede llegar a resultados mejores que el que obtienen arquitecturas como RNN y LSTM con toda la información disponible. Parece ser que los keywords no son tan importantes para obtener mejores resultados. En la tabla 3 se muestran algunos ejemplos de los resúmenes generados por GPT-3.

Tabla 3. Ejemplos de resúmenes reales contra resúmenes generados por GPT-3.

Resumen Original	Resumen Generado por GPT-3
<p>Traffic congestion is rapidly increasing in urban areas, particularly in mega cities. To date, there exist a few sensor network based systems to address this problem. However, these techniques are not suitable enough in terms of monitoring an entire transportation system and delivering emergency services when needed. These techniques require real-time data and intelligent ways to quickly determine traffic activity from useful information. In addition, these existing systems and websites on city transportation and travel rely on rating scores for different factors (e.g., safety, low crime rate, cleanliness, etc.).</p>	<p>Traffic congestion, a major problem in urban areas, portugal has one of the highest prevalence of driverthermal injuries in europe. furthermore, a combination of traffic control measures, such asrots and divided- plays, which provide a safety platform for engaged travelers, has been regarded as one of the main preventsesosmarismas of vibrant cities.</p>
<p>Sentiment Analysis is an active area of research and has presented promising results. There are several approaches for modeling that are capable of performing classifications with good accuracy. However, there is no approach that performs well in all contexts, and the nature of the corpus used can exert a great influence. This paper describes a research that presents a convolutional neural network approach to the Sentiment Analysis Applied to Hotel's Reviews, and performs a compar- ison with models previously executed on the same corpus.</p>	<p>Convolutional neural network (cnn) is a type of artificial neural network that can perform impressionist operations better than marchants' arendas. therefore, these are the operations that are performed by the cnn. this paper describes a research that presents a convolutional neural network approach to the sentiment analysis applied to hotel's reviews, and performs a comparison with models previously executed on the same corpus.</p>
<p>Original: Purpose – The purpose of this paper is to use Twitter analysis to explore diner perceptions of four types of Asian restaurants (Chinese, Japanese, Korean and Thai). Design/methodology/approach – Using 86,015 tweets referring to Asian restaurants, this research used text mining and sentiment analysis to find meaningful patterns, popular words and emotional states in opinions. Findings – Twitter users held mingled perceptions of different types of Asian restaurants. Sentiment analysis and ANOVA showed that the average sentiment scores for Chinese restaurants was significantly lower than the other three Asian restaurants. While most positive tweets referred to food quality, many negative tweets suggested problems associated with service quality or food culture. Research limitations/implications – This research provides a methodology that future researchers can use in applying social media analytics to explore major issues and extract sentiment information from text messages.</p>	<p>Originality/value–Limited research has been conducted applying social media analysis in hospitality research. This study fills a gap by using social media analytics with Twitter data to examine the Twitter users' thoughts and emotions for four different types of 86,015 tweets referring to Asian restaurants.</p>

5. Conclusiones y trabajo a futuro

En este trabajo de investigación se evaluaron diversas representaciones para resolver la tarea de generación automática de resúmenes científicos. Se probaron 3 tipos de representaciones iniciales para generar los resúmenes. Una basada en el título del artículos, otra agregando keywords y finalmente otra agregando datos importantes de la investigación. Para poder llevar a cabo los experimentos, se generó una base de datos con 227 artículos científicos de PLN en turismo. Los resultados indican que utilizar un método con pre-entrenamiento como GPT-3 puede obtener buen rendimiento en la tarea.

Estructurar la información relevante de la investigación como base de datos, resultados, país de origen, entre otros parece ser una buena opción para representar la información en esta tarea. Como trabajo a futuro se propone explorar cómo poder generar automáticamente textos científicos más largos como secciones completas de un artículo o inclusive el artículo completo.

Referencias

1. Álvarez-Carmona, M., Franco-Salvador, M., Villatoro-Tello, E., Montes-y Gómez, M., Rosso, P., Villaseñor-Pineda, L.: Semantically-informed distance and similarity measures for paraphrase plagiarism identification. *Journal of Intelligent & Fuzzy Systems*, vol. 34, no. 5, pp. 2983–2990 (2018) doi: 10.3233/JIFS-169483
2. Álvarez-Carmona, M., López-Monroy, P., Montes-y Gómez, M., Villaseñor-Pineda, L., Jair-Escalante, H.: INAOE's participation at PAN'15: Author profiling task. *Working Notes Papers of the CLEF*, pp. 103 (2015)
3. Álvarez-Carmona, M. Á., Aranda, R., Arce-Cardenas, S., Fajardo-Delgado, D., Guerrero-Rodríguez, R., López-Monroy, A. P., Martínez-Miranda, J., Pérez-Espinosa, H., Rodríguez-González, A. Y.: Overview of rest-mex at iberlef 2021: Recommendation system for text mexican tourism, (2021)
4. Aragon, M., Carmona, M. A., Montes, M., Escalante, H. J., Villaseñor-Pineda, L., Moctezuma, D.: Overview of mex-a3t at iberlef 2019: Authorship and aggressiveness analysis in mexican spanish tweets. pp. 478–494 (2019)
5. Aragon, M., López-Monroy, A., Gonzalez Gurrola, L. C., Montes, M.: Detecting depression in social media using fine-grained emotions. *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies*, pp. 1481–1486 (2019) doi: 10.18653/v1/N19-1151
6. Celikyilmaz, A., Clark, E., Gao, J.: Evaluation of text generation: A survey, (2020) doi: 10.48550/arXiv.2006.14799
7. Dale, R.: GPT-3: What's it good for? *Natural Language Engineering*, vol. 27, no. 1, pp. 113–118 (2021) doi: 10.1017/S1351324920000601
8. Daza, A., Calvo, H., Figueroa-Nazuno, J.: Automatic text generation by learning from literary structures. *Proceedings of the Fifth Workshop on Computational Linguistics for Literature*, pp. 9–19 (2016) doi: 10.18653/v1/W16-0202
9. Dehouche, N.: Plagiarism in the age of massive generative pre-trained transformers (GPT-3). *Ethics in Science and Environmental Politics*, vol. 21, pp. 17–23 (2021) doi: 10.3354/ese00195
10. Du, J., Vong, C. M., Chen, C.: Novel efficient RNN and LSTM-like architectures: Recurrent and gated broad learning systems and their applications for text classification. *IEEE transactions on cybernetics*, vol. 51, no. 3, pp. 1586–1597 (2021) doi: 10.1109/TCYB.2020.2969705
11. Floridi, L., Chiriatti, M.: GPT-3: Its nature, scope, limits, and consequences. *Minds and Machines*, vol. 30, no. 4, pp. 681–694 (2020) doi: 10.1007/s11023-020-09548-1
12. Guerrero-Rodríguez, R., Álvarez-Carmona, M. Á., Aranda, R., López-Monroy, A. P.: Studying online travel reviews related to tourist attractions using NLP methods: the case of Guanajuato, Mexico. *Current Issues in Tourism*, pp. 1–16 (2021) doi: 10.1080/13683500.2021.2007227
13. Iqbal, T., Qureshi, S.: The survey: Text generation models in deep learning. *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 6, pp. 2515–2528 (2022) doi: 10.1016/j.jksuci.2020.04.001

14. Mishra, M., Nayak, J., Naik, B., Abraham, A.: Deep learning in electrical utility industry: A comprehensive review of a decade of research. *Engineering Applications of Artificial Intelligence*, vol. 96 (2020) doi: 10.1016/j.engappai.2020.104000
15. Ramchandran, A., Sangaiah, A. K.: Unsupervised deep learning system for local anomaly event detection in crowded scenes. *Multimedia Tools and Applications*, vol. 79, no. 47, pp. 35275–35295 (2020) doi: 10.1007/s11042-019-7702-5
16. Santoso Gunawan, A. A., Phan Iman, A., Suhartono, D.: Automatic music generator using recurrent neural network. *International Journal of Computational Intelligence Systems*, vol. 13, no. 1, pp. 645–654 (2020) doi: 10.2991/ijcis.d.200519.001
17. Selçuk, A. A.: A guide for systematic reviews: PRISMA. *Turkish archives of otorhinolaryngology*, vol. 57, no. 1, pp. 57–58 (2019) doi: 10.5152/tao.2019.4058
18. Shewalkar, A., Nyavanandi, D., Ludwig, S.: Performance evaluation of deep neural networks applied to speech recognition: RNN, LSTM and GRU. *Journal of Artificial Intelligence and Soft Computing Research*, vol. 9, no. 4, pp. 235–245 (2019) doi: 10.2478/jaiscr-2019-0006
19. Voytovich, L., Greenberg, C.: Natural language processing: Practical applications in medicine and investigation of contextual autocomplete. *Machine Learning in Clinical Neuroscience*, pp. 207–214 (2022) doi: 10.1007/978-3-030-85292-4_24
20. Wang, H., Qin, Z., Wan, T.: Text generation based on generative adversarial nets with latent variables. pp. 92–103. Springer (2017) doi: 10.48550/arXiv.1712.00170

Construcción de una red neuronal replicadora de datos y una aplicación a clonación de voz

Anel Ramírez Álvarez, Mario Anzures García,
José Alejandro Rangel Huerta

Benemérita Universidad Autónoma de Puebla, México

anel.ramirez.al@gmail.com, marioanzuresg@gmail.com,
jose.rangelhuerta@viep.com.mx

Resumen. Las redes neuronales son sistemas de ecuaciones que nos permiten modelar comportamientos acordes a su modelo de construcción y entrenamiento. Como sabemos las redes neuronales realizan una pequeña emulación del comportamiento del cerebro humano, gracias a que forman nuevas conexiones internas en función de su aprendizaje. En este artículo de investigación se realiza la construcción de una red neuronal capaz de replicar datos mediante la compensación de diferencias entre los datos, esta construcción de red está inspirada en los sistemas de clonación de voz, donde para cierto timbre de voz se busca llegar a otro timbre de voz distinto, presentando así una red neuronal capaz de clonar datos. Por último, se presenta una aplicación de esta construcción neuronal a una clonación de voz a pequeña escala, teniendo en cuenta que en general una red neural aprende sola, no se tendrá que solucionar un problema tan complejo como lo es la detección de patrones bajo criterios estáticos que reconstruyan una voz, sino que la misma red es capaz de aprender a llegar al resultado deseado siendo entrenada. Este sistema, aunque prometedor, tiene limitantes dado la cantidad de información a la que debe acceder, por lo que se realiza una propuesta de evaluación para la red neuronal bajo la clonación de voz.

Palabras clave: Red neuronal, extracción automática de patrones, replicar datos, red neuronal simétrica, entrenamiento, compensar diferencia entre datos, clonación de voz, deep learning.

Construction of a Data Replicating Neural Network and an Application to Voice Cloning

Abstract. Neural networks are systems of equations that allow us to model behaviors according to their construction and training model. As we know, neural networks perform a small emulation of the behavior of the human brain, thanks

to the fact that they form new internal connections based on their learning. In this research article, the construction of a neural network capable of replicating data by compensating for differences between the data is carried out. This network construction is inspired by voice cloning systems, where for a certain voice timbre it is sought to reach another different timbre of voice, thus presenting a neural network capable of cloning data. Finally, an application of this neural construction to small-scale voice cloning is presented, considering that in general a neural network learns by itself, it will not have to solve a problem as complex as pattern detection based on criteria. Static that reconstructs a voice, but the network itself is capable of learning to reach the desired result by being trained. This system, although promising, has limitations given the amount of information it must access, so an evaluation proposal is made for the neural network under voice cloning.

Keywords: Neural network, automatic pattern extraction, replicate data, symmetric neural network, training, compensate difference between data, voice cloning, deep learning.

1. Introducción

El uso de Deep Learning, como lo son redes neuronales, resulta útil ya que solucionar problemas complejos donde la única ventaja es la consistencia en los datos es de gran utilidad ya que estas encuentran solos los patrones necesarios para converger en la solución requerida. Algunos de estos problemas complejos donde las redes neuronales trabajan eficientemente son en la clasificación de imágenes y textos, reconstrucción de imágenes, predicción de sucesos, procesos de control, entre otros.

Estos sistemas, aunque versátiles, tiene sus limitantes por la cantidad de información a la que debe acceder para poder lograr dicho fin, por ejemplo, las personas aprendemos de experiencias, cuanto más experiencia en un trabajo tengamos mejor podría ser nuestra respuesta a dicha labor, las redes neuronales funcionan de forma similar, cuanto más información recibe más precisa es su respuesta ante el problema específico en el cual se diseñó, es decir, el aprendizaje profundo se beneficia enormemente del acceso a grandes cantidades de datos.

Otro aspecto a tener en cuenta cuando trabajamos con redes neuronales es el diseño de su construcción; el número de neuronas en la capa de entrada, capas ocultas y neuronas de la capa de salida, ya que según sea su forma de construcción podremos modelaremos soluciones a problemas específicos, por ejemplo, para los ámbitos de clasificación tenemos que, para x número de neuronas de entrada, según sea el tamaño del dato a clasificar, solo tendremos cierto N número de salidas según el número de valores a identificar.

Ahora abordaremos la construcción de una red replicadora de datos, donde para x entradas tendremos el mismo número de salidas. Esta propuesta de construcción surgió como inspiración de los sistemas de clonación de voz (voice cloning), que son tecnologías desarrolladas que buscan imitar voces, estos sistemas se encuentran ya muy

presentes en la actualidad siendo usados a nivel comercial ya desde hace años por múltiples empresas de tecnología. Aunque también se plantea la cuestión de los posibles usos malintencionados como copias digitales de los patrones de voz, las cuales podrían burlar los complejos sistemas de seguridad.

Distintas medidas están siendo tomadas para combatir los intentos de aquellos que quieran usar de forma maliciosa las grabaciones. Estos sistemas desarrollados por empresas de tecnología han logrado grandes avances en este rubro por lo que es importante tomar a consideración sus avances y aportaciones. En la actualidad estos sistemas usan técnicas de deep learning, procesamiento de lenguaje natural, acoplamiento a voces artificiales, y/o muchas otras técnicas, además del uso de tecnologías que den soporte a grandes cantidades de información a las cuales necesitan acceder. A continuación, daremos un marco de contexto de estas tecnologías, en la metodología abordaremos el proceso de solución al problema planteado y concluyendo en la aplicación de esta propuesta de solución.

1.1. Modelos de inteligencia artificial

En ciencias de la computación, [1] una máquina inteligente ideal es un agente flexible que percibe su entorno y lleva a cabo acciones que maximicen sus posibilidades de éxito en algún objetivo o tarea. Coloquialmente, el término inteligencia artificial se aplica cuando una máquina imita las funciones cognitivas que los humanos asocian con otras mentes humanas, como, por ejemplo: percibir, razonar, aprender y resolver problemas.

Varios ejemplos se encuentran en el área de control de sistemas, planificación automática, la habilidad de responder a diagnósticos y a consultas de los consumidores, reconocimiento de escritura, reconocimiento del habla y reconocimiento de patrones. Los sistemas de IA actualmente son parte de la rutina en campos como economía, medicina, ingeniería, el transporte, las comunicaciones y la milicia, y se ha usado en gran variedad de aplicaciones de software, juegos de estrategia, como ajedrez de computador, y otros videojuegos.

El entrenamiento de un modelo de red neuronal en esencia significa seleccionar un modelo de la serie de modelos permitidos (o, en un bayesiano marco, la determinación de una distribución en el conjunto de modelos permitidos) que minimiza el criterio de costo. Hay numerosos algoritmos disponibles para la formación de los modelos de redes neuronales; la mayoría de ellos puede ser vista como una aplicación directa de la teoría de optimización y la estimación estadística.

1.2. Redes neuronales

El objetivo de la red neuronal [2] es resolver los problemas de la misma manera que el cerebro humano, aunque las redes neuronales son más abstractas. Cada neurona está conectada con otras a través de unos enlaces. En estos enlaces el valor de salida de la neurona anterior es multiplicado por un valor de peso.

Estos pesos en los enlaces pueden incrementar o inhibir el estado de activación de las neuronas adyacentes. Del mismo modo, a la salida de la neurona, puede existir una

función limitadora o umbral, que modifica el valor resultado o impone un límite que no se debe sobrepasar antes de propagarse a otra neurona.

Esta función se conoce como función de activación. Estos sistemas aprenden y se forman a sí mismos, en lugar de ser programados de forma explícita, y sobresalen en áreas donde la detección de soluciones o características es difícil de expresar con la programación convencional.

Para realizar este aprendizaje automático, normalmente, se intenta minimizar una función de pérdida que evalúa la red en su total. Los valores de los pesos de las neuronas se van actualizando, buscando reducir el valor de la función de pérdida. Este proceso se realiza mediante la propagación hacia atrás.

1.3. Aprendizaje de una red neuronal

Lo que ha atraído el mayor interés en las redes neuronales es la posibilidad de aprendizaje. Dada una determinada tarea a resolver, el aprendizaje [3] consiste en utilizar un conjunto de observaciones para encontrar la cual resuelve la tarea de alguna forma óptima. Hay tres grandes paradigmas de aprendizaje, cada uno correspondiente a una tarea de aprendizaje abstracto en particular.

Estos son el aprendizaje supervisado, el aprendizaje no supervisado y el aprendizaje por refuerzo. Al igual que calcular una fórmula, es cómo funcionan las redes neuronales, éstas se basan en el cálculo de sumas ponderadas las cuales al calcular el valor de los coeficientes son capaces de acoplarse a nuevos comportamientos de sistemas en análisis. A continuación, se mostrará la solución al problema a la construcción de red neuronal propuesta.

2. Trabajos relacionados

Existen ya muchos sistemas desarrollados que han logrado grandes avances en el rubro de la clonación de voz por lo que es importante tomar a consideración sus avances y aportaciones. Algunos de ellos son Lyrebird una empresa emergente con sede en Montreal fue lanzada en 2017. Su objetivo: utilizar la inteligencia artificial para "crear las voces artificiales más realistas del mundo".

Su software se alimenta con y así decir lo que quiera, con la misma voz. El modelo de aprendizaje automático se creó para determinar los factores que hacen que cada voz sea única. A esto lo llaman el ADN de la voz. Cada vez que prueban una nueva voz, el algoritmo averigua en qué se diferencia de las otras voces de nuestra base de datos y en qué se parece. Su inteligencia artificial basada en redes neuronales y algunos petaflops de potencia informática de alto rendimiento trabajando detrás de escena para darle a una persona o marca una voz (casi) real.

Existen también desarrollos creados por investigadores de IA con artículos publicados [4] y otros como el gigante chino Baidu, quien con sólo enunciar una oración su sistema es capaz de crear una copia sintética de la voz humana. Hasta no hace mucho la industria de la clonación de voces se concentraba en un nicho de mercado

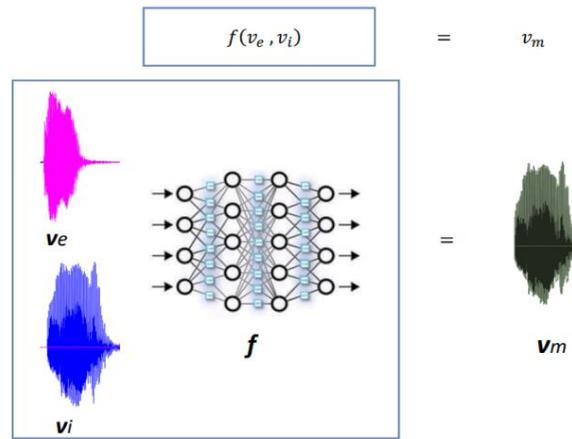


Fig. 1. Diagrama de representación de la ecuación 1.

muy reducido que buscaba atender las necesidades de aquellos que por motivos médicos o particulares [5].

3. Método

Como se vio en el apartado anterior esta construcción de red neuronal propuesta surge como inspiración de los sistemas de voice cloning, que usualmente utilizan en todo su proceso algún sistema de deep learning basados en una representación del conocimiento por medio de listas de datos de la voz. Para plantear una metodología, primero planteamos los requerimientos iniciales del Sistema, que son los siguientes: procesamiento de grandes cantidades de datos, sistema sometido a entrenamiento, sistema que converge en una solución, moldeable a modificaciones requeridas por el sistema, someter la información a modificaciones e iterar hasta encontrar la solución.

3.1. Propuesta de red neuronal

Se propone la construcción de una red simétrica de una sola capa y tres interconexiones para cada neurona, llamada en la literatura como Red Neuronal Monocapa. A continuación, se explica el motivo de cada uno de los parámetros mencionados que son requeridos por el sistema y el diagrama general de la red. La construcción general de la RN se puede reducir a la siguiente ecuación:

$$f(v_e[\text{amplitud}], v_i[\text{amplitud}]) = v_m[\text{amplitud}], \quad (1)$$

donde f representa la construcción general de la RN en función de v_e (voz de entrada) y v_i (voz imitada) para tener como salida la voz imitada. La ecuación anterior requiere

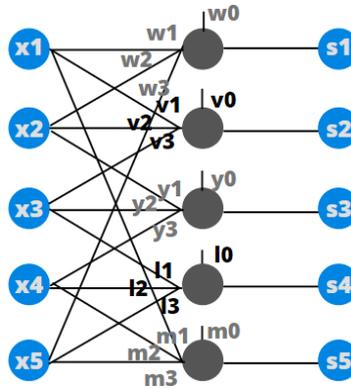


Fig. 2. Red mono capa simétrica de 5 neuronas.

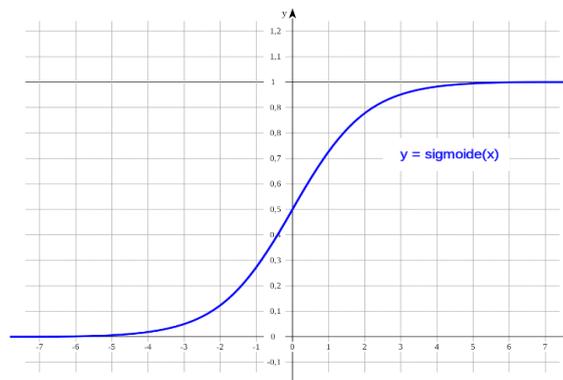


Fig. 3. Función de activación Sigmoide.

de una simetría del tamaño de número de datos necesarios, es decir, que el tamaño de audios debe ser del mismo número de muestras y sujetas al mismo tiempo de muestreo, por lo cual se fijó este requerimiento para la construcción de la RN.

La ecuación anterior f (voz de entrada, voz a imitar) se desglosa por la RN con la totalidad de las sumas ponderadas que conforman la red. Para explicar en un nivel más detallado se muestra a continuación un diagrama de dicha ecuación.

A continuación, se explicará por puntos la construcción de la red a partir de las entradas y salidas requeridas por el sistema $f()$:

- Ya que buscamos crear un modelo en el que tendremos entradas y salidas de las mismas dimensiones, se propone construir una red simétrica que no modifique el tamaño de la red.
- Ya que para manipular datos simples sólo necesitamos una capa se propone el uso de una red monocapa. Para evitar así el uso de más variables que resultarían innecesarias para los resultados.
- El motivo de las tres interconexiones es por motivos de ahorro de variables.

Con esto en mente se construyó una red mono capa simétrica de 5 neuronas, es decir, que requerirá 5 datos de entrada vi , 5 datos de salida vm y 5 datos a los que se desea imitar vi .

Para definir la función de activación lo realizamos en función del resultado que necesitamos. Ya que las señales son normalizadas, es decir, el resultado está en el rango de $[0,1]$ en pasos de hasta 0.1 décimas. Se propone el uso de la función sigmoide. Es muy importante decir que el peso de wi , vi , yi , li y mi . Se definió a partir de la siguiente ecuación: $-5 = w(0.1)$, ver la gráfica siguiente ya que la sigmoide comienza a converger en 1 y 0 para los valores de 5 y -5 respectivamente. despejando entremos que el peso es de $w = 50$ para ambos rangos entonces definimos los pesos en el intervalo de $[-50, 50]$. Donde tenemos que $y = f(x) = s$; es decir, que al tener valores en x de -5 a 5 tendremos como resultado para s un rango de valores de 0 a 1, que son los valores de nuestras señales de audio a variar.

Teniendo ya definida la función de activación a usar y los pesos que podrán ir recorriendo todos los valores para s de 0 a 1. Se muestran a continuación las ecuaciones que forman el diagrama de la Fig.2. Las ecuaciones siguientes son obtenidas de la generalización de la regla delta.

Ecuaciones requeridas para el sistema de red neuronal:

$$S_1 = f(w_1x_1 + w_2x_2 + w_3x_5 + w_0), \quad (2)$$

$$S_2 = f(v_1x_1 + v_2x_2 + v_3x_3 + v_0), \quad (3)$$

$$S_3 = f(y_1x_2 + y_2x_3 + y_3x_4 + y_0), \quad (4)$$

$$S_4 = f(l_1x_3 + l_2x_4 + l_3x_5 + l_0), \quad (5)$$

$$S_5 = f(m_1x_1 + m_2x_4 + m_3x_5 + m_0). \quad (6)$$

Ecuaciones para el cálculo del error:

$$\Delta_{S_1} = SD_1 - S_1, \quad (7)$$

$$\Delta_{S_2} = SD_2 - S_2, \quad (8)$$

$$\Delta_{S_3} = SD_3 - S_3, \quad (9)$$

$$\Delta_{S_4} = SD_4 - S_4, \quad (10)$$

$$\Delta_{S_5} = SD_5 - S_5. \quad (11)$$

Pesos o coeficientes de las sumas ponderadas:

$$w_0 = w_0 + \Delta_{S_1}, \quad (12)$$

$$w_1 = w_1 + \Delta_{S_1}x_1, \quad (13)$$

Tabla 1. Tabla de datos para la propuesta de red neuronal del sistema (Fig. 9).

Datos iniciales (voz original) v_e X_i	Datos para imitar (voz a imitar) v_i SD_i	Datos de resultado de la Red Neuronal (voz imitada) v_m S_i
0.2 a 0.1e	0.4 ar	0.40000000000000013
0.4 a	0.5 er	0.50000000000000002
0.5 a	0.55	0.55000000000000007
0.9 a	0	0.0020049821385272233 ≈ 0
0.7 a	0	0.0022907827937611648 ≈ 0

$$w_2 = w_2 + \Delta_{S_1} x_2, \quad (14)$$

$$w_3 = w_3 + \Delta_{S_1} x_5, \quad (15)$$

$$v_0 = v_0 + \Delta_{S_2}, \quad (16)$$

$$v_1 = v_1 + \Delta_{S_2} x_1, \quad (17)$$

$$v_2 = v_2 + \Delta_{S_2} x_2, \quad (18)$$

$$v_3 = v_3 + \Delta_{S_2} x_3, \quad (19)$$

$$y_0 = y_0 + \Delta_{S_3}, \quad (20)$$

$$y_1 = y_1 + \Delta_{S_3} x_2, \quad (21)$$

$$y_2 = y_2 + \Delta_{S_3} x_3, \quad (22)$$

$$y_3 = y_3 + \Delta_{S_3} x_4, \quad (23)$$

$$m_0 = m_0 + \Delta_{S_5}, \quad (24)$$

$$m_1 = m_1 + \Delta_{S_5} x_1, \quad (25)$$

$$m_2 = m_2 + \Delta_{S_5} x_4, \quad (26)$$

$$m_3 = m_3 + \Delta_{S_5} x_5, \quad (27)$$

$$l_0 = l_0 + \Delta_{S_4}, \quad (28)$$

$$l_1 = l_1 + \Delta_{S_4} x_3, \quad (29)$$

$$l_2 = l_2 + \Delta_{S_4} x_4, \quad (30)$$

$$l_3 = l_3 + \Delta_{S_4} x_5. \quad (31)$$

Condiciones de paro, como bien sabemos son a variar:

1. $\sum_{i=0}^{N=5} \Delta S_i \leq 0.0001$,
2. Número de iteraciones < 1000 .

Este proceso se implementó en Python y se puso a prueba para ciertos datos de ejemplo, dándonos los resultados siguientes.

Los Pesos de w_i , vi , yi , li y mi calculados por la Red Neuronal, se comprobaron en el sistema y posteriormente en la función de activación para validar la información resultante. Hay que recordar que los datos solución en modelos de redes neuronales cambian con una nueva corrida de la red, es decir, nuevos datos que de igual forma convergerán a la solución deseada. No son datos de solución únicos para algún modelo de entrenamiento, por ejemplo, en este caso para nuestra Tabla 1, habrá distintas combinaciones de valores para w_i , vi , yi , li que solucionan estos datos de entrenamiento.

La propuesta de construcción se sometió a múltiples pruebas como los de la Tabla 1 y vemos como los resultados son los deseados, de igual forma se probó para más datos dando resultados igualmente favorables, a continuación, probaremos esta construcción para una cantidad mayor de datos, para el fin de este artículo, con datos de señales de audio.

3.2. Modificación de datos

El procesamiento digital de señales o DSP (sigla en inglés de digital signal processing) es la manipulación matemática de una señal de información para modificarla o mejorarla en algún sentido. A continuación, se presentan las técnicas más comunes para la modificación de sonoridad de señales de audio.

3.3. Manipulación de volumen de una señal de audio

Para la variación de volumen se modifica la amplitud de la señal, esto mediante la multiplicación de la señal por una constante. Si multiplicamos por una constante mayor a uno esta aumentará su volumen a dicha proporción y si se multiplica por una constante menor a uno esta disminuirá su volumen.

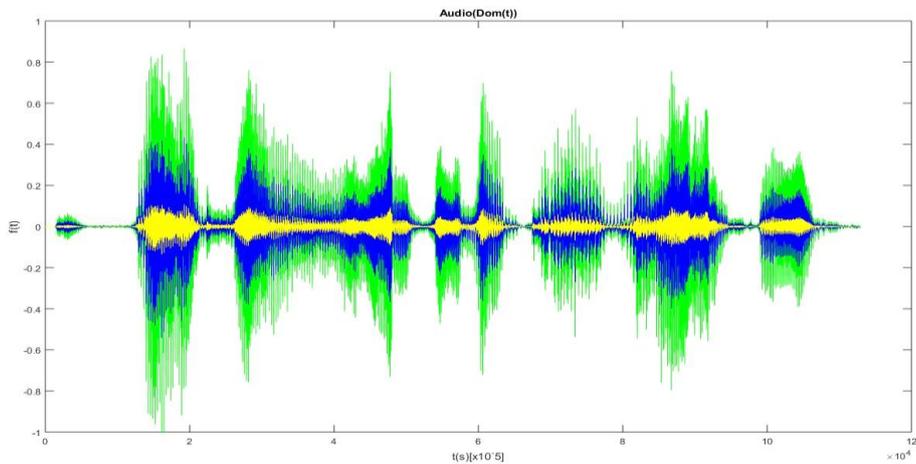


Fig. 4. Modificación de una señal de audio.

Aumento de volumen:

$$\text{Amplitud de la señal} \cdot X \geq \text{Amplitud de la señal}, \quad (32)$$

donde es un entero ≥ 1 .

Disminución de volumen:

$$\text{Amplitud de la señal} \cdot X < \text{Amplitud de la señal}, \quad (33)$$

donde es un entero < 1 .

En la Fig. 4, podemos ver que la gráfica azul es la señal original, la gráfica verde es la original multiplicada por una constante positiva (aumento de volumen) y la gráfica amarilla es la original multiplicada por una fracción (disminución de volumen).

3.4. Implementación de una colección de datos a la red neuronal

Hasta aquí hemos visto uno de los requisitos para voice cloning que es el tratamiento de las señales, ahora se implementó la propuesta algorítmica por Redes Neuronales que modificara sistemáticamente los datos hasta lograr su fin, llegar a una nueva colección de datos deseada, ver Fig. 5. En las siguientes pruebas se usará Keras de Tensorflow para probar la propuesta de construcción de la RN para palabras simples.

3.5. Pruebas del modelo

Para la implementación del Sistema se realizó el entrenamiento con 5 palabras simples de la Tabla 2. Esto bajo la premisa de volver experta a la red en ciertas formas de conocimiento de sonidos similares para que a la hora de someterlo a pruebas pueda ser capaz de formar predicciones bajo conocimientos adquiridos, intentando llenar así lagunas de no conocimiento, dado que la red neuronal funciona bajo memoria de

Tabla 2. Tabla de datos para el entrenamiento de la red neuronal.

Audio Hombre	Audio Mujer
Palabra 1: casa	Palabra 1: casa
Palabra 2: bueno	Palabra 2: bueno
Palabra 3: rama	Palabra 3: rama
Palabra 4: coma	Palabra 4: coma
Palabra 5: codera	Palabra 5: codera
Palabra 6: hola	Palabra 6: hola
Palabra 7: correr	Palabra 7: correr

conocimientos, ver Fig. 6, donde intentamos predecir un audio no conocido (circulo verde) a partir del conocimiento de audios conocidos similares (circunferencias verdes).

Como sabemos para realizar cualquier uso de datos debemos de tener una preparación o consideraciones en los datos, en este caso las palabras se grabaron bajo las mismas condiciones de frecuencia de muestreo en la captura de la señal. Y posteriormente se normalizaron, es decir, se buscó el valor máximo de la señal y se dividido la señal entre este para que todos los audios tuviesen un rango de -1 a 1. De esta manera, la red neuronal no intentará compensar el volumen de las muestras (remediar el volumen de las señales), y solo se centrará en la sonoridad. Esto con el fin de mejorar la eficiencia de la red.

A continuación, se muestran los puntos para preparar las señales:

- Normalizar los audios, con el fin de que el factor de volumen no sea compensado por la red.
- Recortar la señal a la misma cantidad de datos para todos los audios (misma longitud de datos).
- Imprimir los datos en un array ordenado, preparado para Python.
- Implementación del trining con Keras.
- Pruebas de la red entrenada resultante en Matlab.

3.6. Implementación del sistema con keras de tensorflow

En Python usando el entorno de Colab se implementó la red neuronal de la Fig 7 con la ayuda de la API de Keras de Tensorflow, el uso de esta herramienta se debio a la flexibilidad de uso y acoplamiento a grandes cantidades de datos a los que necesitamos procesar. A continuación, se explican los pasos seguidos en el código.

1 Importamos Tensorflow y la librería de NumPy.

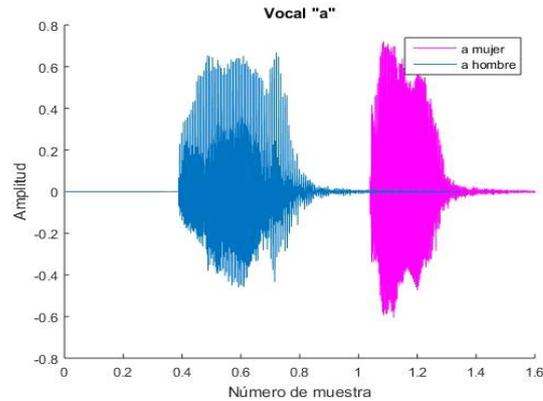


Fig. 5. Señales de audio de la vocal “a” en voces de una mujer y un hombre.



Fig. 6. Convergencia a la solución deseada no conocida por la red.

- 2 Declaramos los datos usados en la red neuronal, los datos de la Tabla 2.
- 3 Declarara en un solo array los datos de entrada y en un segundo array los datos de salida de la red neuronal, con los cuales se somete a entrenamiento la red.
- 4 Declarar el número de entradas de la capa de entrada de la red, en este caso es el tamaño de cada muestra de audio, construcción de la Fig. 2, 1900 neuronas en la capa de entrada y salida.

3.7. Resultados

Se entrenó la Red Neuronal con palabras simples. Con una voz de hombre e intentando llegar a una voz de mujer, de igual forma otra red se entrenó con voz de mujer para obtener la voz de un hombre respectivamente. Ver Fig. 7. La longitud de todas las palabras ingresadas fue de 19 mil datos en los arrays. Con lo cual el resultado de la red es de igual forma de 19 mil. Por lo que las pruebas se realizaron con palabras que puedan entrar dentro de este rango de longitud.

En la Fig. 8 y 9 podemos ver los datos de entrenamiento a los cuales fue sometido el sistema en Tensorflow.

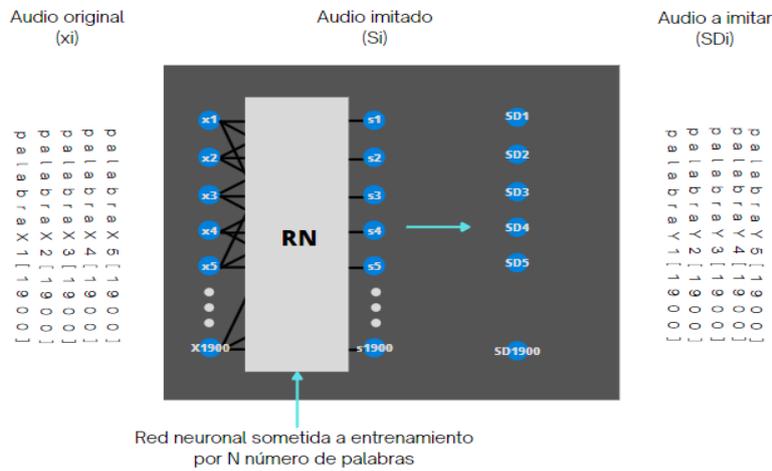


Fig. 7. Entrenamiento de la Red Neuronal propuesta.

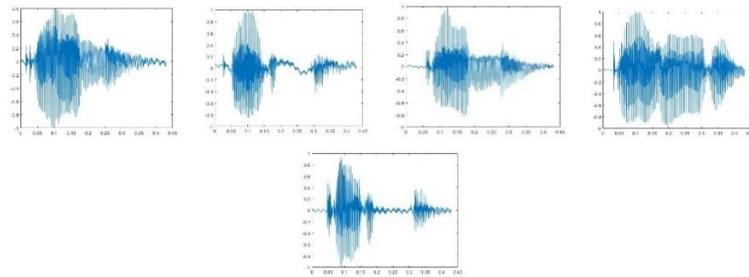


Fig. 8. Audios implementados en el entrenar el sistema: Tipos de Audios Deseados, Tono de voz de una Mujer.

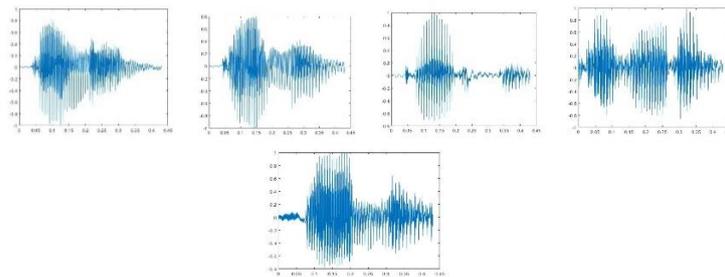


Fig. 9. Audios implementados en el entrenar el sistema: Audios de Entrada, Tono de voz de un Hombre.

En la Fig. 10. Podemos ver la estabilización de la función de pérdida de la red neuronal al intentar llegar a los resultados deseados, en función del número de épocas o iteraciones que tuvo que realizar la red para lograr minimizar errores, logrando así el

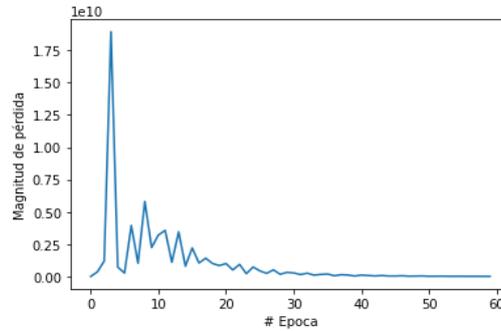


Fig. 10. Estabilización de la Red Neuronal para llegar al resultado deseado.

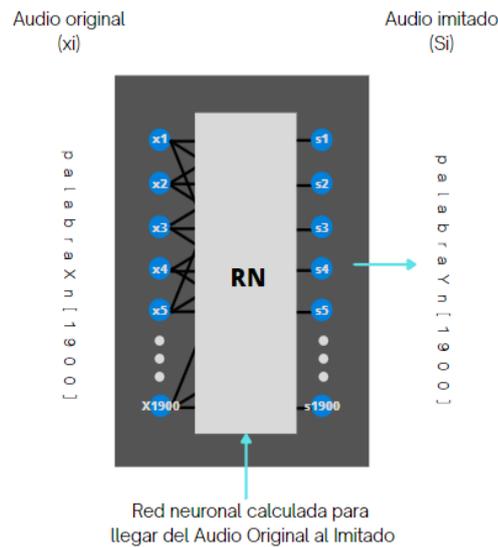


Fig. 11. Prueba de la Red Neuronal propuesta.

propósito de los resultados. Se implementaron las palabras conocidas por el sistema, es decir, se ingresaron todas las palabras con las que se entrenó el sistema en la etapa anterior, ver Fig. 11.

Esta etapa dio resultados correctos para todos los datos, es decir, se ingresó la palabra “coma” en la red neuronal con la voz de un hombre y sonó como resultado la palabra “coma” con la voz de una mujer, es importante mencionar que el resultado arrojó los sonidos deseados, pero con ruido en ellas, ver Fig. 12.

Donde se escuchaba la palabra deseada junto con ruido. Existen técnicas de eliminación de ruido, pero esta implementación solo buscaba el estudio de la red neuronal propuesta de la Fig. 2.

Para la prueba final del sistema de igual forma se implementó la Fig. 11, pero para palabras no conocidas por el sistema, palabras no ingresadas en la etapa de

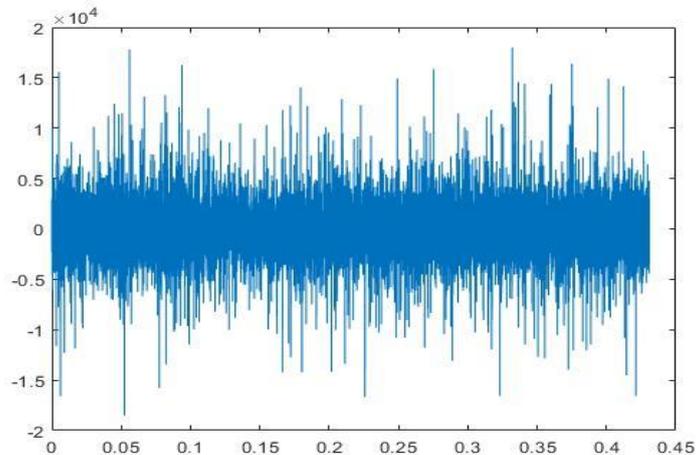


Fig. 12. Palabra imitada por la RN con ruido generado por la misma RN.

entrenamiento del Modelo, en base a la Fig. 6. Ingresando datos de voz de un hombre y dando como resultado un timbre de voz de mujer.

En esta etapa de prueba los resultados fueron favorables ya que todos se recreaba la voz de una mujer, pero con ciertos problemas de vocalización al pronunciar las palabras, dado lo complejidad del problema, ya que son sistemas que requieren de más información dado el funcionamiento de las redes neuronales. [6]

4. Conclusiones y trabajo futuro

La construcción de red neuronal propuesta logro resultados favorables al replicar los datos deseados de ciertos datos origen, el sistema se puso a prueba con múltiples valores, como el ejemplo de la Tabla 1, dando resultados siempre favorables. Dado esto se buscó la implementación de datos más complejos para simular a pequeña escala los sistemas de clonación de voz, donde la red neuronal termino construyendo un sistema para llegar de la voz de un hombre a la voz de una mujer, intentando en todo momento llegar a la voz a clonar.

Con ello podemos decir que la RN propuesta es capaz de replicar datos de N dimensiones, probando el sistema con pocos datos y hasta miles de datos, esto último llevándolo a temas de clonación de voz. Por lo cual, la red logró recrear sonidos simples artificialmente. Aunque no sea un sistema escalable por sus altos requerimientos y baja flexibilidad para acoplarse a palabras de distintos tamaños queda analizar cuestiones como aumentar el entrenamiento de la RN con cantidades masivas de datos para que pueda ser capaz de obtener más información de la identidad de los datos a los que la RN replica en el entrenamiento.

Referencias

1. Uday-Kamath, J. L., James, W.: Deep learning for NLP and Speech. USA: Springer (2019)
2. Navin, M.: Deep Learning with Applications Using Python: Chatbots and Face, Object, and Speech Recognition with TensorFlow and Keras. USA. Apress (2018)
3. Rudolph, R.: Redes Neuronales: Guia Sencilla de Redes Neuronales Artificiales. USA. CreateSpace Independent Publishing Platform (2018)
4. Ye, R., Weiss, F., Wolfgang, M., Melvin, J., Zhifeng, C., Yonghui, W.: Direct speech-to-speech translation with a sequence-to-sequence model.
5. Sercan, Ö., Arık, C., Kainan, P.: Neural Voice Cloning with a Few Samples
6. Anel, R.A.: Modelo de entrenamiento y testing de la RN. *Neuralnetworkstest* (2022)

Clasificación bi-clase de canciones infantiles aplicando inteligencia artificial y procesamiento de lenguaje natural

David Soto Osorio¹, Jesús Jaime Moreno Escobar¹,
Liliana Chanona-Hernández¹, Grigori Sidorov²,
César Jesús Núñez-Prado¹

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

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

davidsotoesime@outlook.com, jmorenoe@ipn.mx
{lchanona, cesar.jnprado}@gmail.com, sidorov@cic.ipn.mx

Resumen. Desde hace muchos años, las canciones infantiles han formado parte del crecimiento de los seres humanos, ya que ofrecen grandes beneficios tales como; el desarrollo de la inteligencia, la enseñanza de nuevos valores y buenos hábitos; pero es un hecho, que del mismo modo en que las canciones dirigidas a los niños pueden tener una influencia positiva, también lo pueden hacer de manera negativa. Por esta causa, en el presente trabajo se propone un algoritmo de clasificación bi-clase que emplea técnicas de procesamiento de lenguaje natural y el modelo de inteligencia artificial K vecinos más cercanos, para clasificar canciones infantiles en positivas y negativas.

Palabras clave: Canciones infantiles, inteligencia artificial, procesamiento de lenguaje natural, clasificación bi-clase.

Bi-Class Analysis and Classification of Songs for Children Applying Natural Language Processing and Artificial Intelligence

Abstract. For many years, children's songs have been part of the growth of human beings, since they offer great benefits such as; the development of intelligence, the teaching of new values and the learning of good habits; but it is

a fact, that in the same way that songs aimed at children can influence them in a positive way, they can also influence them in a negative way. For this reason, in the present work a bi-class classification algorithm is proposed that uses natural language processing techniques and the K nearest neighbors' artificial intelligence model to classify children's songs into positive and negative.

Keywords: Children's songs, artificial intelligence, natural language processing and bi-class classification.

1. Introducción

La música es considerada como un elemento cultural que fortalece el aprendizaje y la memoria de los seres humanos; inclusive cuando se trate de géneros diferentes. A partir del siglo XX, el género de canciones infantiles comenzó a ser considerado como una parte muy importante en el crecimiento del ser humano, ya que cuenta con grandes beneficios como son; el desarrollo de la inteligencia, la enseñanza de nuevos valores y permite que los niños aprendan buenos hábitos para su futuro.

En el siglo XXI el desarrollo tecnológico ha permitido que las personas puedan estar conectadas entre sí a través de dispositivos electrónicos, lo cual facilita la búsqueda de información y la interacción entre los seres humanos, sin embargo; las nuevas generaciones pueden aprender cosas tanto positivas como negativas y esto los convierte en una generación vulnerable, que puede ser influenciada por información errónea o por géneros musicales que no son aptos para su edad o inclusive pueden tener una mala influencia de aquellas canciones que son consideradas infantiles pero que transmiten un mensaje que puede perjudicar en su comportamiento o vocabulario.

Por lo anteriormente descrito, en este trabajo se propone desarrollar un corpus lingüístico digital con canciones dirigidas hacia niños y aplicar un algoritmo de aprendizaje supervisado basado en técnicas de procesamiento de lenguaje natural y el modelo k vecinos más cercanos para realizar una clasificación bi-clase de letras de canciones infantiles.

2. Trabajos relacionados

Los trabajos seleccionados en nuestro estado del arte realizan un análisis de letras de canciones de diferentes géneros musicales por medio de técnicas de procesamiento de lenguaje natural y algoritmos de inteligencia artificial.

En [1] se menciona la necesidad de la industria musical para mejorar la experiencia que tienen sus usuarios, es por ello que aplican un algoritmo de clasificación probabilístico clásico conocido como *Naive Bayes*, este algoritmo se implementó utilizando el módulo *AI de Perl*. Utilizan una clasificación de 5 diferentes clases, el etiquetado se realizó manualmente y se aplicó un método de validación cruzada para separar el corpus lingüístico digital en conjuntos de entrenamiento y prueba, y obtuvieron un resultado del 82 % de precisión.

Por otra parte, en [2] aplican un algoritmo de aprendizaje supervisado con el que se efectúa la clasificación de sentimientos en las canciones hindi en 5 clases; para la generación del corpus lingüístico usaron diferentes fuentes de internet, se hizo una asignación de clases de forma manual, para este caso; solicitaron el apoyo de dos estudiantes a los que se les asignó la tarea de leer la canción y clasificarla. Una vez que se contó con el corpus lingüístico etiquetado se aplicó la técnica de preprocesamiento de eliminación de *stop-words*. Se realizó la separación en conjunto de entrenamiento y prueba aplicando un método de validación cruzada de 10 iteraciones.

Para la clasificación de las letras se usó el módulo *LibSVM* de la herramienta *WEKA*¹, al probar el algoritmo se logró obtener un porcentaje de precisión del 38.49 %, y concluyeron que este porcentaje se debe a que no consideraron varias características textuales es por ello que como propuesta a futuro buscan aumentar las características textuales para incrementar el porcentaje de precisión. En [3] aplican una clasificación bi-clase de canciones de diferentes géneros musicales.

Para la generación del corpus lingüístico utilizaron la lista de *Jamrock Entertainment* de las 100 mejores canciones por año. Se generó un corpus de 420 letras, de las cuales 210 letras son positivas y 210 letras son negativas, para realizar la clasificación se usó el algoritmo *Present In One*. En este algoritmo realizan dos conteos, el primero almacena el número de palabras que son consideradas como positivas mientras que el segundo almacena el número de palabras que son clasificadas como negativas, al final dependiendo del conteo mayor, se asigna la clase a la letra, como resultado de esta metodología obtuvieron un porcentaje de precisión del 66%.

En [4] se realiza un análisis de un corpus lingüístico conformado por letras de canciones de artistas que se han suicidado, este trabajo tiene como propósito identificar tendencias suicidas por medio de las letras escritas por algunos autores o artistas. Se generó un corpus lingüístico de 533 canciones de las cuales 253 letras fueron escritas por 4 artistas que no han tenido tendencias suicidas y 280 escritas por 5 artistas que cometieron suicidio.

Para realizar la clasificación se realizó la separación del corpus en conjunto de prueba y conjunto de entrenamiento, el conjunto de prueba estuvo conformado por 63 letras de artistas que se suicidaron y 46 letras de artistas que no han presentado tendencias suicidas. Sobre el corpus creado se aplicaron técnicas de preprocesamiento, tales como; tokenización y lematización usando la librería *OpenNLP*².

Para el algoritmo de clasificación se usó la librería *WEKA* y se ejecutaron diferentes algoritmos de clasificación y se determinó que el algoritmo *SimpleCart* obtiene el mayor porcentaje de precisión con un 70.6%, concluyen que el procesamiento de lenguaje natural puede ser una herramienta muy poderosa para realizar diferentes análisis dentro de las letras.

Por último, en el trabajo [5] plantean el hecho de que la música forma parte de la vida diaria de todos los seres humanos y el impacto que ésta puede producir en ello y es por ello que realizan un análisis de letras de canciones en idioma tailandés. Para la clasificación consideran sólo dos clases (feliz y triste), para generar el corpus

¹ <https://www.cs.waikato.ac.nz/ml/weka/>

² <https://opennlp.apache.org/>

lingüístico consultaron la página de *Chord Café*³ y obtuvieron un corpus lingüístico de 427 canciones felices y 317 canciones tristes. Para el preprocesamiento se usó la segmentación de las letras y para el clasificador se consideró una red neuronal multicapa, y con ello obtuvieron un 68 % de precisión.

3. Desarrollo del proyecto

Para el desarrollo del algoritmo de clasificación bi-clase se consideran un conjunto de etapas que tienen como propósito pre-procesar las letras de canciones infantiles para su posterior clasificación en clase positiva y negativa. A continuación, se explican las etapas que se llevaron a cabo para la clasificación.

3.1. Corpus lingüístico digital

Un corpus lingüístico es un conjunto de textos que tienen un mismo origen y que tiene por objetivo almacenar un conjunto de documentos o textos con el fin de usar estos datos para realizar un análisis. Para el desarrollo del corpus lingüístico digital del presente trabajo, se consultaron diferentes páginas web en las que se presentan letras de canciones infantiles de diferentes artistas, películas y programas de televisión. El corpus lingüístico integra canciones en español de la década de los cincuenta hasta la actualidad. Todas las letras descargadas fueron almacenadas en un solo directorio en texto plano con una numeración consecutiva. Como resultado de esta metodología de búsqueda se generó un corpus lingüístico de 220 letras⁴.

3.2. Etiquetado del corpus lingüístico

Se decidió realizar el etiquetado del corpus lingüístico digital sólo con dos posibles clases: positivo y negativo. Para realizar esta tarea, se solicitó el apoyo de 5 profesionales en el campo de la psicología infantil de las cuales son todas mujeres con un rango de edad de 28 a 40 años y una experiencia laboral de 5 a 10 años. La tarea asignada a las psicólogas fue determinar si la canción transmite un mensaje positivo o negativo a los niños con base en su propia experiencia profesional y que justificara la clasificación redactando un pequeño párrafo.

Para realizar la asignación de la clase final de cada una de las letras, se consideró la siguiente métrica: Si por lo menos 3 especialistas clasificaron como positiva la misma canción, a ésta se le asignaría la clase positiva, en caso contrario se clasificaría como negativa. Se consideraron como letras negativas aquellas que transmiten un mensaje con lenguaje inapropiado o que difunde el miedo hacia algunos personajes o también incentiva el machismo, la desigualdad y el racismo. Al terminar la asignación de clases se tiene que de las 220 letras que forman el corpus lingüístico digital a 172 canciones se les asignó la clase positiva y 48 fueron consideradas negativas.

³ <https://en.chordcafe.com/>

⁴ Proximamente estará disponible en: <https://www.cic.ipn.mx/~sidorov/>



Fig. 1 Tokenización.



Fig. 2 Eliminación de Stop-words.

3.3. Preprocesamiento del corpus lingüístico

En esta etapa, se aplicaron técnicas de preprocesamiento sobre el corpus lingüístico con el propósito de reducir el tiempo de procesamiento y aumentar el porcentaje de precisión del algoritmo de clasificación. Para aplicar estas técnicas se utilizó el lenguaje de programación *Python* junto con las librerías *NLTK*, *Pandas* y *Spacy*.

De manera general, la tokenización consiste en separar las oraciones en palabras, a estas palabras se les conoce como tokens. En la Figura 1 se puede observar la frase “La niña está en su cama de madera”, al aplicar la tokenización se tiene el siguiente resultado.

Una vez que se realizó la tokenización se procede a realizar la eliminación de *stop-words*. Se consideran como *stop-words* a todas aquellas palabras que aparecen con una frecuencia muy alta en los textos.

La eliminación de *stop-word* consiste en eliminar todas aquellas palabras que no se consideran para el análisis del texto, como los conectores, artículos, pronombres, preposiciones o conjunciones; para aplicar la eliminación de *stop-words* se utilizaron los diccionarios de las librerías *NLTK* (*Natural Language Toolkit*) y *Spacy*.

En la Figura 2 se puede observar la frase “La niña está en su cama de madera”, los recuadros verdes indican que pertenecen al conjunto de *stop-words* y los recuadros amarillos indican que nos quedaremos con estos tokens.

Por último, se aplicó la lematización, la cual consiste en llevar a todas las palabras a su forma base, es decir; a su forma infinitiva. A las palabras obtenidas del proceso de lematización se les conoce como lemas. Para realizar este proceso se usó un diccionario de la librería *Spacy* en español. En la Figura 3, la primera columna corresponde a verbos conjugados y la segunda columna a los verbos lematizados.

3.4. Asignación de números de identificación y conteo de palabras

En esta etapa se realizó la asignación de números de identificación y conteo de palabras en las letras, con la función *CountVectorizer()* de la librería *Scikit-learn*. Estos identificadores fueron utilizados posteriormente para hacer un conteo del número de

Palabras	Lemas
Corriendo	Correr
Comiendo	Comer
Saltaba	Saltar

Fig. 3 Lematización

Tabla 1. Porcentaje de precisión.

No. de vecinos	Porcentaje de precisión
1	78%
3	80%
5	84%
7	81%
9	83%

veces que aparece una palabra en una canción y de esta manera se generó una bolsa de palabras.

3.5. Aplicación del método de validación simple

Un método de validación simple consiste en dividir de manera aleatoria el conjunto de datos en dos conjuntos, una se usa para el entrenamiento del algoritmo y el otro para realizar las pruebas. Para llevar a cabo la separación del corpus lingüístico, se utilizó la función “*train_test_split()*” de la librería *Scikit-learn*.

Para el presente proyecto, se aplicó la métrica 70-30, es decir; 70 % del corpus lingüístico se utilizó para el entrenamiento del algoritmo K vecinos más cercanos y el 30 % del corpus lingüístico, para realizar las pruebas de funcionamiento y para determinar el porcentaje de precisión.

3.6. K vecinos más cercanos

El algoritmo de inteligencia artificial k vecinos más cercanos utiliza la distancia euclidiana para determinar qué datos están más cercanos al dato a clasificar. Dependiendo del conteo mayor de datos, se asignará la clase al nuevo dato. Para aplicar el algoritmo de clasificación KNN se utilizó la librería *Scikit-Learn*.

En esta última etapa se ingresó el número de vecinos a considerar para la clasificación, de acuerdo con el estado del arte, al tratarse de una clasificación bi-clase se recomienda que el número de vecinos para la clasificación sea impar para evitar empates.

Posteriormente se realizó el entrenamiento del algoritmo k vecinos más cercanos con el 70 % del corpus lingüístico previamente procesado y etiquetado. Una vez que se entrenó al algoritmo de k vecinos más cercanos se procedió a verificar el funcionamiento del algoritmo con el conjunto de prueba, como resultado se obtuvo el porcentaje de precisión del algoritmo de clasificación y las letras clasificadas.

4. Resultados

Al tratarse de un algoritmo que considera los vecinos más cercanos al dato a clasificar se consideraron 1, 3, 5, 7 y 9 vecinos más cercanos, los porcentajes de precisión que se obtuvieron con cada métrica se presentan en la Tabla 1.

Como se observa en la Tabla 1 al variar el número de vecinos para la clasificación de la letra, también varía el porcentaje de precisión. Observando las pruebas se puede identificar que se obtiene un mayor porcentaje de precisión al considerar solamente a los 5 vecinos más cercanos al dato a clasificar.

5. Conclusiones y trabajo a futuro

Al variar el número de vecinos para la clasificación de la letra varía el porcentaje de precisión por lo que en este algoritmo se consideran los 5 vecinos más cercanos. Para lograr este porcentaje de precisión se realiza un conjunto de técnicas de preprocesamiento que permiten eliminar palabras no necesarias para el análisis del texto, así como permiten reducir el tiempo de procesamiento del corpus, de esta forma se pueden ahorrar recursos computacionales.

El algoritmo propuesto para la clasificación de canciones infantiles solamente funciona con letras en español debido a que los diccionarios que se usan para la eliminación de stop-word y lematización se encuentran en español, por lo que una palabra en inglés será desconocida para los procesos de eliminación de stop-words y lematización. También este algoritmo es muy vulnerable a los errores ortográficos, ya que si una palabra está mal escrita será completamente diferente para el algoritmo, estos errores ortográficos pueden afectar el porcentaje de precisión del algoritmo de clasificación.

Debido a los problemas antes mencionados para trabajos futuros se propone complementar las técnicas de preprocesamiento con diccionarios en otros idiomas y por último se propone que las letras que son usadas para el desarrollo del corpus serán verificadas anteriormente de forma manual para corregir las faltas ortográficas.

Referencias

1. Mahedero, J. P. G., Martínez, A., Cano, P.: Natural language processing of lyrics. In: Proceedings of the 13th annual ACM International Conference on Multimedia, pp. 475–478 (2005) doi: 10.1145/1101149.1101255.

2. Patra, B., Das, D., Bandyopadhyay, S.: Mood Classification of Hindi Songs based on Lyrics. In: Conference on Natural Language Processing, pp. 261–267 (2015)
3. Ashley, M., Sarah, E.: Identifying the Emotional Polarity of Songs Lyrics through Natural Language Processing. (2010)
4. Mulholland, M., Quinn, J.: Suicidal Tendencies: The Automatic Classification of Suicidal and Non-Suicidal Lyrics Using NLP. International Joint Conference on Natural Language Processing, pp. 680–684 (2013)
5. Srinilta, C., Sunhem, W., Tungjitnob, S., Thasanthiah, S., Vatathanavaro, S.: Lyric-based Sentiment Polarity Classification of Thai Songs. In: Proceedings of the International MultiConference of Engineers and Computer Scientists vol. 1 (2017)
6. Bird, S., Klein, E., Loper, E.: Natural Language Processing with Python, Analyzing Text with the Natural Language Toolkit. (2009)
7. Hervás, M., Barrio, G.: Influencia de las actividades audio musicales en la adquisición de la lectoescritura en niños y niñas de cinco años. (2017)
8. Orantes, A.: La influencia de la música en nuestros niños. (2001)
9. Martínez, B.: Canciones infantiles de siempre con un nefasto mensaje para los niños. (2020)
10. Pandas. [Online] Available: <https://pandas.pydata.org/>
11. Numpy [Online] Available: <https://numpy.org/>
12. ScikitLearn [Online] Available: <https://scikit-learn.org/>

Asistente virtual de respuestas referentes a las becas ofertadas en ESIME Zacatenco

Sahian D. Garcia Camacho¹, Brian J. Figueroa Acevedo¹, Liliana Chanona-Hernández¹, César Jesús Núñez-Prado¹, A. J. Salazar Cisneros¹, Grigori Sidorov²

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

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

{sasmith.cs, brianjohanfigac, lchanona, cesar.jnprado}@gmail.com
alvaro-javier12@hotmail.com, sidorov@cic.ipn.mx

Resumen. Es posible observar el progreso en todas las ciudades del mundo desde las propias aulas de estudio. La automatización de los procesos impacta de manera favorable, en la reducción del trabajo de las personas que se encargan de las actividades burocráticas dentro de las instituciones educativas. En este proyecto, se busca automatizar las respuestas a las dudas generadas por parte de la comunidad estudiantil de la Escuela Superior de Ingeniería Mecánica y Eléctrica (ESIME) del Instituto Politécnico Nacional (IPN) a través de un sistema que funciona como filtro a través de procesar la información referente a las becas ofertadas, aplicando 4 diferentes algoritmos de inteligencia artificial y utilizando los servicios de mensajería de *WhatsApp* y *Twitter* para ofrecer la respuesta a la pregunta solicitada.

Palabras clave: Automatización, sistema de respuestas, inteligencia artificial, servicios de mensajería.

Virtual Assistant of Responses Related to Scholarships Offered in ESIME Zacatenco

Abstract. It is possible to observe the progress in all the cities of the world from classrooms of study. The automation of processes impacts favorably, in the reduction of the work of the people who are in charge of bureaucratic activities within educational institutions. In this project, the aim is to automate the answers to the doubts generated by the student community of the Higher School of

Mechanical and Electrical Engineering (ESIME) of the National Polytechnic Institute (IPN) through a system that works as a filter through process the information regarding the scholarships offered, applying 4 different artificial intelligence algorithms and using the WhatsApp and Twitter messaging services to offer the answer to the requested question.

Keywords: Automation, response system, artificial intelligence, messaging services.

1. Introducción

La demanda escolar en las universidades públicas de México siempre supera a la cantidad de lugares disponibles, lo cual implica; que cada inicio de semestre, estas universidades cuentan con el 100% de sus aulas ocupadas (para primeros semestres). Al ser universidades públicas, siempre se pueden encontrar becas ofertadas para los alumnos que cumplen con los requisitos en tiempo y forma. La información referente a este tipo de apoyo, generalmente es publicada en carteles impresos en puntos específicos dentro de las mismas instituciones y también a través de redes sociales en las cuentas oficiales de cada institución.

Para el caso de la Escuela Superior de Ingeniería Mecánica y Eléctrica del Instituto Politécnico Nacional, unidad Zacatenco, en el área de control escolar se encuentra un departamento que se encarga de recibir la documentación de los estudiantes para postularse a cualquiera de las becas ofertadas dentro del plantel. Este departamento, se ocupa también de dar solución a cualquier tipo de duda generada dentro de la comunidad estudiantil relacionada con las postulaciones de las becas en dos turnos (matutino y vespertino).

Cabe mencionar, que la ESIME Zacatenco es una de las escuelas del IPN que cuenta con una matrícula considerable, es decir; tiene aproximadamente 11 mil estudiantes inscritos cada ciclo escolar en ambos turnos. Al tener una de las comunidades estudiantiles más grandes, el trabajo administrativo desarrollado por el área de control escolar es muy desgastante para los mismos trabajadores de esta área y por otra parte, el tiempo que debe pasar el alumno en las filas para poder ser atendido, es en promedio de una hora¹.

Por otra parte, en universidades importantes de otros países, se ha demostrado con la práctica, que el automatizar ciertos procesos administrativos reduce considerablemente el estrés de los trabajadores y de los mismos estudiantes. Es por ello, que en este proyecto se busca crear un sistema que funcione como filtro entre el departamento de becas de ESIME Zacatenco y su comunidad estudiantil.

Dicho sistema debe ser capaz de responder las preguntas expresadas por los alumnos a través de los servicios de mensajería de *WhatsApp* y mensajes directos de *Twitter*, realizando el procesamiento de lenguaje natural de las convocatorias de las becas ofertadas y ofreciendo una respuesta al alumno con dudas, el sistema estará disponible

¹ Dato extraído de encuestas realizadas a los alumnos

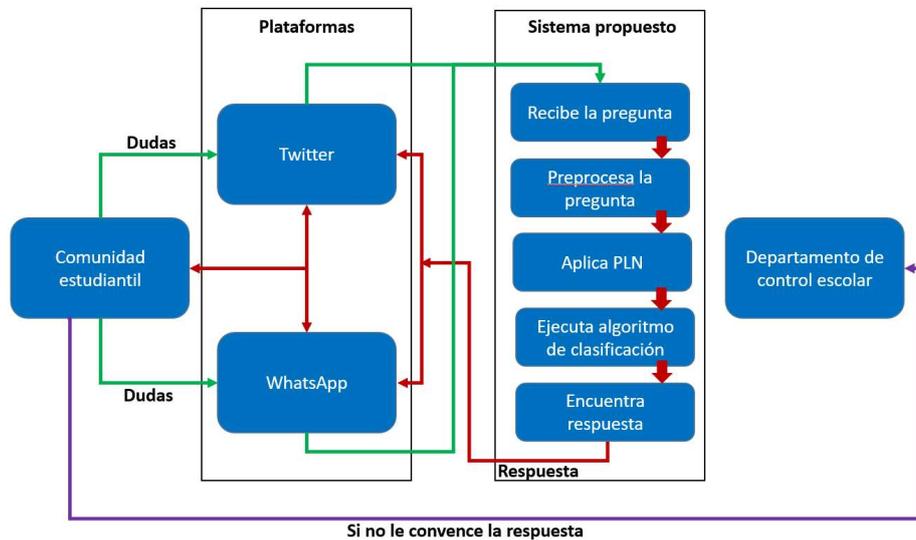


Fig. 1. Diagrama general del sistema desarrollado.

las 24 horas del día, los 7 días de la semana y con ello, también se busca reducir el número de dudas que llegan directamente al departamento de becas.

2. Trabajos relacionados

Para el desarrollo del presente proyecto, se consultó el estado del arte actualizado referente al desarrollo de aplicaciones para instituciones educativas, implementación de *chatbots*, aplicación de técnicas de procesamiento de lenguaje natural e inteligencia artificial. A continuación, se mencionan algunos de los trabajos más relevantes. Con la llegada de la presente era digital a nuestras vidas, el internet y las aplicaciones de inteligencia artificial han marcado un crecimiento de uso año tras año.

Aún más, con la aparición de la pandemia por COVID, las ventas del mercado electrónico a través de aplicaciones como Mercado Libre o Amazon se incrementaron exponencialmente. Detrás de estas aplicaciones, es posible encontrar algoritmos sofisticados de inteligencia artificial que son capaces de recomendar ciertos productos de acuerdo al perfil del mismo usuario y de sus búsquedas anteriores.

Relacionado al tópico de inteligencia artificial, podemos encontrar que marcó un parte aguas el trabajo [1] en donde se presentó a ELIZA, un sistema de conversación que era capaz de analizar la información que recibía y construir una respuesta a través de la descomposición de reglas. Estaba habilitado para identificar palabras claves dentro de la conversación, descubrir el contexto mínimo, elegir las transformaciones apropiadas y generar una respuesta.

Fue puesto a prueba con discusiones psicológicas y los resultados fueron tan prometedores que hoy en día, el desarrollo de aplicaciones que sean capaces de interactuar con el ser humano en lenguaje natural, es uno de los campos de estudio más recurrentes.

Las aplicaciones del procesamiento de lenguaje natural, no sólo están enfocadas al análisis de textos, sino que se busca implementarlas en la vida cotidiana del ser humano para resolver tareas metódicas, como por ejemplo, en [2] a través del internet de las cosas (*IoT* por sus siglas en inglés, *Internet of Things*) buscan automatizar el control de los apagadores de las luces, televisiones inteligentes y otros dispositivos electrónicos a través de realizar el procesamiento de lenguaje natural de los comandos proporcionados por el usuario en un *chatbot*.

Referente a las aplicaciones desarrolladas con fines académicos, encontramos a [3] el cual fue desarrollado en la Universidad de Guayaquil con el fin de ayudar en la comunicación entre los estudiantes y las áreas académicas de la universidad. Desarrollaron un sistema que provee información sobre las carreras ofertadas, horarios de clases, ubicación de aulas, entre algunos temas.

Fue implementado por etapas en donde se definen las variantes para entregar una respuesta y se detallan las tareas que se deben realizar. Tiene una integración con los servicios cognitivos de *Microsoft*, en donde se hace uso de las herramientas de compilación, implementación y administración que éste otorga para facilitar la construcción, depuración y publicación. Finalmente se implementó en *Facebook* a través de la *API* usando *Facebook Messenger* como canal de comunicación.

Finalmente, en [4] presentan un sistema automatizado que ofrece respuestas a consultas de los usuarios de un sistema educativo, utilizando una base de datos local y una base de datos web. Emplean algoritmos de aprendizaje automático, técnicas de procesamiento de lenguaje natural, concordancia de patrones y algoritmos de procesamiento de datos para mejorar el rendimiento del sistema.

En este trabajo aplican diferentes algoritmos tales como: adaptadores de entrada; los cuales permiten recibir o recuperar información de una determinada fuente, es decir; convertir los datos para que el *chatbot* pueda entenderlos. Por otra parte, se encuentra el adaptador lógico, que determina de forma racional como el *chatbot* seleccionará respuestas a diversas declaraciones de entrada. Para el entrenamiento cargan diálogos de ejemplo de la base de datos.

3. Metodología

Para el desarrollo del asistente virtual presentado en ésta investigación, se utilizó una metodología en cascada, la cual consistió de 4 etapas principales. La primera etapa consistió en generar las cuentas de perfil y de desarrollador en *Facebook* y *Twitter*, con la finalidad de utilizar las *API*'s (Interfaz de programación de aplicaciones) de cada una de estas redes sociales. Dentro de la documentación del manejo de la *API* de cada una de las aplicaciones anteriormente mencionadas, se proveen los métodos y librerías necesarias para poder realizar la conexión con diferentes lenguajes de programación.

Además de aperturar la cuenta de desarrollador en *Facebook*, se generó una cuenta en *Amazon Web Services* para utilizar los servicios del servidor. En la siguiente etapa se recolectaron las convocatorias de las 5 becas existentes en ESIME Zacatenco disponibles para el periodo 2021-2, también se recopilaban las dudas por parte de los

Tabla 1. Exactitud con similitud coseno.

Similitud Coseno			
Método de validación	90 – 10	80 – 20	70 – 30
Exactitud	66.66%	56.60%	66.66%

Tabla 2. Exactitud con máquina de vectores de soporte.

SVM CLASSIFIER			
Método de validación	90 – 10	80 - 20	70 – 30
Exactitud	37.03%	50.94%	45.56%

Tabla 3. Exactitud con K vecinos más cercanos.

K NEIGHBORS CLASSIFIER			
Método de validación	90 – 10	80 – 20	70 – 30
Exactitud (con k = 5)	74.07%	60.37%	60.75%
Exactitud (con k = 7)	66.66%	62.26%	62.02%
Exactitud (con k = 9)	62.96%	66.03%	50.63%

alumnos, expresadas en la cuenta oficial de Becas ESIME Zacatenco² en la red *Facebook*. También se realizó una encuesta a 107 alumnos, la cual tenía como finalidad el identificar las preguntas más frecuentes por parte de los alumnos en los procesos de convocatorias de becas.

El contenido de las convocatorias, respuestas de las encuestas y publicación de dudas en *Facebook* se convirtió a texto plano y en la siguiente etapa se llevó a cabo el preprocesamiento de la información, el cual consistió en la conversión del texto a minúsculas, tokenización, eliminación de nombres, números, palabras vacías (*stop-words*), emojis y lematización.

Estos procesos fueron realizados a través de *Stanza*³ y *NLTK*⁴ (*Natural Language ToolKit*) para la generación del corpus lingüístico digital.

En la cuarta etapa se aplicaron 4 clasificadores del paradigma supervisado provistos por *Scikit-Learn*.⁵ Los clasificadores aplicados son: similitud coseno (*cosine similarity*), máquina de vectores de soporte (*support vector machine*), k-vecinos más cercanos (*k-nearest neighbors*) y redes neuronales (perceptrón multicapa).

Para conocer la efectividad de cada uno de los clasificadores, se dividió la información de manera aleatoria en porcentajes fijos (70 – 30, 80 – 20 y 90 – 10); el porcentaje mayor se destinó para el entrenamiento del clasificador y el resto de la información para prueba del clasificador.

² <https://es-la.facebook.com/becasesimezac/>

³ <https://stanfordnlp.github.io/stanza/>

⁴ <https://www.nltk.org/book/>

⁵ <https://scikit-learn.org/stable/>

Tabla 4. Exactitud con perceptrón multicapa.

MULTI-LAYER PERCEPTRON					
Funciones de activación					
Optimizadores	identity	logistic	tanh	relu	Método de validación
lbfgs	48%	49%	51%	49%	70-30
sgd	58%	61%	58%	51%	
adam	49%	43%	49%	47%	
lbfgs	49%	58%	49%	53%	80-20
sgd	58%	60%	58%	55%	
adam	60%	72%	55%	57%	
lbfgs	56%	56%	56%	59%	90-10
sgd	63%	70%	56%	67%	
adam	59%	52%	56%	52%	

4. Resultados

Para esta investigación, se realizaron dos pruebas. En la prueba 1, los clasificadores supervisados se evaluaron aplicando los métodos de validación 70 – 30, 80 – 20 y 90 - 10. Para la prueba 2, se escogió aplicar el algoritmo de similitud coseno sobre la cuenta de *Twitter* y al clasificador k vecinos más cercanos sobre *WhatsApp*.

4.1 Prueba 1

Los resultados obtenidos en la prueba 1 se muestran a continuación. El clasificador de similitud coseno obtuvo un empate del 66.66% entre el método de validación 90–10 y 70–30 y tuvo una disminución del 10% cuando se consideró el 80% de entrenamiento y 20% para prueba. El método de validación con el mejor resultado sobre el clasificador de máquina de vectores de soporte fue el 80–20 obteniendo el 50.94% de Exactitud.

Con el clasificador K vecinos más cercanos utilizando las mismas métricas de validación, se varió el número de vecinos con k =5, 7 y 9. Con k = 5 y k=7, el mejor resultado lo obtuvo 90–10 con un 74.07% y 66.66% respectivamente, mientras que para k=9, lo obtuvo 80–20 con una Exactitud del 66.03%.

Con el perceptrón multicapa se utilizaron los optimizadores: lbfgs, sgd y adam y las funciones de activación: identity, logistic, tanh y relu y los mismos métodos de validación. Para el 70 – 30, el mejor resultado fue obtenido por la función logistic y el optimizador sgd con un 61%.

Para el caso del 80–20, logistic y el optimizador adam obtuvieron el 72% y finalmente, el 90–10 con la combinación logistic y sgd obtuvieron el 70%. Estos resultados fueron obtenidos asignando 3,000 épocas de aprendizaje.

4.2 Prueba 2

Se realizaron 50 pruebas con alumnos de la ESIME Zacatenco en la red social *Twitter* aplicando el algoritmo de similitud coseno como modelo de predicción, los alumnos realizaron en promedio dos preguntas por prueba, sumando un total de 119 preguntas de las cuales 93 fueron acertadas y 26 fueron incorrectas, obteniendo una Exactitud del 78.15%.

Por otra parte, se realizaron 50 pruebas en la red social *WhatsApp* aplicando el algoritmo de K vecinos más cercanos como modelo de predicción, los alumnos hicieron en promedio tres preguntas por prueba, sumando un total de 160 preguntas de las cuales 123 fueron acertadas y 37 fueron incorrectas, obteniendo una Exactitud del 76.8%.

5. Conclusiones y trabajo a futuro

Se pudo identificar con éxito toda la información referente a los procesos de las convocatorias de las 5 becas ofertadas en la ESIME Zacatenco para el periodo 21-2, las cuales incluyen Institucional, Manutención, BEIFI, Bécalos y Excelencia. Se generó un corpus digital con las dudas expresadas en la cuenta oficial de becas de ESIME Zacatenco por parte de la comunidad estudiantil, que contiene 16 clases y 262 líneas de texto. Y se aplicó una encuesta de 10 preguntas a 116 alumnos de diferentes semestres para ampliar el corpus original. Se creó y se desarrolló un asistente virtual que identifica la respuesta a ciertas preguntas recurrentes que son solicitadas a través de un mensaje de texto dentro de las redes sociales *WhatsApp* y *Twitter*. Tal respuesta es generada mediante algoritmos de clasificación supervisados. Referente a los clasificadores utilizados, el mejor resultado lo ofreció k-vecinos más cercanos con $k = 5$ con una Exactitud del 74.03% y el segundo lugar lo obtuvo el algoritmo de similitud coseno con una Exactitud del 66.66%.

Analizando los resultados obtenidos en ambas redes sociales se concluye que el asistente en *WhatsApp* fue de un mayor agrado por parte de los alumnos involucrados obteniendo una aceptación del 63.3%, aunque no se podría considerar con una mala aceptación el asistente implementado en *Twitter* que obtuvo el 53.1% porque posiblemente se debe al tiempo de espera entre una respuesta y otra debido a que durante la ejecución se requiere de un retraso solicitado por la API de *Twitter* para la lectura de nuevos mensajes directos.

Los alumnos que realizaron las pruebas sobre ambas plataformas indicaron que les gustaría que el asistente se implementara de manera permanente debido a que superó sus expectativas. El trabajo actual podría ser escalable en alguna otra red social tal como *Facebook*, la cual es una de las aplicaciones con el mayor número de usuarios a nivel mundial y, por lo tanto, la información a la comunidad de la ESIME Zacatenco tendría un mayor alcance. Además, existen otras aplicaciones que se podrían utilizar para tener un alcance mayor con la comunidad como *Instagram*, *LinkedIn* o *Google+*.

Este sistema también podría ser implementado en una aplicación móvil por parte de la ESIME, con la finalidad de que el usuario (la comunidad estudiantil) tenga una mejor experiencia a la hora de resolver sus dudas y acudir por información. Se podrían

implementar diferentes clasificadores y métricas de validación con la intención de comparar cuales dan mejores resultados e implementarlos en la aplicación final.

Referencias

1. Weizenbaum, J.: ELIZA - a computer program for the study of natural language communication between man and machine. Association for Computing Machinery, vol. 9, no. 1, pp. 36–45 (1966). doi: 10.1145/365153.365168.
2. Baby, C. J., Khan, F. A. Swathi, J. N.: Home automation using IoT and a chatbot using natural language processing. *Innovations in Power and Advanced Computing Technologies*, pp. 1–6 (2017). doi: 10.1109/IPACT.2017.8245185.
3. Delgado-Guerrero, J. S., León-Bazan, Y. Y., Sánchez-Moreno F. J.: Desarrollo de chatbot usando bot framework de Microsoft. *Espirales Revista Multidisciplinaria de Investigación*, vol. 1, no. 11 (2017). doi: 10.31876/re.v1i11.133.
4. Gelbukh, A., Sidorov G: Procesamiento automático del español con enfoque en recursos léxicos grandes, pp. 301 (2010)
5. Bahit, E.: Curso: Python para principiantes (2012)
6. Belfin, R. V., Shobana, A. J., Manilal, M., Mathew, A. A., Babu, B.: A graph based chatbot for cancer patients. In: 5th International Conference on Advanced Computing & Communication Systems, pp. 717–721 (2019). doi: 10.1109/ICACCS.2019.8728499.
7. Bird, S., Klein, E., Loper, E.: *Natural Language Processing with Python* (2009)
8. Sidorov, G.: Construcción no lineal de n-gramas en la lingüística computacional. *Sociedad Mexicana de Inteligencia Artificial* (2013)
9. Guyon, Isabelle, Bennett, K., Cawley, G., Jair-Escalante, H., Escalera, S., Kam-Ho, T., Macià, N., Ray, B., Saeed, M., Statnikov, A., Viegas, E.: Design of the 2015 ChaLearn AutoML challenge. In: International Joint Conference on Neural Networks, pp. 1–8 (2015). doi: 10.1109/IJCNN.2015.7280767.
10. Kelleher, J. D., Namee, B. M., D'Arcy A.: *Fundamentals of machine learning for predictive data analytics. Algorithms, worked examples, and case studies* (2015)
11. Mosley, L.: A balanced approach to the multi-class imbalance problem (2013) doi: 10.31274/etd-180810-3375
12. Mauldin, M. L.: Chatterbots, tinymuds, and the turing test entering the Loebner prize competition. In: *Proceedings of the Twelfth AAAI National Conference on Artificial Intelligence*, pp.16–21 (1994)
13. Qi, P., Zhang, Y., Zhang, Y., Bolton, J., Manning, C. D.: Stanza: A python natural language processing toolkit for many human languages. In: *Association for Computational Linguistics (ACL) System Demonstrations* (2020)
14. Kohavi, R.: A study of cross-validation and bootstrap for accuracy estimation and model selection. In: *Proceedings of the 14th international joint conference on Artificial intelligence*, Morgan Kaufmann Publishers Inc, vol. 2, pp. 1137–1143 (1995)

Predicción de victoria en videojuego competitivo con Deep Learning

Juan Alberto González Gutiérrez, Cristian Ramón Hernández Ordoñez,
Marco Antonio Vidal Flores, Melina Alexia Zárate Zenteno

Universidad Autónoma del Estado de México,
Centro Universitario UAEM Nezahualcóyotl,
México

{alexzarate0007, cristian.her.ord,
rapsoryt, shsjuan}@gmail.com

Resumen. League of Legends es un videojuego de estrategia competitivo 5 vs 5, se puede observar un amplio margen de personajes (más de 150) disponibles para escoger; existe un interés importante en conocer el resultado de una partida antes de comenzarla, y aunado a ese interés una creencia popular, “Las partidas se pueden ganar desde la selección de personajes, ya que existen tipos de personajes que son mejores contra otros”. Abriendo la posibilidad de poder predecir el resultado de una partida con solo saber sus condiciones iniciales, basándose en los personajes. Aquellas personas que pudieran acceder a esta información antes de comenzar a jugar tendrían una ventaja crucial. Para corroborar esta hipótesis se recurrió al uso de Deep Learning con el modelo de red neuronal “Perceptrón Multi Capa”. Mediante una base de datos con un total de 62 mil partidas, las cuales fueron obtenidas directamente de la API oficial de la desarrolladora del videojuego RIOT GAMES, de esta forma se buscó como resultado que el algoritmo aprenda y finalmente realice una predicción acerca de qué equipo es vencedor en el estilo de juego clásico.

Palabras clave: API, Deep Learning, inteligencia artificial, red neuronal, videojuegos, predecir, perceptrón multicapa.

Prediction of Victory in a Competitive Video Game with Deep Learning

Abstract. A system developed in Python capable of predicting the victory of a competitive game, which is League of Legends, will be implemented. Through a large amount of data which will be obtained from the official API of the video game developer. The aim is to train a neural network, in this way the algorithm

will learn and finally make a prediction about which team is the winner in a specific style of play.

Keywords: Artificial intelligence, API, neural network, videogames, deep learning, predict, multilayer perceptron.

1. Introducción

1.1. Inteligencia artificial y League of Legends

La inteligencia artificial en pocas palabras podría definirse como “desarrollo de métodos que permitan comportarse a las computadoras de modo inteligente” [1], en otras palabras, es hacer computacional el conocimiento humano por procedimientos simbólicos o conexionistas. En términos coloquiales la IA se usa cuando una máquina es capaz de imitar las funciones cognitivas propias de la mente humana [2]. De hecho, los videojuegos y la inteligencia artificial son dos elementos que siempre han estado fuertemente ligados. Desde el inicio de la computación, la idea de conseguir que un ordenador tenga un comportamiento similar al de una persona humana autónomamente, ha supuesto un gran reto [3].

Los videojuegos son un mundo enorme, repletos de campos de investigación fascinantes, los cuales tienen posibilidades amplias para ser explorados por la gran cantidad de interacciones con varios procesos psicológicos. Los mundos virtuales allí presentes simbolizan un nuevo contexto en el cual las personas pueden interactuar y relacionarse de maneras que en el mundo real serían poco accesibles. Por otra parte, en los videojuegos se pueden experimentar situaciones que requieren diversas habilidades tanto motoras como cognitivas para la consecución de un objetivo determinado.

Cabe destacar que los usuarios de este tipo de tecnologías no solo interactúan con agentes de inteligencia artificial, sino que lo pueden hacer con otras personas permitiendo una amplia gama de actividades psicológicas [4] y cuando se trata de videojuegos competitivos, estos aspectos se llevan al límite. Se han realizado muchas estadísticas de los juegos más populares en 2021, los cuales, si se clasifican, el juego que más resalta y con diferencia, es League of Legends. Llegando a tener alrededor de 100 millones de jugadores activos cada mes. League of Legends (por sus siglas LoL) pertenece a una categoría de juegos llamada arena de combate multijugador en línea, más conocida como M.O.B.A (Multiplayer Online Battlefield Arena) [5].

Al ser tan competitivo las personas buscan adquirir la mayor cantidad de información posible, existen muchos servicios que ofrecen estadísticas a los jugadores, tanto del juego como de sus contrincantes en tiempo real, pero ninguna es capaz de ofrecer algo más complejo. La primera etapa de toda partida consiste en la selección de personajes; los jugadores tienen a su disposición 159 opciones, dejando miles de millones de posibles escenarios diferentes. Aunado a esto existe la creencia común de que las partidas se empiezan a ganar desde que se seleccionan los personajes, ya que algunos pueden tener fortalezas o debilidades dependiendo de a que otros se enfrentan.

A pesar de que es una creencia muy arraigada, aun no existe un sistema o método establecido para conocer realmente si se tiene o no ventaja conforme al equipo enemigo. Bajo este contexto, se plantea realizar una inteligencia artificial capaz de predecir qué equipo será el vencedor, solamente conociendo las características de los personajes seleccionados en el estilo clásico del juego. Información que de ser posible obtener, resultaría en una gran ventaja para aquellos que la aprovechen.

1.2. Redes neuronales artificiales y Deep Learning

Por definición, Deep Learning es un subconjunto dentro del campo del Machine Learning, el cual predica con la idea del aprendizaje desde el ejemplo. Su uso es excelente extrayendo patrones a partir de datos en bruto. Esperando que con el tiempo esos modelos sean capaces de solucionar el problema de forma extremadamente precisa, gracias a dicha capacidad de extraer patrones. Las Redes Neuronales Artificiales (RNA) se crearon para simular los componentes y las funciones del cerebro humano.

Las propiedades de las RNA permiten aplicarlas en problemas de reconocimiento de patrones y clasificación, dado que son capaces de descubrir relaciones no aparentes entre variables, y por la tanto, dan significado a los datos. Se describe un agente inteligente cuyo elemento de desempeño son modelos supervisados de RNA [6]. Como algoritmo de aprendizaje se hará uso del perceptrón multicapa [7] ya que se requiere la clasificación de dos clases (victoria y derrota), es uno de los algoritmos de aprendizaje más estudiados y utilizados en las redes neuronales artificiales, pero por su misma naturaleza presenta notables deficiencias, como por ejemplo la lentitud en su proceso de aprendizaje o su nula flexibilidad al ingreso de nuevas características una vez entrenada.

Con la ayuda de las Redes Neuronales Supervisadas se logrará hacer que la Red Neuronal aprenda y clasifique conforme a la base de datos previamente obtenida. El Algoritmo Backpropagation (BP), es tan solo un Entrenamiento Supervisado. En este algoritmo de Backpropagation se considera una etapa de funcionamiento donde se presenta, ante la red entrenada, un patrón de entrada y éste se transmite a través de las sucesivas capas de neuronas hasta obtener una salida y, luego una etapa de entrenamiento o aprendizaje donde se modifican los pesos de la red de manera que coincida la salida deseada con la salida obtenida por la red [8].

1.3. Trabajos relacionados

La predicción utilizando sistemas de aprendizaje profundo (comúnmente redes neuronales), ha sido manejada en diferentes áreas, como el clima, economía, medicina, industria, etc. Sin embargo, en el videojuego League of Legends no hay formalmente un desarrollo para saber predicciones utilizando Deep Learning, todo lo que tenemos son predicciones basadas en sistemas de Machine Learning y Reglas Fijas impuestas por estadísticas registradas. No obstante, se tiene en cuenta que en donde existen datos masivos, se puede trabajar para obtener predicciones con redes neuronales (Deep

Learning). Lo más que podemos obtener de trabajos previos son estadísticas históricas muy elaboradas referentes al videojuego. En sitios como:

- OP.gg [9]. Proporciona perspectivas a los jugadores de League of Legends acerca de su jugabilidad y sobre el juego en general.
- League Of Graphs [10]. Alternativa similar a OP.gg, proporcionando estadísticas avanzadas y patrones de jugabilidad.
- LolProfile [11]. Página para ver datos históricos sobre jugadores.
- Porofessor.gg [12]. Estadísticas por jugador en tiempo real.
- MOBAlytics [13]. Aplicación de estadísticas históricas y en tiempo real.
- League of Legends Wiki [14]. Proporciona información en general sobre el juego.
- Entre otros...

2. Método

2.1. Obtención de datos

Se tuvo que crear una cuenta en la API [15] de RIOT [16] la cual es la desarrolladora del videojuego. Una vez creada, RIOT asigna una “API Key” [17] por defecto que es necesaria para realizar consultas y obtener los datos.

A través del lenguaje de programación Python [18], se implementó un programa capaz de hacer consultas a la API con base a los datos que se requieren obtener.

En League of Legends existen múltiples modos de juego, en esta aplicación solo se centrará en el estilo clásico de juego ya que este es el modo competitivo.

2.2. Procesamiento de datos

Durante el proceso surgieron varias complicaciones. Una de ellas fue la limitación de consultas a la API en un determinado tiempo.

Por lo tanto, la solución fue que el programa estuviera en ejecución por horas en múltiples computadoras para obtener la cantidad suficiente de datos para un análisis bien sustentado de las partidas. Consiguiendo un total de 60,410 registros.

Los datos a obtenidos se dividieron en dos bases de datos. La primera base de datos se utiliza para almacenar a los campeones (personajes del juego), los cuales tienen sus propias características necesarias que los diferencian unos a los otros como: ataque, armadura, velocidad, magia, dificultad, etcétera.

Tabla 1. Configuraciones propuestas para la red neuronal. [Creación propia].

No.	Atributo	Definición	Descripción
1-10	Champ1-10	Personajes	Se refiere a los 10 personajes utilizados en la partida
11-12	Attack1-2	Ataque	Indica las sumatorias de ataque de los personajes de cada equipo
13-14	Defense1-2	Defensa	Indica las sumatorias de la dificultad de los personajes de cada equipo
15-16	Difficulty1-2	Dificultad	Indica las sumatorias de la defensa de los personajes de cada equipo
17-18	Magic1-2	Magia	Indica las sumatorias de la magia de los personajes de cada equipo
19-20	Armor1-2	Armadura	Indica las sumatorias de la armadura de los personajes de cada equipo
21-22	AttackDamage1-2	Daño de ataque	Indica las sumatorias del daño de ataque de los personajes de cada equipo
23-24	AttackRange1-2	Rango de ataque	Indica las sumatorias del rango de ataque de los personajes de cada equipo
25-26	AttackSpeed1-2	Velocidad de ataque	Indica las sumatorias de la velocidad de ataque de los personajes de cada equipo
27-28	Health1-2	Vida	Indica las sumatorias de la vida de ataque de los personajes de cada equipo
29-30	MagicResist1-2	Resistencia mágica	Indica las sumatorias de la resistencia mágica de los personajes de cada equipo
31-32	Mana1-2	Maná	Indica las sumatorias del maná de los personajes de cada equipo
33-34	MoveSpeed1-2	Velocidad de movimiento	Indica las sumatorias de la velocidad de movimiento de los personajes de cada equipo
35	Win	Victoria	Indica el equipo que ganó la partida
36	Duration	Duración	Indica la duración de la partida

La segunda base de datos es referente a la información de las partidas, requirió un proceso mayor ya que cada uno de los 10 personajes en partida cuenta con sus propias estadísticas, se estarían teniendo 24 características por personaje lo que al acumular los 10 personajes nos deja con 250 datos en cada registro, teniendo un total de 15,102,500 datos, saturar de información irrelevante a una red neuronal puede llevar a resultados ambiguos o erróneos, por lo que se realizó un procesado más profundo de los datos resumiendo la información de los personajes de cada equipo en sumatorias de sus características en común.

Cada equipo contará con las respectivas sumatorias de las características de los campeones seleccionados y finalmente el resultado de la victoria y duración de la partida.

Quedando así la base de datos, normalizada y lista para entrenar la red neuronal. Los patrones de entrada de la arquitectura (1 - 34) y las clases o salidas (35 - 36) quedaron organizados con la siguiente estructura: Previo a la implementación de la red neuronal, se realizó una limpieza a la base de datos para eliminar todas aquellas partidas con una

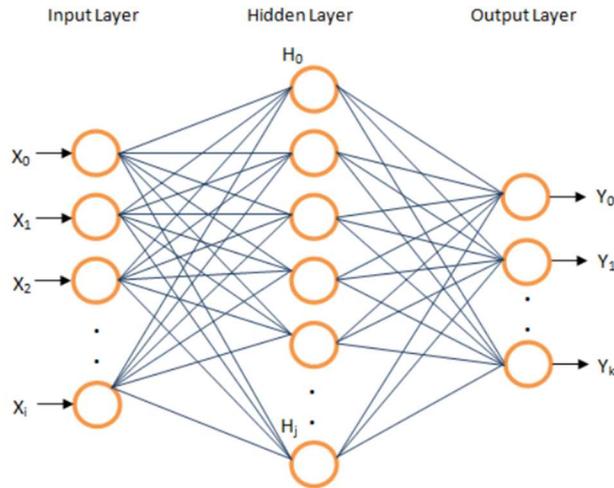


Fig. 1. Ejemplificación del perceptrón multicapa. (Tipo Red Neuronal utilizada) [19].

duración menor a 15 minutos, ya que estas suelen ser partidas que finalizan por factores externos al propio juego como el abandono de jugadores, mantenimiento, perdidas de conexión o errores en los servidores y hacer uso de dichas partidas en el entrenamiento no harían más que generar ruido y por ende un aprendizaje deficiente. Quedándonos con un total de 43,908 registros de 36 datos cada uno (1,580,688 datos en total).

2.3. Desarrollo de la red neuronal

Se utilizó Python como lenguaje de programación ya que el desarrollo de aplicaciones relacionada a la ciencia de datos es más eficiente por la amplia variedad de librerías relacionadas al campo de investigación. La estructura de un perceptrón multicapa modelo luce de la siguiente forma: Se hizo uso múltiples módulos de la librería *scikit-learn* [20], librería especializada en Machine Learning y Deep Learning, específicamente se utilizaron los siguientes módulos:

Classification_report: Este módulo nos permite obtener análisis de resultados complejos. Se tienen 3 datos que son de vital importancia:

Accuracy: Es la relación de predicciones correctas de forma global. Se obtienen al dividir el número total de aciertos entre el número total de registros [21]:

$$\text{Accuracy} = \frac{\text{Verdaderos positivos} + \text{Verdaderos Negativos}}{\text{Todas las muestras}} \quad (1)$$

Recall: Es la relación de predicciones correctas que hay por cada clase. Se obtiene al dividir los aciertos de cada clase entre la cantidad de valores existentes de esa clase en el conjunto de prueba [21]:

$$\text{Recall} = \frac{\text{Verdaderos positivos}}{\text{Todas los positivos reales}} \quad (2)$$

Tabla 2. Configuraciones propuestas para la red neuronal. [Creación propia].

Nº de iteraciones	Nº de neuronas	Nº de capas ocultas
10,000	10	7
10,000	10	5
15,000	50	5

Tabla 3. Propiedades de la función de activación con figura [22].

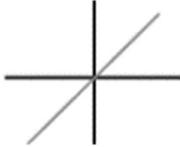
Función de activación	Representación matemática	Figura
Identity	$f(x) = x$	

Tabla 4. Resultados completos de configuración 1 [Creación propia].

	Accuracy	Recall 1	Recall 2
1	47.58	9.17	95.35
2	48.73	2.77	99.19
3	52.02	60.11	41.95
4	55.28	99.17	0.68
5	51.92	52.85	50.76
6	53.06	67.27	35.39
7	55.49	86.71	16.65
8	55.86	75.83	28.79
9	55.03	85.14	17.58
10	51.69	42.24	63.45

Train_test_split: Nos permite dividir rápidamente nuestros datos en dos conjuntos, prueba y entrenamiento. Esto debido a que no se puede hacer uso de los datos de entrenamiento para comprobar la eficacia de la red neuronal ya que se estarían prediciendo datos vistos previamente. Por la gran amplitud de datos que poseemos, se escogió un 88% (53,161 registros) de los datos de manera aleatoria, para realizar el entrenamiento, y un 12% (7,249 Registros) para la fase de pruebas.

Neural_network MLPClassifier: Este módulo contiene herramientas para un rápido desarrollo de la red neuronal Perceptrón Multicapa. Entre los aspectos más importantes al momento de desarrollar este tipo de redes neuronales está la correcta selección de un porcentaje del conjunto de prueba el cuál para este caso fue del 12% de patrones y la cantidad de capas ocultas, iteraciones de entrenamiento, función de activación y neuronas por capa oculta que tendrá el perceptrón. La red neuronal puede ser muy sensible a los cambios de dichos parámetros, por lo que se usaron 3 diferentes configuraciones.

Cabe destacar que en cada configuración se hizo uso de la función de activación identidad, esto con la intención de hacer un análisis individual para cada caso predictivo y así obtener un porcentaje preciso de posible resultado, más sin en cambio, en las

Tabla 5. Resultados resumidos de configuración 1 [Creación propia].

	Accuracy	Recall 1	Recall 2
Mínimo de aciertos	47.58%	2.77%	0.68%
Máximo de aciertos	55.86%	99.17%	99.19%
Promedio	52.666%	58.126%	44.979%

Tabla 6. Resultados completos de configuración 2 [Creación propia].

	Accuracy	Recall 1	Recall 2
1	55.39	97.56	2.93
2	50.99	47.03	55.91
3	48.41	10.68	95.35
4	44.8	1.57	98.59
5	45.64	8.04	92.41
6	45.43	2.7	98.59
7	55.94	92.29	10.73
8	55.03	82.09	21.37
9	45.35	2.25	98.78
10	46.36	8.42	93.56

Tabla 7. Resultados resumidos de configuración 2 [Creación propia].

	Accuracy	Recall 1	Recall 2
Mínimo de aciertos	44.8%	1.57%	2.93%
Máximo de aciertos	55.94%	97.56%	98.78%
Promedio	49.334%	35.263%	66.822%

salidas finales los resultados se tornan a sus extremos, dando un resultado de victoria o derrota:

3. Resultados

Si bien el desempeño de la red neuronal suele ser inconsistente se realizaron pruebas con las configuraciones propuestas para la red neuronal (ver tabla 2) para confirmar la consistencia de los datos obtenidos y asegurar la mínima variabilidad al cambiar los factores de influencia.

Para cada configuración fueron realizadas 10 redes neuronales con inicialización de pesos aleatorios para evitar cualquier influencia en los resultados.

Configuración 1: (Número de iteraciones = 10000, número de neuronas = 10, número de capas ocultas = 5). Los resultados completos se muestran en las Tablas 4 y 5. La primera red neuronal mostró un promedio de Accuracy muy cercano al 50% lo que no queda muy lejos del azar y los aciertos en las predicciones de victoria y derrota de igual manera están muy alejados unos de otros por lo que pareciera no existir relevancia en las predicciones.

Tabla 8. Resultados completos de configuración 3 [Creación propia].

	Accuracy	Recall 1	Recall 2
1	55.38	85.34	18.1
2	48.35	9.34	96.89
3	47.35	7.36	97.1
4	49.74	19.51	87.35
5	55.45	90.89	11.37
6	55.53	98.45	2.12
7	55.39	88.92	22.4
8	53.82	72.61	30.45
9	51.43	37.11	69.25
10	52.68	53.02	52.25

Tabla 9. Resultados resumidos de configuración 3 [Creación propia].

	Accuracy	Recall 1	Recall 2
Mínimo de aciertos	47.35%	7.36%	2.12%
Máximo de aciertos	55.53%	98.45%	97.1%
Promedio	52.512%	56.255%	48.728%

Configuración 2: (Número de iteraciones = 10000, número de neuronas = 10, número de capas ocultas = 7). Los resultados completos se muestran en las Tablas 6 y 7.

Configuración 3: (Número de iteraciones = 15000, número de neuronas = 50, número de capas ocultas = 5). Los resultados completos se muestran en las Tablas 8 y 9. Como se puede observar los resultados con diferentes redes neuronales se mantienen con un margen de mínima alteración, rondando el 50% de precisión.

4. Conclusiones y trabajo a futuro

Se partió de la hipótesis de que los personajes seleccionados en una partida clásica tenían la capacidad de influir sobre el resultado de esta respecto a qué equipo sería el ganador, creencia que es muy popular entre la comunidad y tomada casi como una verdad absoluta. Realizando el análisis de las 43,908 partidas con redes neuronales no ha sido posible el obtener más allá de un 51.504% de precisión al intentar predecir la victoria con esta información.

Esto es prácticamente equivalente a tirar una moneda al aire para conocer el resultado de una partida clásica en el videojuego League of Legends. Conocer esta información nos resulta relevante ya que no solo demuestra que dicha creencia popular es falsa, sino que, el cómo concluye cada enfrentamiento en el juego depende de las decisiones y acciones que ejecute cada persona y no de la herramienta que decida utilizar, en este caso sus personajes. Actualmente existen herramientas que antes de

empezar una partida proporcionan estadísticas de los personajes brindando una supuesta ventaja de conocimiento a los jugadores.

Lo cual queda demostrado como una pérdida de tiempo valioso en la preparación de una partida, ya que este tiempo no supera los dos minutos. Al demostrar que los resultados de las partidas están influenciados completamente por el comportamiento de los jugadores podría resultar viable no analizar los personajes sino a los mismos jugadores, datos como: tiempo de experiencia en el juego, victorias y derrotas de partidas recientes, experiencia con el personaje, tendencias de jugabilidad entre otros.

Este tipo de datos podría proporcionar resultados basados en comportamientos y tendencias humanas cosas como el cansancio del jugador, su humor o incluso su grado de concentración. También queda la opción a futuro de no solo analizar las victorias ya que se sabe que los jugadores son la principal influencia en el juego se puede intentar predecir la duración de las partidas y datos relevantes respecto al comportamiento que tendrán los jugadores durante la partida otorgando una ventaja competitiva a aquellos que posean dicha información.

Referencias

1. Oracle México: ¿Qué es la inteligencia artificial (Artificial Intelligence, AI)? (2022)
2. Pérez Orozco, B., Rentería Rodríguez, M. E.: Inteligencia Artificial. INCyTU. Oficina de Información Científica y Tecnología para el Congreso de la Unión. Nota-INCyTU, no. 12, pp. 1 (2018)
3. Utande, P.: Aplicación de Inteligencia Artificial en videojuegos. Uso de la variante del algoritmo MINIMAX PODA Alpha-Beta para su desarrollo. Escuela Técnica Superior de Ingeniería en Sistemas Informáticos. Universidad Politécnica de Madrid. Madrid, España. pp. 5 (2017)
4. Leyva-Rodríguez, J. E., Varela-García, J. D.: El videojuego League of Legends y su efecto en memoria de trabajo visual y solución de problemas. Programa de Psicología. Escuela de Medicina y Ciencias de la Salud. Universidad del Rosario. Bogotá, Colombia. pp. 6 (2016)
5. League of Legends. Interaxion group (2019)
6. Mariño, S. I., Primorac, C. R.: Diseño de un agente inteligente basado en una red neuronal artificial supervisada. Validación en un dominio botánico. Revista de la Escuela De Perfeccionamiento en Investigación Operativa, vol. 29, no. 49, (2021)
7. IBM. Perceptrón multicapa. Documentación de IBM (2021)
8. Hidalgo-Cajo, I. M., Yasaca-Pucuna, S., Hidalgo-Cajo, B. G., Hidalgo-Cajo, D. P., Latorre-Benalcázar, N. B.: Estudio comparativo de los algoritmos BackPropagation (BP) y multiple linear regression (MLR) a través del análisis estadístico de datos aplicado a redes neuronales artificiales. Revista Boletín Redipe, vol. 9, no. 3, pp. 144–152 (2020) doi: 10.36260/rbr.v9i3.939.
9. OP.gg. LoL Stats, Record Replay, Database, Guide (2022)
10. LeagueOfGraphs. LoL Champions, Summoners Stats & Rankings (2022)
11. LolProfile. League of Legends: Summoner Search & Stats (2022)
12. Porofessor.gg. Búsqueda de partidas en directo de League of Legends, y estadísticas de jugadores en tiempo real. (2022)

13. MOBAlytics. League of Legends: Summoner Stats, Match History and Champions Builds (2022)
14. League of Legends Wiki. Resource for League of Legends by a Wiki Fandom (2022)
15. Red Hat. ¿Qué es una API?. El concepto de las interfaces de programación de aplicaciones (2017)
16. Riot Games. Riot Developer Portal. League of Legends developer community with access to game data (2021)
17. Surática Software. ¿Qué es una API Key?. Diccionario (2021)
18. Python Software Foundation. Python 3.10.3 documentation (2022)
19. Neural Networks and MLP. Red Neuronal Perceptron Multicapa. DotNetLovers (2020)
20. scikit-learn. Machine Learning in Python (2022)
21. Lawtomed. 4 Things you need to know about AI: accuracy, precision, recall and F1 scores (2019)
22. Karakaya-Ulucan, D., Ulucan, O., Turkan, M.: Electronic Nose and Its Applications: A Survey. *International Journal of Automation and Computing*. Vol. 17, pp. 179–209, (2019) doi: 10.1007/s11633-019-1212-9.
23. Guerrero, S.: *Machine Learning, el futuro de la Inteligencia Artificial* (2019)
24. Neural Networks and MLP. Red Neuronal Perceptron Multicapa. DotNetLovers (2020)

Segmentación probabilística de carriles mediante parametrización lineal de baja dimensión

Carlos Acuña, Mario Castelán, Gustavo Arechavaleta

Instituto Politécnico Nacional,
Centro de Investigación y de Estudios Avanzados
Grupo de Robótica y Manufactura Avanzada
México

{mario.castelan, garechav}@cinvestav.edu.mx
carlos.acuna@cinvestav.mx

Resumen. La detección de carriles es un módulo importante para los sistemas de seguridad activa, ya que permite aumentar la seguridad y reducir los accidentes de tráfico causados por falta de atención del conductor. Los cambios de iluminación u oclusiones hacen que la detección de carriles sea una tarea desafiante, sobre todo si la detección se realiza a partir de una sola imagen. En consecuencia, este trabajo presenta un enfoque probabilístico basado en el filtro de Kalman, que mediante información de cuadros de imágenes anteriores logra estimar el carril que no pudo ser detectado en el cuadro de imagen actual, considerando la incertidumbre tanto en la predicción como en la detección. Para ello, se introduce un análisis de componentes principales de la curvatura segmentada con el fin de reducir la dimensionalidad, pasando de una representación de píxeles de gran dimensión a una representación de espacio considerablemente reducido.

Palabras clave: Detección de carril, filtro de Kalman, reducción de dimensionalidad.

Probabilistic Lane Segmentation Using a Low-dimensional Linear Parametrization

Abstract. Lane detection is an important module for active safety systems, as it increases safety and reduces traffic accidents caused by driver inattention. Illumination changes or occlusions make lane detection a challenging task, especially if the detection is performed from a single image. Consequently, this paper presents a probabilistic approach based on the Kalman filter, which using information from previous image frames manages to estimate the lane that could not be detected in the current image frame, considering uncertainty in the prediction as well as in the detection. To this end, a principal component analysis of the segmented curvature is introduced with the purposes of dimensionality reduction, moving from a large dimensional pixel representation to a considerably reduced space representation.

Keywords: Lane detection, Kalman filter, dimensionality reduction.

1. Introducción

La robótica autónoma tiene como objetivo percibir el mundo físico a través de dispositivos mecánicos controlados por computadora para navegar en consecuencia [12]. La fiabilidad de la navegación autónoma depende directamente de la capacidad de percibir el entorno. En los vehículos autónomos se han incorporado sistemas avanzados de asistencia al conductor para alertar en situaciones de peligro o para participar activamente en la conducción. El principal cuello de botella en el desarrollo de estos sistemas es el problema de la percepción [11], que tiene dos elementos: la detección de carriles y de obstáculos.

Tanto en la detección como en el seguimiento de carriles, existen básicamente dos clases de representación de carriles: las representaciones basadas en características y las basadas en modelos. La primera parametriza los carriles combinando características de bajo nivel, como líneas pintadas, bordes y segmentos de carril. En consecuencia, esta técnica depende del número de características extraídas del carril y tiene la desventaja de no imponer restricciones sobre la forma o la longitud [14]. Por otro lado, las técnicas basadas en modelos solo utilizan algunos parámetros para representar los carriles, asumiendo que las formas de los carriles pueden parametrizarse mediante modelos matemáticos, como líneas rectas, curvas parabólicas, espirales o splines.

Los enfoques basados en modelos son más robustos frente al ruido y la falta de datos que las técnicas basadas en características [3]. Para abordar los problemas anteriores, en este artículo presentamos un detector de carril, basado en el filtro de Kalman, con el fin de incorporar información de cuadros de imágenes¹ anteriores. En aras de lograr esta tarea, se desarrolló un Análisis de Componentes Principales (PCA) generado a partir de rutas parametrizadas de carriles segmentados, obteniendo modelos de carriles basados en ejemplos, los cuales son ideales para la reducción de dimensionalidad.

Como consecuencia, en lugar de tener el filtro de Kalman definido en el espacio de píxeles (considerando cien posiciones de píxeles), usamos un espacio proyectado de PCA reducido de tres parámetros. El resto del artículo está organizado de la siguiente manera: en la sección 2 se abordan detalles de la detección determinística de carriles y la construcción del modelo PCA; en la sección 3 se muestra la parametrización en términos de la reducción de dimensionalidad; en la sección 4 se presenta el algoritmo propuesto basado en filtro de Kalman; la sección 5 describe la evaluación experimental del método propuesto y, finalmente, la sección 6 ofrece comentarios finales e ideas para el trabajo futuro.

1.1. Trabajo relacionado

La detección de carril es un desafío importante para hacer posible el futuro de los vehículos autónomos. En los últimos años, se han propuesto varios métodos basados en técnicas de filtrado y procesamiento de imágenes, tal como se informa en la literatura [6]. Dichos trabajos se han centrado en resolver el problema utilizando métodos clásicos de visión por computadora, es decir, segmentación de color y operadores de gradiente, [13].

¹ imagen estática registrada cada determinado tiempo t_s .

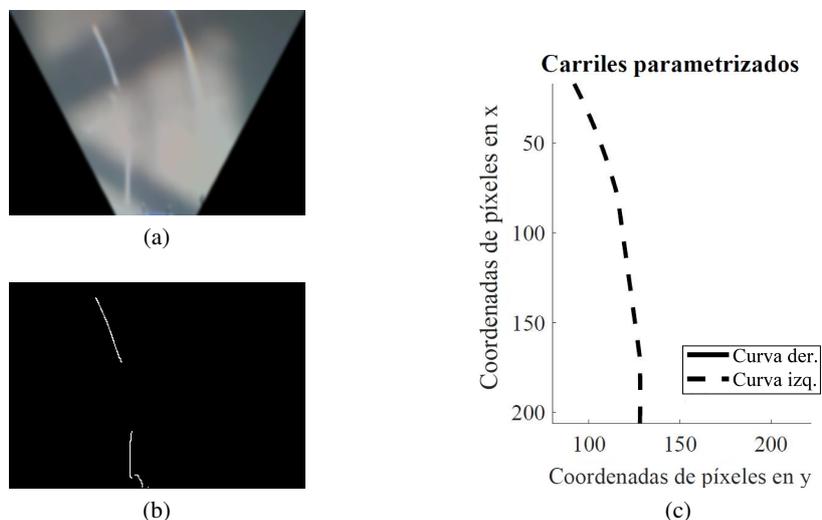


Fig. 1. Caso de carriles no segmentables. (a) Cuadro de imagen con homografía. (b) Cuadro de imagen aplicando detección de bordes. (c) Parametrización de carriles de un marco de imagen con exposición a la luz.

Por ejemplo, en [5], el espacio RGB se usó para encontrar la marca del carril en contraste con el color de la carretera a través de un umbral, que se definió de acuerdo con experimentos realizados en diferentes momentos del día. Sin embargo, este método mostró problemas en presencia de sombras proyectadas sobre el carril. Otros enfoques han explorado el efecto del uso de operadores de gradiente sobre imágenes en escala de grises, como [1] basado en el algoritmo de Canny, que realiza la detección de bordes con un tiempo de procesamiento reducido.

Desafortunadamente, los métodos anteriores no consideran la información previa, por lo tanto, no se benefician de la continuidad que exhiben naturalmente las trayectorias de los carriles [15]. Una forma alternativa de abordar el problema del cuadro de imagen único en la detección de carriles es a través de enfoques probabilísticos, los cuales fueron propuestos para mejorar la solidez y la estabilidad del seguimiento de carriles [7]. Por ejemplo, en [2] se diseñó un filtro de partículas en cascada para trabajar con un sistema no lineal y reducir el costo computacional del filtro de partículas convencional.

Sin embargo, el principal problema de este filtro es que las partículas se degradan a medida que disminuye la importancia del peso de cada partícula. Otro filtro comúnmente utilizado en el seguimiento de carriles es el filtro de Kalman que permite modelar y reducir la variación de la estimación de carriles a través de una etapa de corrección, obteniendo coherencia temporal [9].

En aplicaciones prácticas de este filtro, los parámetros de entrada son generalmente desconocidos y ajustarlos es una tarea difícil. Si se realiza manualmente, esto representaría una carga considerable para el usuario. Varios trabajos han desarrollado estrategias de autoajuste como en [8,10], donde se demostró que es posible controlar el comportamiento del filtro ante la presencia de valores atípicos.

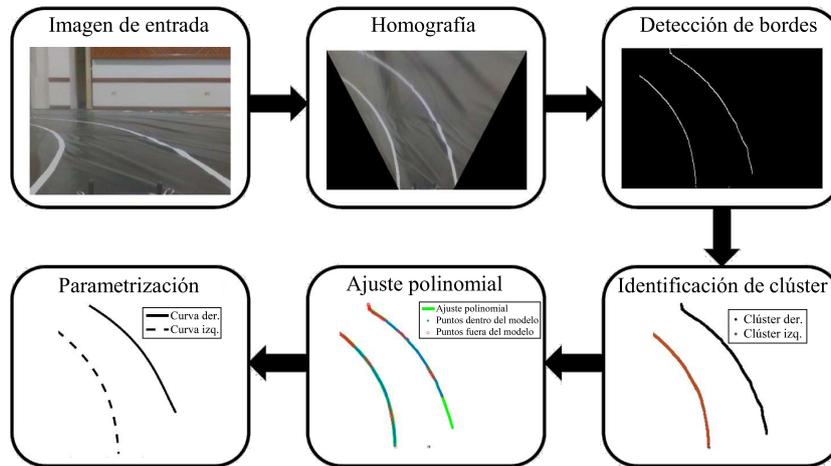


Fig. 2. Modelo determinístico de detección de carril. Se muestran las diferentes etapas que componen al modelo de medición.

2. Materiales y métodos

Nuestro objetivo es obtener tanto una predicción como un modelo lineal que sean adecuados para ser utilizados en un marco de filtro de Kalman que a partir de la información disponible en el momento $t - 1$, realiza un paso de actualización mediante información adicional disponible en el momento t . Teniendo en cuenta la naturaleza lineal del filtro, realizamos un PCA basado en carriles segmentados parametrizados a partir de un entrenamiento obtenido en una competición de vehículos a escala. En la figura 1 se puede notar que no todos los carriles son segmentables, principalmente por problemas de iluminación.

2.1. Detección determinística de carriles

El modelo de medición se basa en la extracción de características implementando el algoritmo de Canny y en la parametrización de curvas mediante polinomios de primer orden para formas rectas y polinomios de tercer orden para formas curvas. Con esto, es posible recopilar rutas para una base de datos y luego representar dichas rutas a través de un conjunto de puntos equidistantes. Para distinguir los carriles izquierdos de los derechos, se utiliza una técnica de agrupación jerárquica [4]. El proceso del modelo de medición, que se utilizó para construir la base de datos de carriles, se muestra en la Figura 2.

2.2. Reducción de dimensionalidad

Uno de los principales objetivos de este trabajo es construir un modelo lineal de carriles segmentados que permita la proyección y reproyección de formas a partir de nuevos ejemplos.

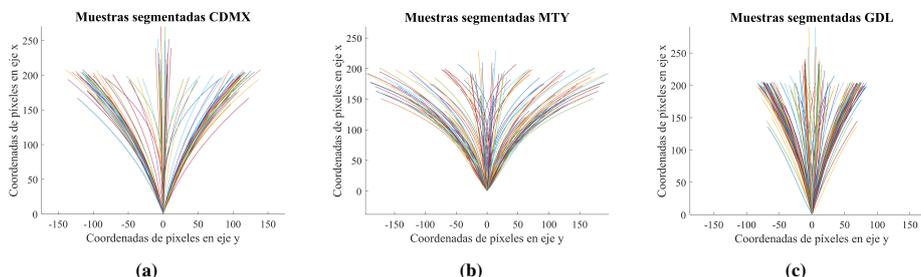


Fig. 3. Bases de datos de muestra segmentadas. (a) Base de datos CDMX, 82 curvas y 38 carriles rectos. (b) Base de datos MTY, 76 curvas y 40 carriles rectos. (c) Base de datos GDL, 154 curvas y 48 carriles rectos.

Nuestros datos fueron recolectados durante tres competencias de vehículos autónomos organizadas por la Federación Mexicana de Robótica, que se llevaron a cabo en la Ciudad de México (CDMX), Monterrey (MTY) y Guadalajara (GDL), de 2017 a 2019. Las rutas segmentadas de cada base de datos se muestran en la figura 3.

En nuestros experimentos, cada carril está representado por una secuencia finita discreta de $m = 102$ puntos (x, y) como $\mathbf{t}_{nx1} = [x_1, x_2, \dots, x_m, y_1, y_2, \dots, y_m]^T$, donde $n = 2m$ es el número de elementos de cada vector \mathbf{t} . Con la ruta promedio $\bar{\mathbf{t}}$ se contruye la matriz de deatos de entrenamiento centrada $\mathbf{T}_{n \times k} = [(\mathbf{t}_1 - \bar{\mathbf{t}}) | \dots | (\mathbf{t}_k - \bar{\mathbf{t}})]$.

El análisis de componentes principales busca un conjunto de $n - 1$ vectores ortogonales que, en el sentido de mínimos cuadrados, minimicen la correlación entre las columnas de \mathbf{T} . La solución se encuentra calculando los vectores propios de la matriz de covarianza $\Sigma_{n \times n} = \mathbf{T}\mathbf{T}^T$. Como Σ es simétrica, siempre existe una base ortogonal $\mathbf{U}_{n \times n}$ y una matriz diagonal $\Lambda_{n \times n}$ que satisface la relación:

$$\Sigma = \mathbf{U}\Lambda\mathbf{U}^T, \quad (1)$$

donde $\mathbf{U}_{n \times n}$ es la matriz de vectores propios y los valores propios de Σ son los elementos diagonales de la matriz Λ . Normalmente, se elige un número de vectores propios l para que se conserve una variabilidad suficiente en el entrenamiento. A partir de ahora, nos referiremos a la matriz $\hat{\mathbf{U}}_{n \times l}$ como el modelo:

$$\mathbf{b} = \hat{\mathbf{U}}^T(\mathbf{t}_0 - \bar{\mathbf{t}}). \quad (2)$$

La aproximación del carril fuera de entrenamiento \mathbf{t}_0 viene dada por:

$$\mathbf{t}_0 \approx \bar{\mathbf{t}} + \hat{\mathbf{U}}\mathbf{b}. \quad (3)$$

3. Parametrización en dimensiones reducidas

De acuerdo con nuestros resultados, se observó que tres vectores propios fueron suficientes para capturar al menos 98 % de la variabilidad del modelo, para lo cual la matriz $\hat{\mathbf{U}}$ se fijó en dimensiones 204×3 . En consecuencia, usando el modelo es posible ajustar una ruta fuera de entrenamiento usando solo tres vectores propios.

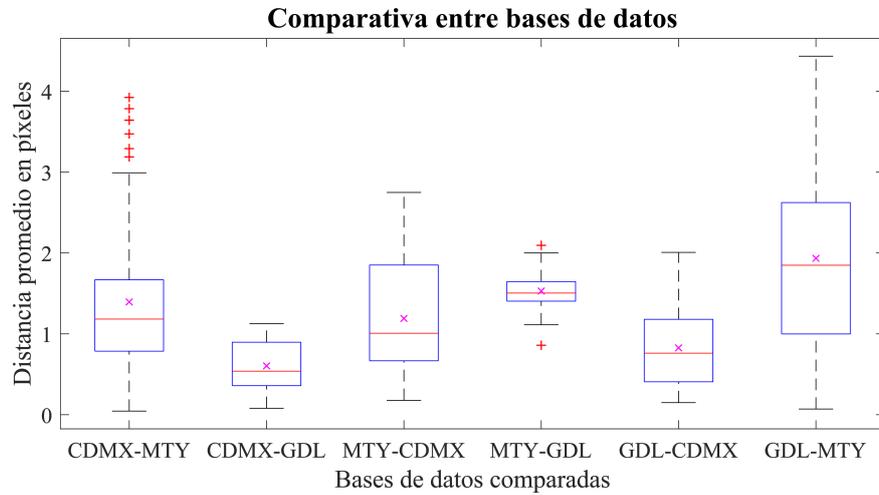


Fig.4. Selección de la base de datos. Cada diagrama de caja se relaciona con un par de entrenamiento-prueba de carriles segmentados.

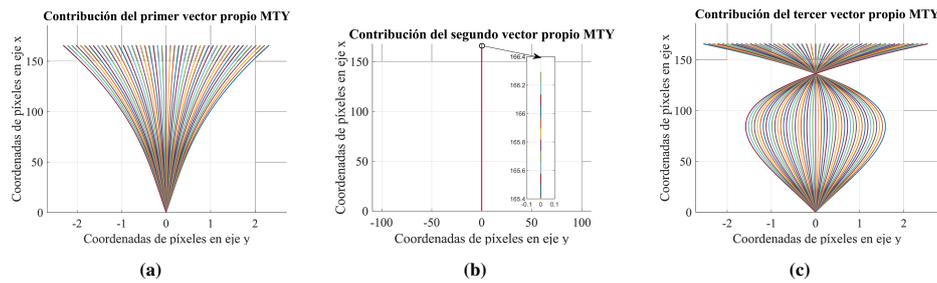


Fig.5. Contribución de los primeros tres vectores propios del modelo MTY. (a) Las variaciones del primer vector propio muestran contribuciones relacionadas con los cambios de curvatura. (b) se relaciona con la longitud. (c) El tercer vector propio se relaciona con el alargamiento de la curvatura.

La figura 4 muestra el comportamiento de los pares de entrenamiento y prueba entre los diferentes conjuntos de datos utilizados en nuestros experimentos. Los diagramas de caja muestran que los datos MTY representan el modelo más robusto al ingresar carriles fuera de entrenamiento, ya que presenta el error de dispersión más pequeño.

Recordemos que, para lograr la reducción de dimensionalidad, primero una imagen de tamaño 480×640 se somete a la segmentación de carriles; luego, se forma un vector 204×1 a través de las coordenadas (x, y) adquiridas de los carriles segmentados; y finalmente, el modelo lo transforma en un vector de dimensión reducida de 3×1 .

Para conocer la contribución de cada vector propio, éstos se re proyectaron mediante la expresión 3 variando el coeficiente de b entre los valores de $[-12$ a $12]$, que representan el 66 por ciento de la distribución de datos b . Las contribuciones de forma y longitud de los carriles re proyectados resultantes se muestran en la Figura 5.

La figura 5a muestra que la contribución del primer vector propio corresponde a las variaciones de izquierda a derecha de los carriles, los cuales presentan una curvatura constante. Así, este vector propio tendrá una mayor participación en la reproyección de los carriles que corresponden a las curvas del camino. La contribución del segundo vector propio corresponde a un alargamiento con respecto al eje de coordenadas y , como se muestra en la figura 5b.

Alternativamente, la figura 5c presenta una contribución relacionada con un punto de inflexión, caracterizando otro tipo de curvatura observada en los carriles. El conocimiento previo de la contribución de los vectores propios permite diseñar una estrategia para el autoajuste de la covarianza dentro del filtro de Kalman propuesto.

4. Segmentación probabilística basada en filtro de Kalman

El filtro de Kalman para la segmentación probabilística se formula a partir del análisis de reducción de dimensionalidad 2.2, que muestra un modelo generado por PCA que permite la reproyección de las rutas segmentadas en términos de tres parámetros. Dado que este modelo es lineal, el filtro de Kalman proporciona una solución recursiva eficiente del método de mínimos cuadrados que consta de dos etapas principales: predicción y actualización, la representación de estado de ambos se muestra en las ecuaciones 4 y 5, respectivamente:

$$\mathbf{x}_t = \mathbf{A}_t \mathbf{x}_{t-1} + \mathbf{B}_t \mathbf{u}_t + \epsilon_t, \quad (4)$$

donde $\mathbf{A}_{n \times n}$ es la matriz de estado, n es la dimensión del vector de estado \mathbf{x}_t . La matriz de control \mathbf{B}_t , es de tamaño $n \times m$, y m es la dimensión del vector de control \mathbf{u}_t . La variable aleatoria ϵ_t es un vector aleatorio gaussiano de la misma dimensión que el vector de estado, con media cero y covarianza \mathbf{R}_t :

$$\mathbf{z}_t = \mathbf{C}_t \mathbf{X}_t + \delta_t. \quad (5)$$

Aquí, \mathbf{C}_t es una matriz de tamaño $k \times n$, donde k comparte la misma dimensión que el vector de medición \mathbf{z}_t . El vector δ_t describe el ruido de la medición y su distribución es gaussiana multivariante con media cero y covarianza \mathbf{Q}_t .

4.1. Modelo de medición

De acuerdo con la ecuación del modelo de medición del filtro de Kalman en su forma de espacio de estados (ec. 5), y la ecuación 2 correspondiente a la reproyección de la ruta segmentada, el modelo de medición del método de segmentación probabilística queda definido por la siguiente expresión:

$$\mathbf{z}_t = \widehat{\mathbf{U}}^T \mathbf{x}_{t,d} + \delta_t, \quad (6)$$

donde $\widehat{\mathbf{U}}$ es el modelo PCA de dimensión 204×3 y $\mathbf{x}_{t,d}$ es el carril segmentado dado por el detector de carril determinista con dimensión 204×1 .

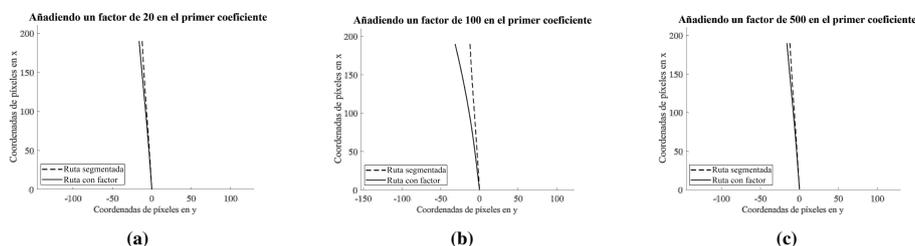


Fig. 6. Variación positiva del primer coeficiente. (a) Añadiendo un factor de 20. (b) Añadiendo un factor de 100. (c) Añadiendo un factor de 500.

4.2. Modelo de predicción

A nivel de imagen, no hay control sobre el comportamiento de los carriles. Por ello, la predicción dependerá directamente del ruido gaussiano añadido, transformando la ecuación 4, correspondiente a la predicción en su forma de espacio de estados como:

$$\mathbf{x}_t = \hat{\mathbf{U}}^T \mathbf{x}_{t-1} + \epsilon_t, \quad (7)$$

donde \mathbf{x}_{t-1} es el estado previo correspondiente a un carril de dimensión 204×1 , re proyectado por la matriz $\hat{\mathbf{U}}$ del modelo PCA de dimensión 204×3 , y ϵ_t es el ruido gaussiano con media cero y matriz de covarianza \mathbf{R}_t de tamaño 3×3 .

4.3. Formulación del algoritmo de filtro de Kalman propuesto

Es importante señalar que no todos los carriles son segmentables por condiciones de iluminación u oclusiones, por tal motivo, el filtro propuesto en este trabajo realiza una estrategia para corregir tales situaciones. Esta estrategia se inspiró en [8], en la que se presentó un método para estimar la covarianza del ruido a partir de los datos del proceso, realizando una actualización de la covarianza durante los periodos transitorios y permitiendo así una rápida convergencia del filtro de Kalman ante la presencia de cambios repentinos en la observación. Este cambio en la segmentación de carriles surge por no detectar una ruta en el modelo de observación, con base en lo anterior, nuestra estrategia mantiene matrices de covarianza constantes mientras que el modelo de medición detecta una ruta y actualiza las covarianzas al no detectar una ruta.

4.3.1. Estrategia de actualización para matrices de covarianza

De acuerdo al análisis realizado en la sección 3, se observó que el primer vector propio corresponde a una contribución respecto a las variaciones de izquierda a derecha de los carriles, así como en la curvatura de la ruta proyectada. Esta contribución es significativa a partir de un factor del orden de las centenas como se muestra en la figura 6, donde se puede observar que al variar con valores positivos, la ruta comienza a adquirir una mayor curvatura, así como un desplazamiento de el punto final a la izquierda; para valores negativos, el comportamiento presenta simetría reflectiva.

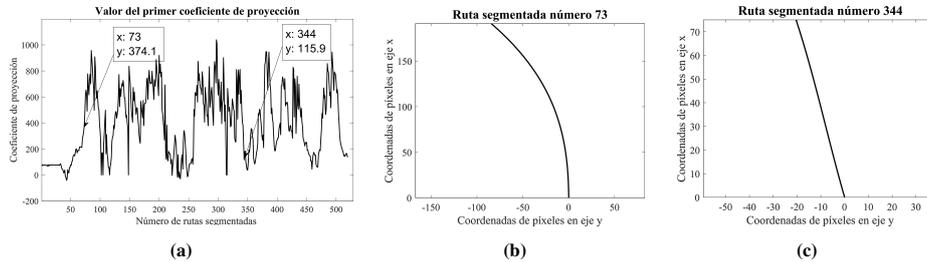


Fig. 7. Definición de umbral para conocer el estado de la ruta segmentada. (a) Primer coeficiente con respecto a la ruta segmentada. (b) Ruta 73 correspondiente a una curva. (c) Ruta 344 correspondiente a una línea recta.

Las figuras 6b y 6c muestran un cambio significativo en la curvatura cuando se suma un factor del orden de centenas, presentando una diferencia de distancia promedio de 8,11 y 40,56 píxeles respectivamente, mientras que en la figura 6a esta distancia es de 1,62 píxeles con un factor de 20. Con este conocimiento previo se puede anticipar el comportamiento de la predicción cuando no se detecta una ruta $x_{t,d}$, identificando si el último estado corresponde a una línea o una curva.

Para obtener un umbral del primer coeficiente que determina si el estado corresponde a una línea recta o a una curva, se segmentaron 520 rutas de la base de datos MTY, de donde se obtuvieron los coeficientes de proyección (ver Figura 7a). Se observó que para valores mayores a 250 del primer coeficiente la ruta corresponde a una curva (ver figura 7b) y para valores menores a 250 la ruta corresponde a una línea recta (ver Figura 7c). Con el umbral que define el estado de la ruta y el conocimiento sobre el comportamiento de la ruta ante la variación de los coeficientes, se desarrolla la estrategia de actualización de las matrices de covarianza en el filtro de Kalman.

4.3.2. Algoritmo de segmentación propuesto basado en el filtro de Kalman

El algoritmo 1 muestra, con base en la sección anterior, la estrategia para actualizar las covarianzas en ausencia de una ruta detectada $x_{t,d}$. En este algoritmo de actualización de covarianza, la función $diag(\text{vec})$ devuelve una matriz diagonal cuyos elementos de la diagonal principal contienen el vector vec . El algoritmo muestra, en la línea 3, que al no obtener una ruta detectada $x_{t,d}$ la incertidumbre en el modelo de medición aumenta por el factor 12. Por otro lado, en el modelo de predicción el factor 0,5 disminuye la incertidumbre. Estos factores aumentan linealmente por cada instante en que no existe ruta $x_{t,d}$ mediante el contador c que aumenta en Δc .

Es importante señalar que se utilizan factores de escala positivos para esta actualización de matrices de covarianza, lo que garantiza que las matrices resultantes Q_t y R_t sean siempre simétricas y definidas positivas. Esto asegura que la estimación resultante mantendrá una distribución normal. En la línea 7 se calcula la diferencia entre la ruta segmentada y la estimación anterior en el espacio de dimensión reducida, lo que permite que desde las líneas 8 hasta la 16 se definan matrices de covarianza según el valor de la diferencia de los coeficientes.

Algoritmo 1 Actualización de covarianzas ($\bar{x}_t, x_{t,d}, R_t, Q_t$).

```

1: Inicio
2:    $c = 1; \Delta_c = 0,2; C_s = 100;$ 
3:   mientras  $x_{t,d} == \{\}$  hacer                                ▷ No hay información en el sensor visual
4:      $F\_Q_t = c * \text{diag}([12, 12]), \quad F\_R_t = c * \text{diag}([0,5, 0,5])$ 
5:     si  $x_{t,d}[1] > 250$  entonces                                ▷ Curva
6:        $M\_R_t = \text{diag}([mrt, 1, 1])$ 
7:        $\Delta_x = x_{t,d} - \bar{x}_t$ 
8:       si  $\Delta[1] > C_s$  entonces
9:          $R_t = F\_R_t * \text{diag}([R_t[1, 1], 0,1, 0,1]) * M\_R_t$ 
10:         $Q_t = F\_Q_t * \text{diag}([Q_t[1, 1], 0,1, 0,1])$ 
11:       si  $\Delta[2] > C_s$  entonces
12:          $R_t = F\_R_t * \text{diag}([0,1, R_t[2, 2], 0,1]) * M\_R_t$ 
13:          $Q_t = F\_Q_t * \text{diag}([0,1, Q_t[2, 2], 0,1])$ 
14:       si  $\Delta[1] > \Delta[2] > C_s$  entonces
15:          $R_t = F\_R_t * \text{diag}([R_t[1, 1], R_t[2, 2], 0,1]) * M\_R_t$ 
16:          $Q_t = F\_Q_t * \text{diag}([Q_t[1, 1], Q_t[2, 2], 0,1])$ 
17:        $c = c + \Delta_c$ 
18:   devolver  $Q_t, R_t$ 

```

Algoritmo 2 Filtro de Kalman propuesto - segmentación ($x_{t-1}, \bar{\Sigma}_t, x_{t,d}, Q_t, R_t$).

```

1: Inicio
2:   si  $t==1$  entonces
3:      $\bar{x}_t = \hat{U}^T(x_{t-1} - \bar{t})$ 
4:   si no
5:      $\bar{x}_t = x_t$ 
6:   si  $x_{t,d} == \{\}$  entonces
7:      $[Q_t, R_t] = \text{Actualizacion\_de\_covarianzas}(\bar{x}_t, x_{t,d}, R_t, Q_t)$ 
8:      $\bar{\Sigma}_t = \bar{\Sigma}_t + R_t$ 
9:      $K_t = \bar{\Sigma}_t(\bar{\Sigma}_t + Q_t)^{-1}$ 
10:     $x_t = \bar{x}_t + K_t(x_{t,d} - \bar{x}_t)$ 
11:     $\Sigma_t = (I - K_t)\bar{\Sigma}_t$ 
12:   return  $x_t, \Sigma_t$ 

```

En caso de que la última ruta detectada sea una curva, se suma un factor para obtener una variación en el primer coeficiente de la estimación. Esto determina la transición de estimaciones en tramos de curva donde no es posible segmentar un carril. Dadas las ecuaciones 6 y 7 correspondientes al modelo de medición y predicción, respectivamente, se propone el algoritmo 2 basado en el filtro de Kalman diseñado en el espacio reducido de tres dimensiones.

En la línea 3 se inicia el filtro considerando una hipótesis del estado anterior x_{t-1} . En este proceso, las matrices de covarianza se mantienen constantes ante la segmentación exitosa de $x_{t,d}$. Sin embargo, cuando la detección de carril no está disponible, las covarianzas se actualizan de acuerdo al algoritmo 1. En la línea 9 se calcula la ganancia de Kalman K_t , la cual regula la contribución de la diferencia entre

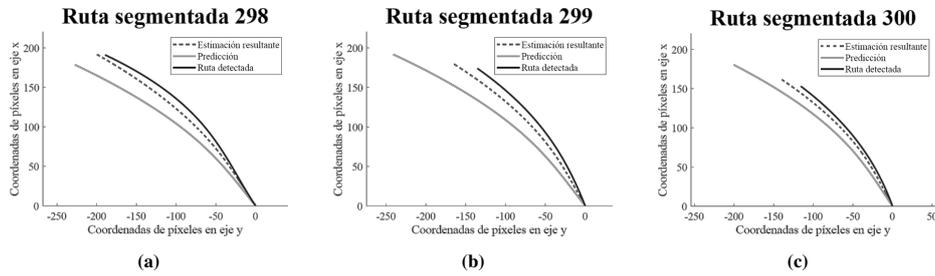


Fig. 8. Estimación del método de segmentación probabilística. (a) Ruta segmentada 298. (b) Ruta segmentada 299. (c) Ruta segmentada 300.

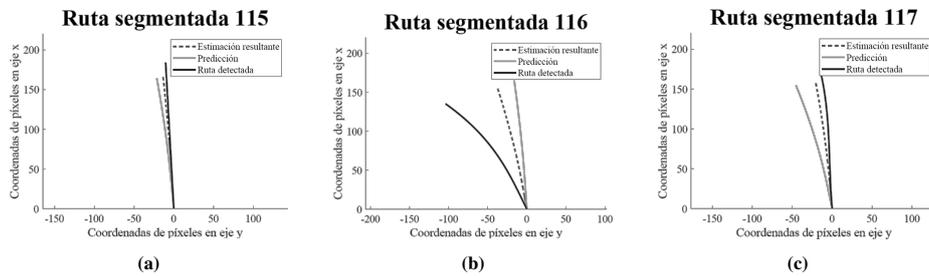


Fig. 9. Regulación de la estimación ante una observación atípica. (a) Ruta segmentada 115. (b) Ruta segmentada 116. (c) Ruta segmentada 117.

la ruta detectada $\mathbf{x}_{t,d}$ y la estimación previa $\bar{\mathbf{x}}_t$, definida en la línea 10. Finalmente, la línea 11 actualiza la matriz de incertidumbre Σ_t . Los resultados obtenidos por el método de segmentación probabilística propuesto se presentan en la siguiente sección.

5. Resultados experimentales

En esta sección se valida la formulación del método de segmentación probabilística propuesto, poniendo a prueba su desempeño en la práctica mediante dos experimentos: el primero corresponde a la capacidad de segmentación al detectar la ruta $\mathbf{x}_{t,d}$; el segundo muestra la capacidad predictiva del método ante la ausencia de información en el sensor visual.

5.1. Experimento 1: Segmentación con información en el sensor visual

Para este experimento se realizaron 520 estimaciones utilizando la bolsa de datos de MTY a 15 cuadros por segundo. De acuerdo con el cálculo de las matrices de covarianza, \mathbf{R}_t presenta una mayor incertidumbre, teniendo una varianza 36,72 % mayor en el primer valor de la diagonal, 36,72 % en el segundo valor y 13,01 % mayor en el tercer valor con respecto a la matriz de covarianza \mathbf{Q}_t . Estas matrices determinan la incertidumbre, indicando el grado de confiabilidad del modelo.

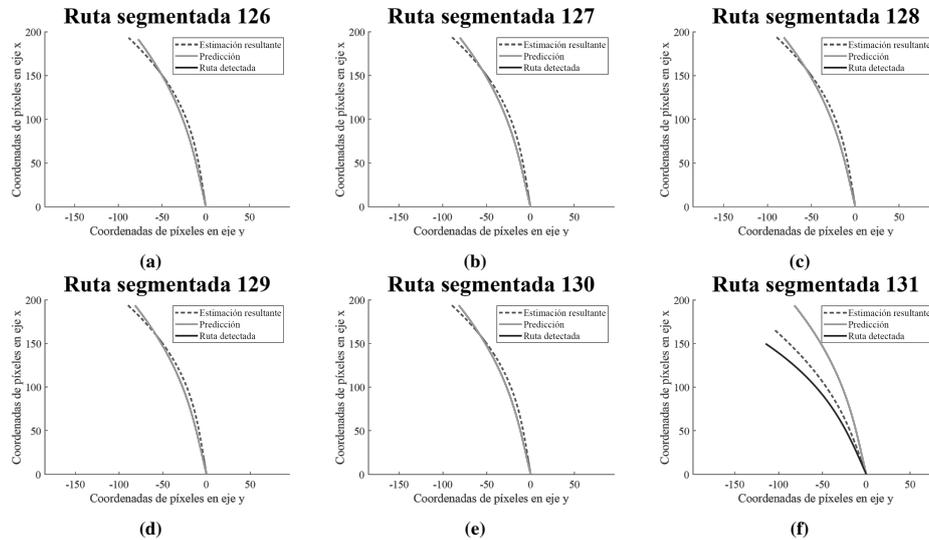


Fig. 10. Sección de curvas, las estimaciones en los cuadros de imagen 126 a 130 se realizan sin observación. (a), (b), (c), (d), (e) Muestran estimaciones realizadas sin observación. (f) Muestra la incorporación de una observación.

Por lo tanto, al tener una ruta detectada $x_{t,d}$ el modelo de medición tiene una menor incertidumbre, y la estimación resultante x_t tendrá una mayor similitud con respecto a la ruta detectada en comparación con la predicción. Lo anterior se muestra en la figura 8, en la cual se presentan tres estimaciones consecutivas. Realizando una comparativa numérica, la predicción en la figura 8a presenta una diferencia en la distancia promedio con respecto a la ruta detectada de 18,74 píxeles, a diferencia de la estimación resultante que tiene una diferencia de 7,14 píxeles con respecto a la ruta detectada.

Debido a que las bolsas de datos se obtuvieron a 15 cuadros por segundo, se espera que el cambio en las rutas segmentadas y el resultado del método de segmentación probabilística tenga una transición suave, inclusive ante la entrada de una observación atípica. Esto debido a la estrategia propuesta de actualización de covarianzas que regula dicha transición y que se muestra en la figura 9. Note que en la figura 9b la estimación resultante tiene una mayor confiabilidad con respecto a la predicción, debido a que la ruta detectada en este cuadro tiene una gran variación con respecto a la información previa.

5.2. Experimento 2: Segmentación con ausencia de información en el sensor visual

Este experimento tiene como finalidad demostrar la capacidad predictiva del método de segmentación probabilística ante la ausencia de información en el sensor visual. Para ello en este experimento se realizaron pruebas en secciones donde se presentan diferentes periodos de oclusión. La figura 10 muestra la variación en la estimación de rutas sin observación, en la que existe una diferencia significativa en el espacio de

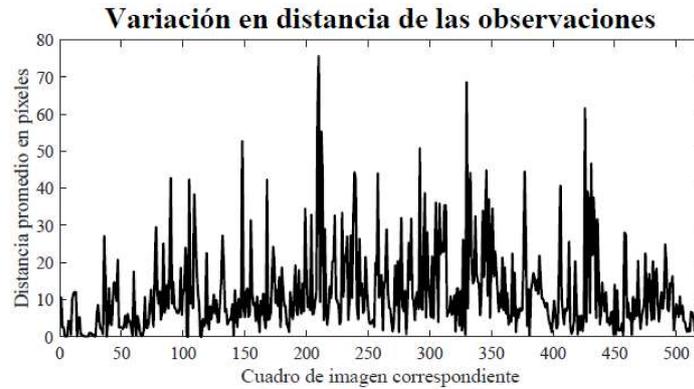


Fig. 11. Diferencia promedio en píxeles entre observaciones deterministas. La variación corresponde a un cambio significativo en el carril.

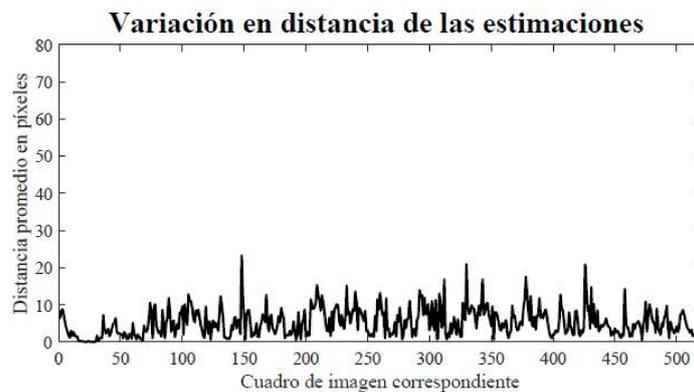


Fig. 12. Diferencia promedio en píxeles entre estimaciones. Se observa una transición suave entre estimaciones.

dimensión reducida de 43.1 píxeles con respecto a la predicción realizada, demostrando la capacidad predictiva del método propuesto. Con estos experimentos se ha mostrado que el método propuesto basado en el filtro de Kalman tiene como resultado una transición suave en las estimaciones, este comportamiento es favorable dado que no se presenta un cambio atípico en los carriles. Por otro lado, a nivel determinístico el modelo de observación presenta cambios significativos de un carril a otro.

En la figura 4.17 se observa este comportamiento para 520 observaciones en el cual se incluyen secciones de recta y curva. Note que existe una gran variación entre las observaciones, ya que se realiza una parametrización de los carriles con la información obtenida en el tiempo t , la cual puede no ser confiable debido principalmente a cambios de iluminación. El filtro de Kalman considera la información previa y regula la contribución de la nueva observación, dando como resultado un comportamiento suave de las transiciones como se muestra en la figura 12.

6. Conclusiones

En este artículo, propusimos un método de segmentación probabilística basado en el filtro de Kalman, el cual fue diseñado en un espacio de dimensión reducida mediante el análisis de componentes principales con el cual se validó un modelo lineal que permite la reproyección de rutas en dicho espacio. La importancia de este modelo radica en que proviene de datos de rutas recolectadas en tres diferentes competencias de vehículos autónomos, lo cual es de relevancia en el estudio de las variaciones principales de rutas para competencias de vehículos a escala. Hemos realizado una evaluación experimental que demuestra la eficiencia en el desempeño del método para estimaciones de forma y dimensión de carriles.

En los experimentos se hace visible la regulación del método para lograr una transición suave entre estimaciones, inclusive ante la entrada de una observación atípica realizada por el detector de carril determinístico, además el método muestra capacidades predictivas ante cuadros de imagen con oclusiones debido a la estrategia desarrollada para la actualización de covarianzas. Como trabajo futuro, planeamos incorporar información adicional de otros sensores en el modelo de medición y el uso de técnicas más sofisticadas para la parametrización de curvaturas.

Referencias

1. Assidiq, A., Khalifa, O., Islam, R., Khan, S.: Real time lane detection for autonomous vehicles. *International Conference on Computer and Communication Engineering*, pp. 82–88 (2008) doi: 10.1109/ICCCE.2008.4580573
2. Danescu, R., Nedeveschi, S.: Probabilistic lane tracking in difficult road scenarios using stereovision. *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 272–282 (2009) doi: 10.1109/tits.2009.2018328
3. Elnaz, B., Ali, G., Zohreh, A., Mansoor Zolghadri, J.: Supervised principal component analysis: Visualization, classification and regression on subspaces and submanifolds. *Pattern Recognition*, vol. 44, no. 7, pp. 1357–1371 (2011) doi: 10.1016/j.patcog.2010.12.015
4. Johnson, S. C.: Hierarchical clustering schemes. *Psychometrika*, vol. 32, no. 3, pp. 241–254 (1967) doi: 10.1007/BF02289588
5. Kuo Yu, C., Sheng Fuu, L.: Lane detection using color-based segmentation. *IEEE Proceedings. Intelligent Vehicles Symposium*, (2005) doi: 10.1109/ivs.2005.1505186
6. Li, M., Li, Y., Jiang, M.: Lane detection based on connection of various feature extraction methods. *Advances in Multimedia*, vol. 2018, pp. 1–13 (2018) doi: 10.1155/2018/8320207
7. Liu, G., Wörgötter, F., Markelić, I.: Combining statistical Hough transform and particle filter for robust lane detection and tracking. *IEEE Intelligent Vehicles Symposium*, pp. 993–997 (2010) doi: 10.1109/ivs.2010.5548021
8. Macias, J., Exposito, A.: Self-tuning of Kalman filters for harmonic computation. *IEEE Transactions on Power Delivery*, vol. 21, no. 1, pp. 501–503 (2006) doi: 10.1109/tpwrd.2005.860411
9. Meuter, M., Muller-Schneiders, S., Mika, A., Hold, S., Nunn, C., Anton, K.: A novel approach to lane detection and tracking. *12th International IEEE Conference on Intelligent Transportation Systems*, pp. 1–6 (2009) doi: 10.1109/itsc.2009.5309855
10. Odelson, B., Lutz, A., Rawlings, J.: The autocovariance least-squares method for estimating covariances: application to model-based control of chemical reactors. *IEEE Transactions on Control Systems Technology*, vol. 14, no. 3, pp. 532–540 (2006) doi: 10.1109/tcst.2005.860519

11. Thorpe, C., Herbert, M., Kanade, T., Shafer, S.: Toward autonomous driving: the CMU navlab I. Perception. IEEE Expert, vol. 6, no. 4, pp. 31–42 (1991) doi: 10.1109/64.85919
12. Thrun, S., Burgard, W., Fox, D.: Probabilistic robotics, pp. 16–39. MIT press (2005)
13. Truong, Q. B., Lee, B. R.: New lane detection algorithm for autonomous vehicles using computer vision. International Conference on Control, Automation and Systems, pp. 1208–1213 (2008) doi: 10.1109/iccas.2008.4694332
14. Yue, W., Eam Khwang, T., Dinggang, S.: Lane detection and tracking using B-Snake. Image and Vision Computing, vol. 22, no. 4, pp. 269–280 (2004) doi: 10.1016/j.imavis.2003.10.003
15. Zou, Q., Jiang, H., Dai, Q., Yue, Y., Chen, L., Wang, Q.: Robust lane detection from continuous driving scenes using deep neural networks. IEEE Transactions on Vehicular Technology, vol. 69, no. 1, pp. 41–54 (2020) doi: 10.1109/tvt.2019.2949603

Diseño automático de filtros analógicos pasivos mediante un algoritmo genético

Aurora Torres Soto¹, María Dolores Torres Soto²,
Eunice Ponce de León¹, Braulio Jesús Montoya Padilla¹

¹ Universidad Autónoma de Aguascalientes,
Departamento de Ciencias de la Computación,
México

² Universidad Autónoma de Aguascalientes,
Departamento de Sistemas de Información,
México

{atorres, mdtorres, eponce}@correo.uaa.mx
a1188428@edu.uaa.mx

Resumen. En el área de diseño de circuitos analógicos, es común trabajar sobre plantillas de topología preestablecidas, en las que los elementos de circuito y sus conexiones no se alteran durante el proceso de optimización de sus valores [1, 2]; sin embargo, aún con topologías prometedoras, es común que éstas no conduzcan a resultados con buen desempeño, de manera que el proceso de optimización se debe comenzar desde el principio [3]. En este trabajo se propone la integración de un grupo de mecanismos de modelación evolutiva de circuitos, con un algoritmo genético para abordar su diseño de manera que se evolucionen tanto la topología como los valores de los elementos. Los resultados obtenidos son muy prometedores, pues el costo de procesamiento es razonable, no se generan circuitos no simulables y el desempeño de los filtros es comparable al obtenido mediante técnicas clásicas; además de que los circuitos generados ya exhiben valores comerciales de elementos de circuito, lo que reduce la brecha entre el dispositivo diseñado y el implementado.

Palabras clave: Algoritmo genético, circuitos analógicos, diseño automático, filtros, simulación, evolución de topología.

Automatic Design of Passive Analog Filters Using a Genetic Algorithm

Abstract. In the analog circuit design area, it is common to work on pre-established topology templates, in which the circuit elements and their

connections are not altered during the optimization process of their values [1, 2]; however, even with promising topologies, it is common that they do not lead to results with good performance, so the optimization process must be started from the beginning [3]. In this paper, the integration of a group of mechanisms for the evolutionary modeling of circuits is proposed, with a genetic algorithm to address their design in such a way that both the topology and the values of the elements evolve. The results obtained are very promising, since the processing cost is reasonable, non-simulable circuits are not generated and the performance of the filters is comparable to those obtained using classical techniques; in addition, the generated circuits already exhibit commercial values of circuit elements, which reduces the gap between the designed device and the implemented.

Keywords: Genetic algorithm, analog circuits, automated design, filters, simulation, topology evolution.

1. Introducción

Con el enorme desarrollo que los sistemas digitales han alcanzado en las últimas dos décadas, es común pensar que los sistemas analógicos han sido completamente desplazados por éstos; sin embargo, la parte analógica de los sistemas mixtos no solamente es insustituible, sino que además consume el 90% del tiempo de diseño de estos sistemas [4]. En la actualidad existe una gran variedad de herramientas de asistencia al diseño de circuitos digitales; desafortunadamente, la curva de crecimiento de este tipo de herramientas para el dominio analógico ha sido siempre más lenta y menos adecuada; probablemente porque el diseño analógico es menos sistemático, además de ser una tarea que requiere de conocimiento especializado y una amplia experiencia [5].

El diseño de circuitos analógicos, frecuentemente descrito como “el arte de crear un dispositivo eléctrico que cumpla un grupo de especificaciones previamente establecidas”; es un proceso que consta de dos pasos: el establecimiento de la topología del circuito y la determinación de los valores de los elementos que lo forman[6]. Mientras que en el primer paso se establece cómo se conectarán unos elementos con otros e incluso el tipo de elementos de circuitos que serán conectados; en el segundo paso se establece la magnitud de cada elemento de circuito.

Teniendo en cuenta, la enorme importancia del diseño de circuitos analógicos en el proceso de elaboración de circuitos integrados, así como su reconocido nivel de complejidad; especialmente en el establecimiento de la topología, en este trabajo proponemos la integración de un grupo de mecanismos de modelación en la construcción y prueba de filtros analógicos pasivos, aplicando un algoritmo genético. El algoritmo deberá establecer tanto la topología como los valores de los elementos partiendo de un diseño “de hoja en blanco”.

En la siguiente sección se expone el problema de diseño de circuitos en la que se incluye una breve descripción de los filtros analógicos, pues el sistema propuesto se ha puesto a prueba en el diseño de un filtro analógico pasa-bajas. En la sección 3 se

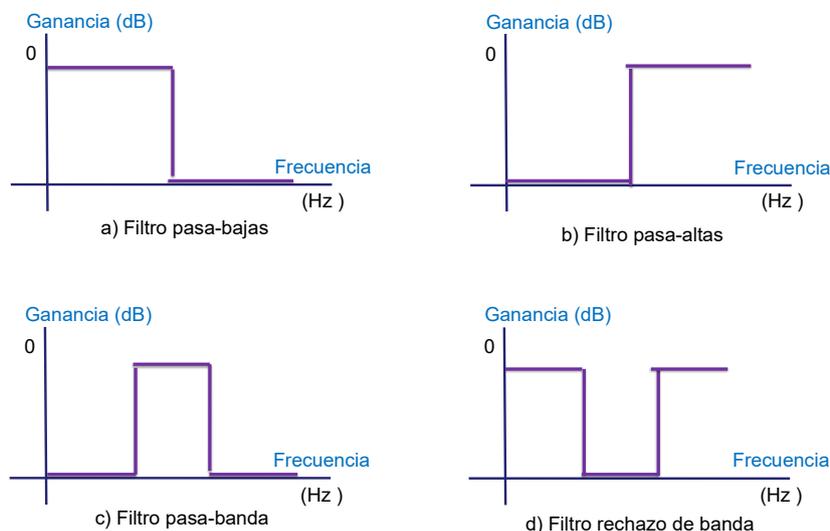


Fig. 1. Filtros ideales.

discuten brevemente los mecanismos de modelación y el algoritmo que los manipula. La sección 4 expone los experimentos realizados seguidos por los resultados y finalmente, en la sección 6 se discuten las conclusiones y el trabajo futuro.

2. Diseño de circuitos

El diseño de circuitos es el proceso que parte del establecimiento de un grupo de especificaciones de desempeño y termina con el dispositivo que exhibe el comportamiento deseado.

En función del tipo de señales que maneja un circuito, éste se puede clasificar como analógico, digital o mixto. Para este trabajo nos centraremos en los primeros. El área de los circuitos analógicos es extraordinariamente rica en variedad y versatilidad; sin embargo, todos comparten el hecho de que procesan señales analógicas o continuas; por esta razón, también son conocidos como sistemas de tiempo continuo. Este tipo de dispositivos tienen como objetivos clásicos: el filtrado, la amplificación y la fidelidad de las señales. Este trabajo se centra en el diseño de circuitos que realizan filtrado.

El problema que se afronta ha sido abordado empleando diferentes estrategias; inicialmente la tendencia fue abordar sus dos fases básicas (establecimiento de la topología y determinación de los valores de los elementos) de manera independiente; sin embargo, debido a que el establecimiento de la topología implica un problema mucho más retador que la optimización de los valores de elementos pre-colocados en un patrón; la optimización de los valores de un circuito fijo, fue la primera tarea abordada [1, 2] y en consecuencia, la primera en la que se emplearon técnicas de búsqueda no convencional [7–9].

Algunos de los primeros trabajos en los que se abordó el diseño desde la etapa de establecimiento de la topología se remontan a finales del siglo pasado [10-12]. Horrocks y Spittle [8] por ejemplo, abordaron el problema de la selección de los valores de los elementos de un filtro pasa-bajas activo usando solamente valores comunes de los elementos; idea que fue tomada posteriormente por Torres y sus colaboradores [13] para la elaboración de un mecanismo de representación que codifica tanto topología como dimensión de circuitos para ser evolucionados por herramientas evolutivas. Las herramientas de optimización y búsqueda heurística han sido protagonistas de numerosos esfuerzos por facilitar el diseño de circuitos durante los últimos veinte años.

Algunas de las metaheurísticas que han mostrado mayores beneficios en este campo son los algoritmos genéticos [6, 7, 14, 15] y la programación genética [12, 16]. Un filtro puede considerarse como un dispositivo selectivo a la frecuencia, que permite dar forma de una manera prescrita a la respuesta en magnitud o en fase [17]; es decir, es un circuito que bloquea las señales eléctricas de cierto rango de frecuencias, mientras permite el paso de aquellas con frecuencias en distinto rango [18].

Cuando un filtro se compone de elementos pasivos (resistencias, capacitores e inductancias), decimos que el filtro es pasivo y si posee elementos activos (transistores, amplificadores operacionales, entre otros); entonces es llamado filtro activo. Algunas de las distintas aplicaciones comunes de los filtros analógicos son: eliminación de ruido en señales de comunicaciones, detección de señales de radio y televisión, telefonía en DSL y audio, entre otras. De acuerdo con las bandas de frecuencia de las señales eléctricas que los filtros permiten pasar o que atenúan, éstos se clasifican en: pasa-altas, pasa-bajas, pasa-banda o rechazo de banda.

La figura 1 muestra el comportamiento ideal de estos circuitos. En esta figura se aprecia que la transición entre las bandas de paso (aquellas en las que la ganancia de voltaje es 0dB) y las de atenuación (aquellas en las que la ganancia es menor de 0dB), es instantánea; la señal de salida no presenta ruido y el grado de atenuación o paso de las señales es del 100% para los filtros ideales. Los filtros reales; sin embargo, se caracterizan por realizar las transiciones entre bandas más lentamente, además de que es muy común que las señales presenten ruido o rizo.

En la medida en la que un filtro se asemeja más al comportamiento ideal, es de mayor orden y por consiguiente de mejor calidad. Los filtros analógicos constituyen además un buen recurso de evaluación del desempeño de algoritmos, mecanismos y sistemas relacionados con el diseño automático de circuitos, pues éstos son los circuitos que los investigadores del área han utilizado con mayor frecuencia. De acuerdo con Goh y Li [7], el diseño de filtros es una disciplina bien comprendida; y teniendo presente que ha sido extensamente estudiada [19-23], su selección posibilita la comparación de los resultados de los mecanismos y algoritmos propuestos.

3. Descripción de la propuesta

Las técnicas de búsqueda y optimización heurísticas han mostrado ser muy eficientes en la solución de problemas complejos; sin embargo, exigen una modelación cuidadosa del fenómeno a estudiar. De acuerdo con Talbi [24], entre los aspectos comunes de las

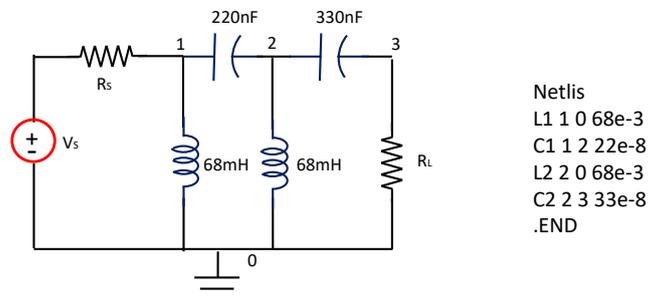


Fig. 2. Circuito eléctrico y su listado de conexiones.

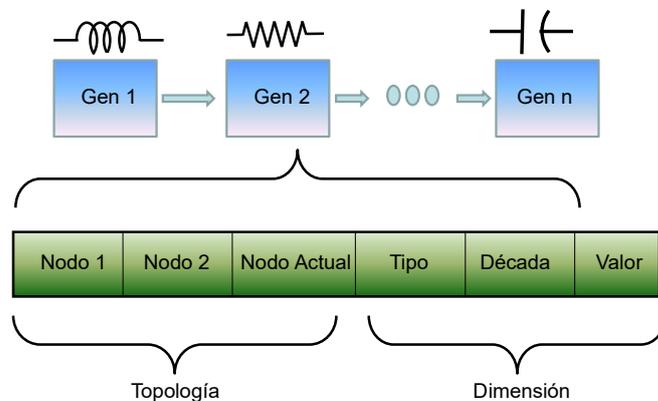


Fig. 3. Mecanismo de representación de circuitos.

metaheurísticas de mayor importancia están la representación de posibles soluciones y la función de adaptabilidad, que es quien conduce la búsqueda. A continuación, se describen tanto el mecanismo de representación, como el de evaluación que son empleados en esta propuesta.

3.1. Representación

En este trabajo, una solución al problema de diseño es una red eléctrica, constituida por elementos de circuito y nodos de conexión topológicamente correcta; es decir, que no debe contener conexiones a tierra desde ciertos nodos, ni tampoco dejar nodos desconectados. Adicionalmente, como se pretende que la herramienta sea capaz de establecer tanto su topología como los valores de los elementos, el mecanismo para su representación debe considerar la exploración de diferentes topologías y tamaños de los circuitos, así como una variedad suficiente de dispositivos comerciales. Finalmente, se pretende que los dispositivos generados por la herramienta sean simulados para comprobar su desempeño, así es que otra característica importante de la representación es su facilidad para la generación del “netlist” (palabra inglesa que significa “listado de conexiones”).

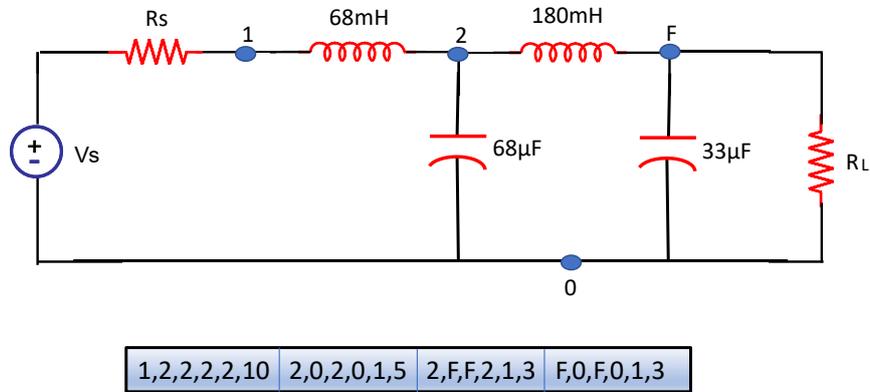


Fig. 4. Circuito y su codificación. Tomado de [13].

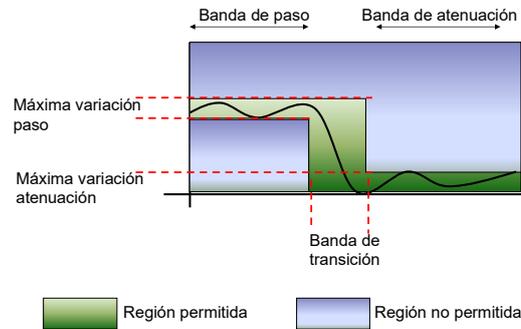


Fig. 5. Respuesta en frecuencia de un filtro pasa-bajas real.

Este listado de conexiones adicionalmente contiene instrucciones que se dictan al simulador para realizar un tipo de análisis. En la siguiente figura se muestran un circuito y su correspondiente netlis. Cabe hacer notar de la figura 2, que no se incluyen en la lista de conexiones la fuente de voltaje con su resistencia interna y la resistencia de carga.

Tomando como base la información contenida en las listas de conexiones y las otras consideraciones discutidas; en este trabajo se usa el mecanismo de representación descrito en [13].

Este mecanismo permite la construcción de circuitos de diferentes tamaños y topologías y adicionalmente usa dispositivos que son comercialmente comunes. La representación de un circuito (cromosoma) se almacena en una lista enlazada, cuya longitud corresponde con el número de elementos que tiene el circuito (genes).

Cada elemento de circuito se codifica en 6 campos que almacenan: los nodos de conexión y el tipo de elemento y su dimensión, como se muestra en la figura 3.

En relación con los valores de los elementos, se usan las series comerciales E6 para capacitores y E12 para inductancias y resistencias. El valor de los elementos se altera en función de la década (ver figura 3), que oscila entre $[10^{-5}, 10^{-9}]$ para capacitores, entre $[10^1, 10^6]$ para resistencias y entre $[10^{-1}, 10^{-6}]$ para inductancias; sin embargo, estos

Algoritmo Evolutivo General

1. Sea $G=0$ el contador de generaciones
 2. Crea e inicializa una población $P(0)$ n -dimensional
 3. Repite
 - Evalúa la aptitud $f(x)$, de cada individuo x , en la población $P(G)$
 - Produce la descendencia
 - Selecciona la población $P(G+1)$ de la nueva generación
 - Avanza a la nueva generación. $G=G+1$
 4. Hasta que la condición de paro sea cumplida
-

Fig. 6. Patrón general de un algoritmo evolutivo.

rangos pueden ser ampliados fácilmente. La figura 4 muestra un circuito y su codificación asociada.

De la figura anterior, cabe señalar que la codificación no incluye información de la fuente de alimentación V_S , de la resistencia de la fuente R_S y de la resistencia de carga R_L (pues esta parte es constante para todos los filtros); por lo que la codificación posee solamente 4 nodos (elementos de circuito) de 6 campos cada uno. Al lector interesado en profundizar sobre este mecanismo se le sugiere consultar [13].

4. Evaluación

Una vez que se ha resuelto el problema de creación de soluciones al problema que se aborda, es necesario establecer la forma como su calidad o desempeño será evaluada. Para este propósito las metaheurística emplean la función de adaptación o “fitness function”, que normalmente es un valor numérico que permite identificar las soluciones que mejor se adaptan al planteamiento del problema.

Si la herramienta de búsqueda realiza maximización, entonces en la medida que la solución es de mejor calidad se asociará con un valor mayor de esta función; y si la herramienta minimiza, entonces se espera que el valor de la función de adaptación sea menor a medida que mejora la calidad de la solución.

La función de adaptación es un elemento clave en el diseño de cualquier herramienta de búsqueda y/o optimización, pues es quien conducirá a la herramienta hacia zonas prometedoras del espacio de soluciones [24]; sin embargo, se debe tener presente que es también una de las partes más consumidoras de tiempo y recursos computacionales [25]. Una de las razones por las que el caso de diseño de circuitos analógicos no es una excepción a esta regla, se debe a que, para designar un valor de adaptabilidad, todos los individuos (circuitos) generados deben ser simulados.

En este trabajo se emplea SPICE (siglas de “*Simulation Program with Integrated Circuits Emphasis*”)[26], que actualmente es el software de simulación de circuitos de uso más extendido. Para definir la función de adaptabilidad, es necesario establecer los

parámetros que ésta maneja. Para ello nos basaremos en la respuesta en frecuencia del filtro pasa-bajas de la figura 5.

En esta figura se presentan los parámetros que deben ser considerados cuando se establecen las especificaciones de diseño de un filtro que tendrá un comportamiento real; en el eje de las abscisas se bosqueja a la frecuencia y en el eje de las ordenadas se tienen niveles de voltaje. A diferencia del filtro ideal (como los mostrados en la figura 1).

Un filtro real no realiza la transición entre bandas de forma instantánea; sino que lo hace de forma paulatina, consumiendo un rango de frecuencias conocido como “*banda de transición*”. El ancho de esta banda es un factor que normalmente está asociado con la complejidad y calidad del filtro. Si esta banda es estrecha, el filtro es mejor, aunque normalmente se constituye de un mayor número de elementos de circuito; y en la medida en que esta banda se vuelve más ancha, el filtro es menos complejo, aunque su desempeño sea menor.

También es común que exista cierta variación en la ganancia de voltaje tanto en la banda de paso, como en la de atenuación (rizado). Esta variación generalmente se acentúa en la medida que el circuito se vuelve más complejo. En la gráfica de la figura 5 se representa esta característica como “*Máxima variación paso y máxima variación atenuación*”. Cuando un filtro sobrepasa estos límites, la función de adaptabilidad debe verse afectada negativamente.

De esta forma, la región denotada como “*región permitida*” establece los límites aceptables para la presencia de señal eléctrica mientras que la “*región no permitida*” representa las zonas en las que la presencia de señal debe ser castigada. La función de adaptación diseñada y probada en este documento se encuentra en la ecuación 1:

$$f = \frac{1}{1 + \sqrt{\frac{\sum_{i=1}^N (\lambda \cdot \text{Error})^2}{N - 1}}}, \quad (1)$$

donde:

N: es el número de puntos de frecuencia para los que se realiza la simulación.

λ : es el valor del castigo que se deberá imponer en función de la frecuencia y la amplitud del voltaje de salida del filtro.

Error: Es la diferencia entre la respuesta en voltaje ideal y la que presenta el filtro diseñado.

La función de la ecuación 1 esta normalizada de manera que el valor para un filtro ideal sería de 1. Así, valores cercanos al 100% describen filtros de muy buena calidad.

5. Algoritmo GA-AF

La evolución natural puede ser vista como un proceso de optimización en el que el objetivo es mejorar la habilidad de organismos o sistemas para sobrevivir en un entorno dinámico y competitivo [27]. Debido al éxito que ha tenido la evolución en la mejora de las especies, la comunidad científica empezó a adoptar algunos de sus mecanismos en la solución de problemas complejos en la forma de lo que conocemos como

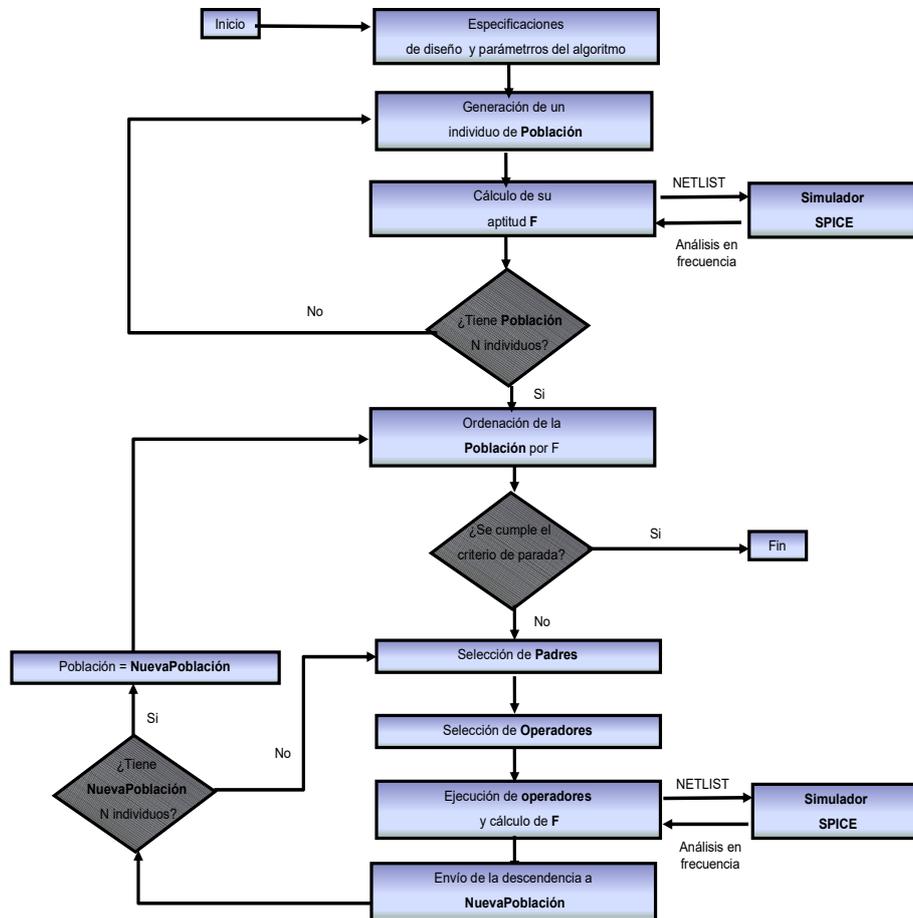


Fig. 7. Diagrama de flujo del AG-GA.

computación evolutiva. En la actualidad existe una amplia variedad de algoritmo evolutivos, sin embargo, todos tienen algunos pasos en común, al grado de que se puede establecer un algoritmo genérico con ellos.

La figura 6 muestra el esquema general de un algoritmo evolutivo. Entre algunos de los algoritmos más populares de la computación evolutiva podemos mencionar al algoritmo genético, cuyos principales precursores fueron John Holland [27] y David Goldberg [28]; la programación genética, inicialmente promovida por John Koza [16]; la programación evolutiva, difundida por Fogel [29]; las estrategias evolutivas, promovidas por Schwefel [30], los algoritmos de estimación de la distribución, descritos detalladamente por Pedro Larrañaga [31] y la evolución diferencial, discutida ampliamente por Vitaliy [32].

Sin embargo, los algoritmos evolutivos por excelencia son los algoritmos genéticos que inicialmente fueron llamado “planes reproductivos” por John Holland; pues fueron de los primeros emuladores de la evolución genética que produjeron resultados

Tabla 1. Especificaciones de diseño, simulación y evaluación.

Especificación	Descripción
Frecuencia de paso (F_p)	Establece el fin de la banda de paso cuando se trata de un filtro pasa-bajas (inicio de la banda de transición) y determina el inicio de la banda de paso (fin de la banda de transición) cuando se trata de un filtro pasa-altas.
Frecuencia de atenuación (F_a)	Establece el inicio de la banda de atenuación (fin de la banda de transición) cuando se trata de un filtro pasa-bajas y determina el fin de la banda de atenuación (inicio de la banda de transición) cuando se trata de un filtro pasa-altas.
Tipo de filtro	Establece el tipo que será diseñado. 0: Pasa-bajas. 1: Pasa-altas.
Objetivo de paso	Establece el valor de voltaje que debería exhibir el filtro ideal en la banda de paso.
Objetivo de atenuación	Establece el valor de voltaje que debería exhibir el filtro ideal en la banda de atenuación.
Error máximo	Establece el valor de la máxima variación de voltaje permitido tanto en la banda de paso como en la de atenuación.
Voltaje de la fuente (V_s)	Determina el valor de voltaje de corriente alterna que será aplicado al circuito para su simulación.
Resistencia de la fuente (R_s)	Es el valor de resistencia asociado con la fuente de alimentación.
Frecuencia inicial (f_i)	Frecuencia en la que se debe iniciar la simulación
Frecuencia final (f_f)	Frecuencia en la que debe terminar la simulación
Castigo (λ)	Es el valor del castigo que será aplicado para las frecuencias en las que el filtro tiene señales en la región no permitida.

prácticos. En 1989, cuando Goldberg publicó su libro [28], ya citaba más de 70 aplicaciones exitosas de este paradigma, que con el paso del tiempo sigue ganando popularidad. En este trabajo, se emplea un algoritmo genético que está basado en el algoritmo genético simple [28]. El diagrama de la figura 7 refleja el flujo de pasos que contempla el algoritmo GA-AF, cuyo propósito es diseñar automáticamente un filtro analógico pasivo que cumpla un grupo de requerimientos establecidos.

En la figura 7 se aprecia que el algoritmo presenta los pasos generales del conocido como SGA y adicionalmente, procesa la información de sus individuos y construye automáticamente el “*netlist*” con la descripción del circuito diseñado y la definición del tipo de análisis que debe ser ejecutado (especificaciones de diseño). Una vez que el simulador “*SPICE*” tiene el resultado, éste es procesado por el algoritmo para el cálculo de su función de adaptación “*F*”.

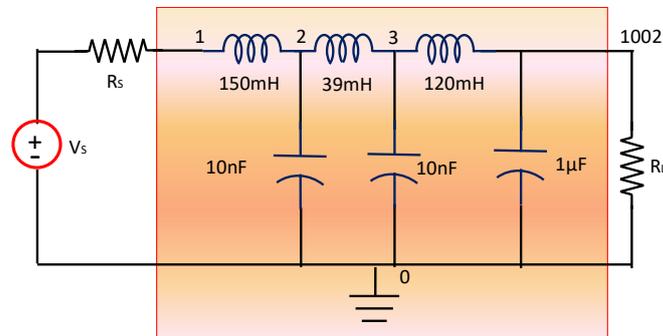


Fig. 8. Filtro pasa-bajas diseñado

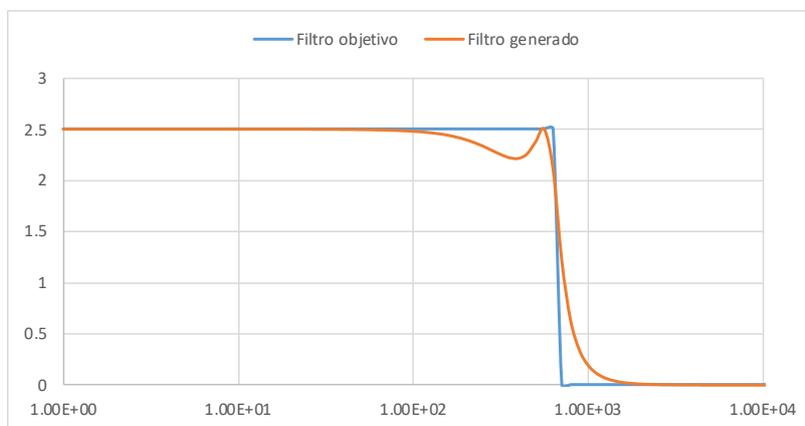


Fig. 9. Respuesta en frecuencia del filtro pasa-bajas.

Los parámetros del algoritmo son aspectos que definen exclusivamente la forma como el algoritmo genético debe ser ejecutado; tales como tamaño de la población, número de generaciones, probabilidad de cruzamiento y probabilidad de mutación. El criterio de paro que se está empleando inicialmente es un número máximo de generaciones.

En la tabla 1 se presentan los requerimientos de diseño de un filtro pasa-bajas, de acuerdo con las características mencionadas en la figura 5; las especificaciones necesarias para establecer el tipo de simulación que se debe realizar, y las especificaciones para la evaluación.

6. Experimentación

Una vez realizada la integración de todos los elementos que constituyen al algoritmo genético para el diseño de filtros analógicos “GA-AF”, se estableció un diseño factorial para el ajuste de los parámetros del algoritmo durante el proceso de diseño de un filtro pasa-bajas. Los parámetros manipulados y sus diferentes niveles se muestran en la tabla

Tabla 2. Diseño factorial.

Parámetro	Valores probados
Tamaño de la población	30, 50, 80, 100
Número de Generaciones	10, 15, 20, 25

Tabla 3. Resultados promedio.

Experimento Poblac_Gener	Evaluación	Tiempo	Número de elementos	Orden
30_5	0.72637	30.252499	4	1
30_15	0.773651	72.071495	7	1
30_20	0.754078	98.592506	5	1
30_25	0.825051	120.401405	4	2
50_5	0.719181	50.686996	4	1
50_15	0.785788	131.832794	5	2
50_20	0.789104	169.884979	5	1
50_25	0.819447	199.445297	4	2
80_5	0.751592	80.316002	4	1
80_15	0.844826	195.389313	4	2
80_20	0.822482	283.444275	5	1
80_25	0.828484	334.687836	5	2
100_5	0.803453	103.451096	4	2
100_15	0.839116	270.96344	4	2
100_20	0.859822	341.658234	6	2
100_25	0.855491	416.877502	5	2

2. Se realizaron 10 réplicas con cada combinación de parámetros, y se realizó un análisis de las variables de respuesta “Evaluación”, “Tiempo”, “Número de elementos”, “Orden”.

Los valores de probabilidad de cruzamiento “Pc” y probabilidad de mutación “Pm” se fijaron empíricamente en 0.9 y 0.3 respectivamente. Las especificaciones del objetivo de diseño son: filtro pasa-bajas con una banda de transición comprendida entre 700 y 750 Hz, con una variación máxima de voltaje en las bandas de paso y atenuación de 0.8V; se esta empleando una fuente de voltaje de corriente alterna con una resistencia interna de 500Ω y se supone que la carga tiene un valor de 500Ω también.

De acuerdo con los valores mostrados en la tabla 2, se realizó un total de 160 ejecuciones. La tabla 3 muestra los resultados promedio de las 10 réplicas para cada

combinación de los parámetros de entrada de este filtro. De la tabla anterior se aprecia que aparentemente el algoritmo exhibe su mejor desempeño con poblaciones de 100 individuos y 20 generaciones.

Luego de realizar un análisis a los resultados de los experimentos, se comprobó estadísticamente que no existe diferencia significativa entre las ejecuciones del algoritmo empleando 80 individuos y evolucionando durante 15 generaciones. Este análisis se realizó mediante la prueba de comparación de grupos independientes de Kruskal-Wallis.

7. Resultados

Para el diseño del filtro descrito en la sección 4, el GA-AF encontró el circuito que se muestra en la figura 8. Como se aprecia, este circuito es de aspecto comparable a aquellos diseñados por ingenieros expertos mediante el procedimiento de Chebyshev, y, sin embargo, fue elaborado sin fijar ni la topología ni los elementos de circuito, pues partió de una perspectiva de hoja en blanco. Con el objeto de validar que el diseño arrojado por el algoritmo efectivamente cumple con las especificaciones establecidas, en la figura 9 se presenta su respuesta bajo diferentes condiciones de frecuencia (línea roja) comparada con la del filtro ideal (línea azul).

Cabe mencionar que este filtro tiene un grado de atenuación de casi 24dB por octava, lo que describe a un filtro de orden 3. En esta figura se aprecia que el filtro ideal presenta la transición en 720 Hz aproximadamente, mientras que el diseñado sigue ese comportamiento general. A frecuencias bajas, el filtro permite el paso de la señal de voltaje (debajo de 700Hz) y por encima de 750Hz, la señal de voltaje se atenúa.

8. Conclusiones y trabajo futuro

Para este trabajo el objetivo principal fue la creación de circuitos que exhibieran el comportamiento dado por las especificaciones de diseño; razón por la cual, la función objetivo que conduce al algoritmo se basa exclusivamente en la respuesta de los circuitos a un análisis de frecuencia realizada mediante simulación.

El resultado obtenido exhibe un comportamiento de alrededor de tercer orden, lo que implica que tiene una banda de transición muy estrecha y, por consiguiente, posee un muy buen desempeño. El número total de circuitos que no fueron simulables se redujo a cero aún cuando el tamaño de los circuitos sugiere que no faltó diversidad en la búsqueda, lo que nos alienta a declarar que la clase de circuitos válidos se mantuvo cerrada durante el proceso de evolución.

Este hecho representa una enorme contribución en este campo, pues uno de los principales problemas reportados en el área, es el enorme costo computacional y tiempo que se deben invertir en la composición de redes eléctricas que no producen resultados como lo reporta Koza [12]. Los mecanismos de modelación empleados son simples y efectivos para mantener la clase de circuitos válidos cerrada.

El tiempo promedio que le lleva al GA-AF encontrar un circuito que se adapte a las especificaciones es de 5 minutos aproximadamente. Finalmente, el algoritmo propuesto

no requiere de grandes poblaciones ni un número considerable de generaciones para encontrar circuitos con una función de adaptación por encima de 0.85; los experimentos realizados sugieren que con poblaciones de 80 individuos evolucionando durante 15 generaciones son suficientes.

Como trabajo futuro, se pretende integrar un enfoque multiobjetivo en el proceso de diseño de circuitos, pues claramente, este es un problema en el que se deben contemplar al menos, complejidad del circuito, tiempo de diseño y desempeño. Adicionalmente se están haciendo las modificaciones necesarias a la función de adaptabilidad para trabajar con otros tipos de filtros.

Referencias

1. El Dor, A., Fakhfakh, M., Siarry, P.: Multiobjective differential evolution algorithm using crowding distance for the optimal design of analog circuits. *Journal of Electrical Systems*, vol. 12, pp. 612–622 (2016)
2. Sanabria-Borbón, A. C., Tlelo-Cuautle, E.: Sizing analog integrated circuits by combining gm/ID technique and evolutionary algorithms. In: *57th International Midwest Symposium on Circuits and Systems*, pp. 234–237 (2014)
3. Shi, G.: Toward automated reasoning for analog IC design by symbolic computation – A survey. *Integration, the VLSI Journal*, vol. 60, pp. 117–131 (2018) doi: 10.1016/j.vlsi.2017.08.005.
4. Ochotta, E. S., Rutenbar, R. A., Carley, L. R.: Synthesis of high-performance analog circuits in ASTRX/OBLX. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 15, pp. 273–294 (1996) doi: 10.1109/43.489099.
5. Das, A., Vemuri, R.: A graph grammar based approach to automated multi-objective analog circuit design. In: *Proceedings of the Conference: Design, Automation & Test in Europe*, pp 700–705 (2009) doi: 10.1109/DATE.2009.5090755.
6. Torres-Soto, A.: *Metaheurísticas evolutivas en el diseño de circuitos analógicos*. Benemérita Universidad Autónoma de Aguascalientes (2010)
7. Goh, C., Li, Y.: GA automated design and synthesis of analog circuits with practical constraints. *Proceedings of the 2001 Congress on Evolutionary Computation* (2001)
8. Horrocks, D., Spittle, M.: Component value selection for active filters using genetic algorithms. *Proceedings of IEE Workshop on Natural Algorithms in Signal Processing*, vol. 1, pp. 131–136 (1993)
9. Horrocks, D. H., Khalifa, Y. M. A.: Genetically derived filter circuits using preferred value components. *IEE Colloquium on Analogue Signal Processing*, (1994)
10. Zebulum, R. S., Pacheco, M. A., Vellasco, M.: Comparison of different evolutionary methodologies applied to electronic filter design. In: *IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence*, pp. 434–439, (1998) doi: 10.1109/ICEC.1998.699812.
11. Lohn, J. D., Colombano, S. P.: Automated analog circuit synthesis using a linear representation. In: Sipper, M., Mange, D., Pérez-Urbe, A. (eds) *Evolvable Systems: From Biology to Hardware*. ICES 1998. *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, vol. 1478, pp. p 125–133, (1998) doi: 10.1007/BFb0057614.

12. Koza, J. R.: Genetic programming as a means for programming computers by natural selection. *Statistics and Computing*, vol. 4, pp. 87–112 (1994). doi: 10.1007/BF00175355.
13. Torres, A., Torres, M., Ponce de León, E.: Mecanismo de representación para la evolución automática de circuitos analógicos. *Research in Computing Science*, vol. 149, pp. 253–265 (2020)
14. Lohn, J. D., Colombano, S. P.: A circuit representation technique for automated circuit design. *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 3, pp. 205–219, (1999). doi:10.1109/4235.788491.
15. Torres, A., Ponce de León, E. E., Torres, M. D., Padilla, F.: Comparison of two evolvable systems in the automated analog circuit synthesis. In: *Mexican International Conference on Artificial Intelligence*, pp. 3–8, (2009) doi: 10.1109/MICAI.2009.25.
16. Koza, J. R., Poli, R.: Genetic Programming. *Search Methodologies*. In: Burke, E.K., Kendall, G. (eds) *Search Methodologies*. Springer, Boston, MA. pp. 127–164. doi: 10.1007/0-387-28356-0_5.
17. Ambardar, A., Urbina, E., Nagore, G., Hernández, L.: *Procesamiento de señales analógicas y digitales*. 2^{da} Ed. Thomson Learning, México (2009)
18. Johnson, C. D.: *Process control instrumentation technology*. Publication History, (1999)
19. He, J., Zou, K., Liu, M.: Section-representation scheme for evolutionary analog filter synthesis and fault tolerance design. In: *Third International Workshop on Advanced Computational Intelligence*, pp 265–270 (2010) doi: 10.1109/IWACI.2010.5585181.
20. Kim, K. J., Wong, A., Lipson, H.: Automated synthesis of resilient and tamper-evident analog circuits without a single point of failure. *Genetic Programming and Evolvable Machines*, vol. 11, pp. 35–59 (2010) doi: 10.1007/s10710-009-9085-2.
21. Ando, S., Iba, H.: Analog circuit design with a variable length chromosome. In: *Proceedings of the 2000 Congress on Evolutionary Computation*, vol. 2, pp 994–1001 (2000) doi: 10.1109/CEC.2000.870754.
22. Sapargaliyev, Y., Kalganova, T.: Constrained and unconstrained evolution of “LCR” low-pass filters with oscillating length representation. In: *IEEE International Conference on Evolutionary Computation*, pp. 1529–1536, (2006) doi: 10.1109/CEC.2006.1688490.
23. Hu, J., Zhong, X., Goodman, E. D.: Open-ended robust design of analog filters using genetic programming. In: *GECCO '05: Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pp. 619–1626 (2005) doi: 10.1145/1068009.1068283.
24. Talbi, E. G.: *Metaheuristics: From Design to Implementation*. Interfaces (Providence), vol. 42, pp. 414–415.
25. McPhee, N. F., Poli, R., Langdon, W. B.: *A Field Guide to Genetic Programming* (2008)
26. Fernández, H., Franco, Z. E.: Usando el PSPICE orientado al análisis de los circuitos del controlador booster instalado en CVG Venalum. *Universidad, Ciencia y Tecnología*, vol. 13, no. 51, pp. 085–095 (2009)
27. Holland, J. H.: *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. MIT Press, Cambridge, MA, USA (1992)
28. Goldberg, D. E., Holland, J. H.: *Genetic Algorithms and Machine Learning*. *Machine Learning*, vol. 3, pp. 95–99 (1988) doi: 10.1023/A:1022602019183.
29. Lawrence, J. F.: *On the Organization of Intellect*. University of California, Los Angeles, Engineering (1964)

Aurora Torres Soto, María Dolores Torres Soto, Eunice Ponce de León, et al.

30. Schwefel, H. P.: *Evolutionsstrategie und numerische Optimierung* (1975)
31. Larrañaga, P., Lozano, J. A.: *Estimation of Distribution Algorithms*. Springer US, Boston, MA (2002)
32. Feoktistov, V.: *Differential Evolution*. Springer US, Boston, MA (2006)

Desarrollo de un módulo para la prevención de la hipertensión usando el paradigma IoT y aprendizaje automático

Juan Antonio Alor-Hernandez¹, Nancy Aracely Cruz Ramos¹,
Giner Alor-Hernández¹, José Luis Sánchez-Cervantes²,
Lisbeth Rodríguez-Mazahua¹

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

{msc.jalor, dci.ncruz}@ito-depi.edu.mx,
{giner.ah, jose.sc, lisbeth.rm}@orizaba.tecnm.mx

Resumen. La hipertensión, también conocida como asesino silencioso es un síndrome clínico cuya principal característica es el aumento de la presión arterial, a su vez es una de las principales causas de enfermedades cardiovasculares y cerebrovasculares. En este trabajo se describe el desarrollo de un módulo para la prevención e identificación de hipertensión mediante técnicas de Aprendizaje Automático e Internet de las cosas. El módulo obtiene información mediante el uso de una pulsera inteligente que se encarga de monitorizar la presión arterial de cada individuo para determinar si una persona es hipertensa o normotensa. El principal beneficio del módulo, es que realiza una monitorización en tiempo real de la presión arterial permitiendo identificar, prevenir y controlar la presión arterial alta, mediante el uso de recomendaciones médicas apropiadas, con el propósito de lograr disminuir las muertes por esta enfermedad.

Palabras clave: Aprendizaje automático, hipertensión, internet de las cosas, inteligencia artificial, wearable.

Development of a Module for the Prevention of Hypertension Using the IoT Paradigm and Machine Learning

Abstract. Hypertension, also known as the silent killer, is a clinical syndrome whose main characteristic is the increased blood pressure, which in turn is one of the main causes of cardiovascular and cerebrovascular diseases. This paper describes the development of a module for the prevention and identification of hypertension using Machine Learning and Internet of Things techniques. The module obtains information using a smart bracelet that is responsible for monitoring the blood pressure of each being to determine if a person is

hypertensive or normotensive. The main benefit of the module is that it performs real-time monitoring of blood pressure, allowing the identification, prevention, and control of high blood pressure, using appropriate medical recommendations, with the purpose of reducing deaths from this disease.

Keywords: Machine learning, hypertension, internet of things, artificial intelligence, wearable.

1. Introducción

La hipertensión es un síndrome clínico cuya principal característica es el aumento de la presión arterial sistémica y se considera como la enfermedad crónica más común, así como el factor de riesgo más importante de enfermedades cardiovasculares y cerebrovasculares.

La hipertensión es la principal causa prevenible de muerte prematura en todo el mundo. Según los datos de la Organización Mundial de la Salud (OMS) la hipertensión afecta más de 250 millones de personas adultas en todo el mundo [1]. Muchos factores conducen a la hipertensión, incluido el estrés occipital y la tensión laboral.

Uno de los principales problemas relacionados con los tratamientos para las condiciones de hipertensión arterial es la detección tardía de personas aparentemente sanas. Algunos estudios demuestran que, entre los individuos con hipertensión, más del 35% desconocían su condición [2]. Por lo anterior, se hace necesario el desarrollo de mecanismos que permitan reconocer la hipertensión lo antes posible, cuyo principal beneficio sea ayudar a las personas a monitorizar en tiempo real la presión arterial y así prevenir, identificar y controlar los niveles de hipertensión.

De esta manera el paciente llevará un tratamiento adecuado y así evitará que se presenten daños irreversibles en el cuerpo humano. Recientemente, la convergencia tecnológica en el cuidado de la salud se ha vuelto relevante porque es una alternativa prometedora para el tratamiento de pacientes con enfermedades crónicas.

Puntualmente, el Internet de las cosas (IoT) logra esquematizar un panorama de tecnologías, que en este trabajo se enfocan al autocuidado y promoción de los hábitos saludables.

La contribución de este trabajo es el desarrollo de un módulo que realiza el proceso de prevención e identificación de hipertensión mediante técnicas de Aprendizaje Automático e Internet de las Cosas (IoT), esto con el fin de lograr disminuir las muertes por esta enfermedad.

Este trabajo está estructurado de la siguiente manera: la sección 2 presenta los trabajos relacionados con la hipertensión y tecnologías basadas en IoT; la sección 3 describe la arquitectura utilizada para el desarrollo del módulo basado en el paradigma IoT; la sección 4 presenta las funcionalidades del módulo; finalmente, la sección 5 presenta las conclusiones y el trabajo futuro.

2. Trabajos relacionados

Sazonov *et al.* [3] desarrollaron un control remoto de sistema de cuidado de la salud conformado por un sensor de bioseñal y un microcontrolador. Se concluyó, que muchos de los problemas se evitan si se mantiene un control y compromiso con el cuidado.

Janjua *et al.* [4] propuso un sistema de monitorización de la PA el cual se colocó en el pecho y midió la rigidez arterial calculando el tiempo de tránsito del pulso. El análisis de Bland-Altman mostró que el algoritmo del dispositivo es compatible con el grado C de los estándares de la British Hypertensive Society para la estimación de la presión arterial sistólica. Ni *et al.* [5] desarrolló un cinturón con un sistema de monitorización continua para leer datos de ECG.

Con el 93.33% de precisión se pudo diferenciar a los pacientes que tenían hipertensión de los que eran sanos. Afeni *et al.* [6] desarrolló un sistema predictivo de hipertensión para aquellos pacientes en tratamiento, utilizando el clasificador Naive Bayes. El clasificador Naive Bayes mostró el mejor desempeño para el diagnóstico de hipertensión. Ruíz *et al.* [7] propuso un nuevo modelo basado en BPM, IoT, principios de teledetección y usabilidad.

Se obtuvo la información del paciente (evolución y tratamiento de su enfermedad). Como resultado, se presentó un modelo flexible que logró ayudar a los médicos a proporcionar un tratamiento preciso en cada momento. Irizarry *et al.* [8] realizó un estudio del uso de la aplicación *MyBP*, el cual consto de tres fases. Se concluyó que la eficacia y la viabilidad de *MyBP* como complemento de la medicina estándar dependen del compromiso del paciente.

Thangada *et al.* [9] realizó estudios que demostraron cómo las aplicaciones para teléfonos inteligentes tratan de mejorar el manejo de la hipertensión. Comparó los resultados de la PA entre pacientes que usaban un sistema de monitorización de PA en el hogar contra un grupo que llevaban el control desde una aplicación. Encontraron que el 72,3% de los pacientes hipertensos que usaban la aplicación alcanzaban una PA <140/90 mmHg>, mientras que el 39,9% del otro grupo alcanzaron una PA <140/90 (0,0001)>.

Singh *et al.* [10] propuso un enfoque de aprendizaje híbrido para la generación de reglas de clasificación para diagnosticar la hipertensión entre individuos diabéticos. Se realizó un entrenamiento con el modelo SVM y se extrajeron reglas de los aprendices basados en árboles C4.5, RF, XGBoost. El modelo obtuvo un 74,7% de precisión y un valor del área bajo la curva (AUC) de 0,810 para la clase positiva. Carvajal *et al.* [11] diseñaron un clasificador GT2 FL para medir los niveles del pulso, posteriormente desarrollaron un hardware del clasificador en el Jetson Nano Development Board.

Se concluyó que el clasificador GT2 FL propuesto logra una mejor clasificación en comparación con los clasificadores difusos de intervalo tipo 2 y tipo 1. En continuación con el trabajo anterior Guzmán *et al.* [12] desarrollaron un clasificador de niveles de presión arterial. Se obtuvieron 45 muestras de presión sistólica y 45 muestras de presión diastólica. Posteriormente se realizó el entrenamiento de la red neuronal, los resultados arrojaron que el sistema tipo 2 realiza una mejor clasificación (99,408%) que el sistema tipo 1 (98%).

Kitt *et al.* [13] analizaron la aplicación *Cardiogram* (reloj Apple®). Estos dispositivos portátiles se probaron en niños, mujeres embarazadas y en personas con fibrilación auricular, se concluyó que los dispositivos portátiles no son confiables en este tipo de población. Mankar *et al.* [14] analizaron diferentes técnicas de aprendizaje automático.

Se recopiló datos de 23,095, se realizó la evolución con seis técnicas de clasificación diferentes para predecir el resultado de la hipertensión. El modelo Random Tree Forest superó significativamente a todos los demás. Wu *et al.* [15] generó un algoritmo ML para predecir los resultados en pacientes jóvenes con hipertensión. Se obtuvieron los siguientes resultados 0,757 (IC del 95%, 0,660 a 0,854) para el modelo ML, para el modelo de regresión de Cox y el de puntuación de riesgo de Framingham recalibrado se obtuvo 0,723 (IC del 95%, 0,636 a 0,810) y 0,529 (IC del 95%, 0,403 0,655).

Kario *et al.* [16] realizó la implementación de dispositivos portátiles. Validaron dispositivos de muñeca tipo reloj para la medición de la PA, los dispositivos HEM6410T-ZM y HEM-6410T-ZL cumplían con los criterios de validación cuando se usaban en posición sentada con la muñeca al nivel del corazón. Welykholowa *et al.* [17] desarrolló un dispositivo no invasivo para la medición la PA. Se analizaron 73 artículos, la mayoría de los estudios utilizaron ECG. Varios prototipos introdujeron modalidades basadas en fotoplestismografía para mejorar la evaluación de la hipertensión

Una de estas modalidades es la recolección simultánea de ECG con PPG para estimar PAT. Se concluyó que las tecnologías actuales enfrentan limitaciones para incorporar con éxito PPG y ECG en un solo dispositivo portátil. Tsoi *et al.* [18] propuso la predicción de los valores de PA a partir de grabaciones de voz de la vocal a. Se entrenó una CNN con dos grupos para predecir los valores de PA.

Se obtuvo una de 93,7% y el RMSE de 0,236. Se concluyó que se requieren más acciones antes del lanzamiento oficial de aplicaciones de IA para prácticas clínicas. Zhang *et al.* [19] realizó un estudio para evaluar el uso de una pulsera de autocontrol durante las 24 horas. Los resultados arrojaron que no existieron diferencias entre los usuarios y no usuarios de dispositivos en relación con las mediciones de la salud cardiovascular.

3. Arquitectura del módulo para la prevención de la hipertensión

En este trabajo se desarrolló una aplicación web que realiza el proceso de identificación de la hipertensión mediante dispositivos *wearables*, Aprendizaje Automático e IoT. La aplicación web aplica algoritmos de clasificación dentro del aprendizaje automático, los cuales permiten predecir una o más variables discretas, basándose en otros atributos de un conjunto de datos.

El esquema general de la aplicación comienza con la extracción de información relevante mediante la pulsera inteligente, así mismo almacena datos sensibles que son de suma importancia para el usuario, se implementa un mecanismo automático para el almacenamiento de dichos datos como son la frecuencia cardíaca, presión arterial, pulso del volumen sanguíneo y temperatura en la piel.

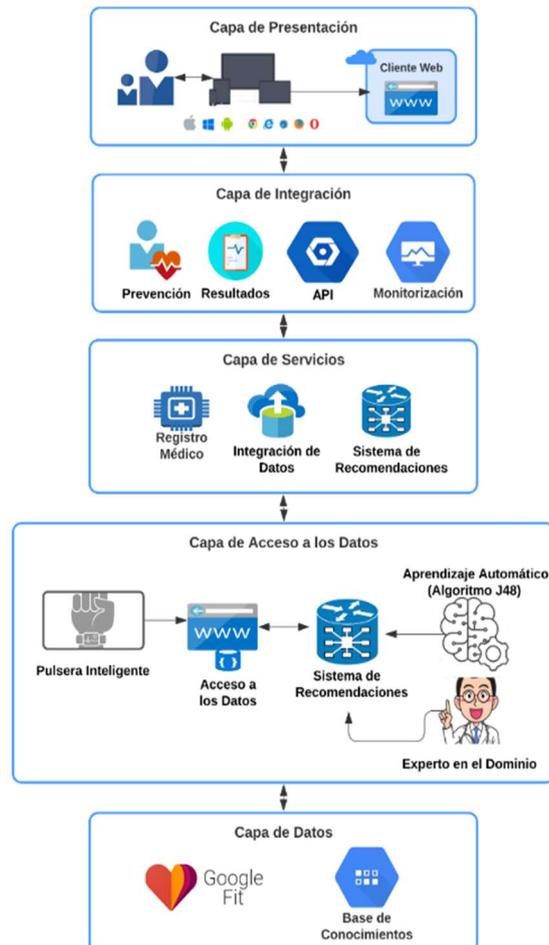


Fig. 1. Arquitectura de la aplicación web para la detección y prevención de la hipertensión.

Una vez extraída la información, mediante el uso de la *Api* de Google Fit™ se obtiene un conjunto de datos que debe validarse por expertos del dominio, de modo que se utilice posteriormente por un algoritmo aprendizaje automático para su clasificación. A continuación, se muestra se describe las principales funciones del módulo:

- **Monitorización:** Permite al usuario, mediante el uso de *wearables*, medir las siguientes variables fisiológicas: Frecuencia Cardíaca, Conteo de Pasos, Distancia Recorrida, Calorías Quemadas, Minutos Activos, Temperatura en la piel, Oxigenación en la Sangre y Presión Arterial. Estas variables se utilizan como parámetros para predecir si una persona es hipertensa o no, mediante el uso del algoritmo de aprendizaje automático J48.

- **Detección:** Mediante el uso de la *Api* de Google Fit™ y el uso de técnicas de aprendizaje automático, las variables que se obtienen del *wearable* y extraídas de la API, se usan para predecir y detectar si una persona es hipertensa o no, estos resultados se consultan en la aplicación Web como parte integral del módulo.
- **Prevención:** Se implementan recomendaciones médicas que ayudan al usuario a realizar acciones, procedimientos o tratamientos, que mejoren la salud del usuario que lo requiera.

La aplicación Web para la detección de hipertensión tiene una arquitectura basada en capas para una mejor escalabilidad, robustez y fácil mantenimiento. Esta arquitectura en capas y sus módulos se describen en la Figura 1.

La aplicación Web tiene como objetivo medir la actividad física diaria y otros factores para detectar la posible hipertensión por parte del paciente. A continuación, se explican los componentes y funcionalidades de la arquitectura.

a) Aplicación Web: La aplicación incorpora un módulo de monitorización para comprobar las constantes vitales y la actividad física del paciente y capturar sus síntomas diarios. Asimismo, la aplicación se apoya en un sistema de alertas y recomendaciones para el seguimiento de los pacientes. Las estadísticas sobre el estado de salud del paciente se presentan a través de las siguientes gráficas:

- **Patrones de actividad.** Recolecta los datos de las actividades diarias mediante los sensores del dispositivo.
- **Minutos activos.** Son los minutos contabilizados por Google Fit™ durante los cuales el paciente realiza cualquier tipo de actividad física (por ejemplo, caminar, spinning, entrenamiento de fuerza, entre otros). Estos minutos se registran de forma automática, manual o a través de aplicaciones de terceros conectadas con Google Fit™.
- **Calorías quemadas.** La aplicación calcula cuántas calorías queman los pacientes de forma constante en función de factores como el sexo, la altura y el peso. Este cálculo indica aproximadamente el total de calorías quemadas e incluye la tasa metabólica basal, no solo las calorías quemadas durante el ejercicio.
- **Recuento de pasos.** Conteo total de pasos cada vez que se realiza una actividad física.
- **Distancia recorrida.** Calcula la distancia total recorrida en los últimos días caminando o corriendo.
- **Frecuencia cardíaca.** Calcula la frecuencia cardíaca media del paciente en los últimos días.
- **Oxigenación en la sangre.** Calcula la hemoglobina en porcentaje en la sangre del paciente.
- **Temperatura.** Calcula la temperatura en la piel del paciente en los últimos días.

b) Sistema de recomendaciones: La aplicación tiene dos repositorios: La nube de Google Fit y una base de conocimientos. En Google Fit™, el sistema de la app

almacena los datos del paciente recogidos por los sensores del wearable, mientras que la base de conocimientos incluye un conjunto de reglas para detectar la hipertensión.

- c) **Sensores y dispositivos del internet de las cosas:** La aplicación registra wearables para monitorizar las variables biométricas del paciente. Dichos dispositivos se sincronizan con la plataforma de Google Fit™ para el almacenamiento de datos. Los wearables recogen los datos del paciente en tiempo real y estos datos se utilizan posteriormente para detectar la hipertensión.

La arquitectura de la aplicación se divide en cinco capas, que se explican a continuación.

Capa de presentación. Recibe las peticiones de los usuarios introducidas a través de un navegador (Chrome, Safari y Firefox) para los sistemas operativos más populares (es decir, Windows, MacOS X y Linux). Sin embargo, dada la modularidad y escalabilidad de la arquitectura, las capacidades de la capa de presentación pueden extenderse a otras plataformas y dispositivos, ya que la capa de integración más baja proporciona una API basada en servicios.

Capa de integración. Ofrece una API para acceder a diversas funcionalidades de las capas más bajas, por lo que ofrece un bajo acoplamiento entre el usuario y la funcionalidad del sistema. A grandes rasgos, esta capa permite a la aplicación introducir nuevos pacientes añadiendo una nueva plataforma o soporte de dispositivo a través de la capa de presentación.

Además, la capa de integración sirve tanto de aislamiento de enlaces como de mecanismo de seguridad para acceder a los procesos y funciones de la capa de servicios. Asimismo, esta capa incluye el acceso a los cuatro servicios principales para la gestión de hipertensión: 1) detección, 2) prevención, 3) pronósticos y 4) tratamiento.

Capa de servicios. Incluye tres servicios: 1) integración de datos, 2) sistema de recomendaciones y 3) registros médicos. El servicio de integración de información es un orquestador de la información procedente de diferentes fuentes de datos, y anonimiza los datos sensibles utilizados en las recomendaciones.

Por su parte, el sistema de recomendación utiliza diferentes algoritmos de recomendación para generar y proporcionar recomendaciones médicas específicas para el tratamiento de la hipertensión. Por último, el servicio de registros médicos controla los expedientes electrónicos de los pacientes, despliega dicha información a través de la capa de presentación y guarda y recupera los datos a través de la capa de acceso a datos.

Capa de acceso a los datos. Da acceso a toda la información de entrada/salida de la arquitectura y facilita el acceso a las distintas fuentes de datos desde la capa de datos. Para ello, la capa de acceso a los datos invoca los tres módulos de acceso: Historial Médico, Recomendaciones y *Wearables*.

El módulo de Historial Médico reenvía a la historia clínica electrónica del paciente y monitoriza las variables médicas y los parámetros físicos del mismo. Las recomendaciones se basan tanto en el usuario (paciente) como en los diagnósticos médicos históricos.

Capa de datos. Almacena todos los datos estructurados y no estructurados utilizados por las capas superiores. Al mismo tiempo, la capa de datos puede añadir nuevas fuentes

de datos al sistema. Todos los datos recibidos de un dispositivo con sensores ópticos de frecuencia cardíaca (por ejemplo, los *wearables* de Xiaomi™ o Amazfit GTR 3 Pro™) se almacenan en Google Fit™.

Posteriormente, estos datos se analizan. La base de conocimientos proporciona a la aplicación Web capacidades de razonamiento para detectar patrones y predecir posibles casos de hipertensión. Para ello, la base de conocimientos se apoya en una serie de variables médicas asociadas a un conjunto de reglas.

4. Caso de estudio: monitorización de personas para la detección y prevención de hipertensión

El caso de estudio consiste en el desarrollo de una aplicación web para la detección y prevención de hipertensión. Para la fase de pruebas, la aplicación se utilizó para supervisar el estado de salud de un paciente de 37 años durante dos meses, monitorizando sus variables biomédicas durante las actividades físicas diarias.

El caso de estudio implicó, por tanto, la prueba de concepto de las principales funcionalidades de la aplicación web. Además, se solicitó al paciente que registrara sus hábitos diarios, antecedentes personales y familiares, así como sus síntomas en la aplicación:

- a) Monitorización de las actividades físicas diarias del paciente en tiempo real mediante un dispositivo inteligente.
- b) Monitorizar la frecuencia cardíaca, calorías quemadas, recuento de pasos, distancia recorrida, frecuencia cardíaca promedio, frecuencia cardíaca en reposo, oxigenación en la sangre, temperatura y minutos de actividad física del paciente en tiempo real.
- c) Generar un informe con los datos del paciente en un periodo de tiempo determinado: minutos activos, calorías quemadas, recuento de pasos, distancia recorrida, frecuencia cardíaca media, frecuencia cardíaca en reposo, oxigenación en la sangre, temperatura y minutos de actividad física.
- d) Detectar la posible hipertensión del paciente.

4.1. Características de la aplicación

La aplicación se desarrolló en el lenguaje de programación PHP junto con el marco de trabajo AngularJS y las bibliotecas D3 CHART y CHARTJS. Además, se utilizó Google™ API Client Library para PHP para conectar con las APIs de Google™. Google API Client Library permite trabajar con las APIs de Google™ como Gmail™, en el servidor. La aplicación se configuró para recopilar los datos necesarios del paciente desde un reloj inteligente Amazfit GTR 3 Pro™ así como de un dispositivo móvil, ambos sincronizados con Google Fit™.

Obsérvese que los datos recogidos del paciente eran privados y solo se mostraban mediante la autenticación del usuario en la plataforma a través de Gmail™. El Amazfit GTR 3 Pro™ cuenta con el mismo sensor que los modelos estándar, es decir, el BioTracker PPG de seis fotodiodos y, como tal, es capaz de monitorizar la frecuencia

Tabla 1. Parámetros asociados con la hipertensión.

Parámetro	Rango	Descripción
Edad	18-99 años	Edad de la persona.
Sexo	Femenino o Masculino	El sexo de la persona. (Hombre=H o Mujer=M).
Presión Arterial (PAS)	PAS: 120-129 mmHg (Normal) PAD: 80-89 mmHg (Normal)	PAS: Presión Arterial Sistólica. Presión de la sangre en la arteria cuando se contrae el corazón. PAD: Presión Arterial Diastólica. Presión de la sangre en la arteria cuando el corazón se relaja entre latidos.
Sedentarismo		Falta de actividad física regular. (Sí=1 No=0)
Índice de masa corporal (IMC)	IMC < 18.5 = Insuficiente. IMC > 18.5 e IMC < 24.9 = Normal o saludable. IMC > 25.0 e IMC < 29.9 = Sobrepeso. IMC >= 30.0 = Obesidad.	Índice de masa corporal de la persona.
Alcohólico		Si la persona bebe alcohol. (Sí=1 No=0)
Fumador		Si la persona fuma. (Sí=1 No=0)
Antecedentes Familiares (AF)		Familiares de la persona, que tuvieron hipertensión u otra enfermedad como diabetes. (Sí=1 No=0)
Etnia Negra *(EN)		Personas con color de piel negra. (Sí=1 No=0)

cardíaca, el sueño, el nivel de oxígeno en sangre, el estrés y el ritmo respiratorio, además de ciclo menstrual.

En cuanto a Google Fit™, utiliza los sensores de movimiento del reloj inteligente para sincronizarse con un dispositivo móvil y el GPS para detectar cuándo una persona está en movimiento. A continuación, utilizando los parámetros obtenidos, calcula la intensidad de la actividad física y otros datos, como las calorías quemadas.

4.2. Parámetros y reglas para la detección de hipertensión

La aplicación Web monitoriza y supervisa la salud del paciente durante sus actividades diarias con una precisión similar a la del sistema de monitorización de ECG ambulatorio. Al registrar los parámetros de las constantes vitales del paciente en tiempo real, la aplicación puede detectar casos de hipertensión. La Tabla 4.2.1 enumera los principales parámetros asociados para la detección de la hipertensión y los factores de riesgo más recurrentes con base en los registros válidos de 14,495 pacientes publicados

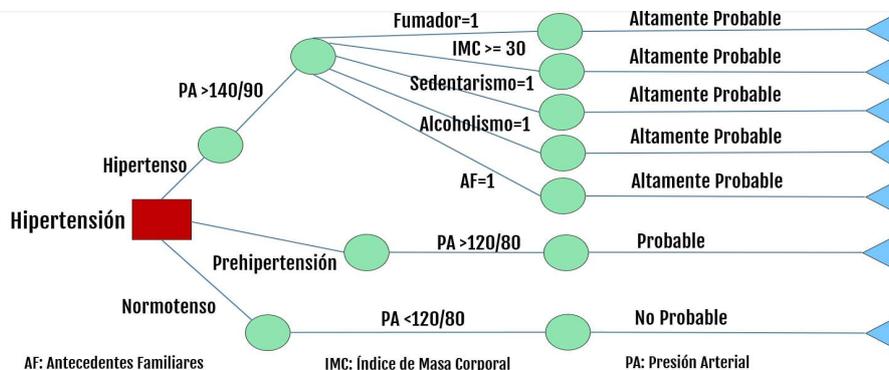


Fig. 2. Árbol de decisión para la detección de la hipertensión.

en diferentes datasets de plataformas especializadas en la ciencia de los datos como son Kaggle [20], Data Europa [21] y Dryad Data [22].

Teniendo en cuenta los parámetros mencionados anteriormente, se desarrolló un conjunto de reglas, mediante el uso del API de Weka que está basado en Java específicamente la versión 3.8.5. Se utilizó el algoritmo J48 que es una implementación open source en lenguaje de programación Java del algoritmo C4.5, esto quiere decir que tanto C4.5 y J48 son algoritmos de clasificación usados para generar árboles de decisión para la detección de hipertensión que se implementaron en el sistema de la aplicación web como se puede ver en la Figura 2.

Adicionalmente, en la Tabla 4.2.2 se muestran las métricas (coeficiente de correlación, error absoluto medio, raíz del error cuadrático medio, error relativo absoluto, raíz del error cuadrático relativo) de los algoritmos basados en árboles de decisiones que se utilizaron en los entrenamientos para la detección de hipertensión mediante el uso de la API de Weka.

A medida que se activan las reglas, se activa la acción del envío de recomendaciones para apoyar al paciente. La Figura 3 ejemplifica cómo la aplicación Web envía recomendaciones a los pacientes.

4.3. Monitorización del paciente

En la sección de “*Mis wearables*” de la aplicación, los usuarios pueden introducir sus datos de salud desde su reloj inteligente. Sin embargo, los usuarios también pueden sincronizar en tiempo real su dispositivo wearable con la aplicación para, finalmente, descargar un informe en PDF de sus datos de salud. En concreto, al seleccionar la opción “*Sincronizar tu cuenta de Google Fit*”, la aplicación recupera los datos almacenados en la cuenta de Google™ en la nube del usuario.

Para ello, el usuario debe autenticar su cuenta con su dirección de correo electrónico y contraseña de Google™ para permitir la sincronización de datos. A continuación, el servidor de Google Fit™ recupera de la nube los datos del wearable registrado. Por último, estos datos se muestran en forma de gráficas en la interfaz de la aplicación. La Figura 4 muestra parte del panel del paciente de la aplicación web.

Tabla 2. Métricas de algoritmos basados en árboles de decisiones.

Algoritmo	Decision Stump	MSP	Random Forest	Random Tree	REP Tree
Coefficiente de correlación	0.2248	0.3588	0.3179	0.1828	0.2938
Error absoluto medio	0.3966	0.3315	0.3585	0.3486	0.3653
Raíz del error cuadrático medio	0.4497	0.4434	0.4461	0.5904	0.4492
Error relativo absoluto	93.35%	78.01%	84.37%	82.03%	85.97%
Raíz del error cuadrático relativo	97.53%	96.16%	96.75%	128.04%	97.42%
Número total de instancias	1007	1007	1007	1007	1007

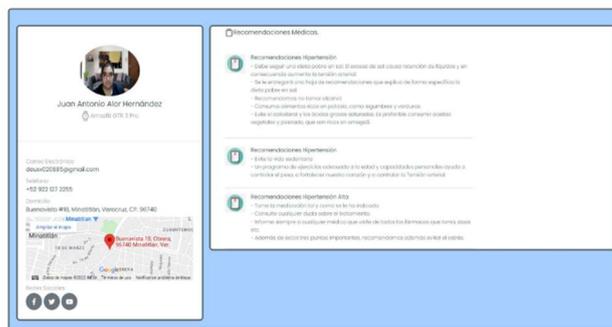


Fig. 3. Ejemplo de recomendaciones emitidas por la aplicación Web de detección de hipertensión.



Fig. 4. Panel del paciente de las variables biométricas de la aplicación Web.

5. Conclusiones y trabajo a futuro

El constante incremento de personas hipertensas es un tema muy alarmante, ya que el no llevar un control adecuado de la presión arterial termina en pérdida de la vida. Es por esto que últimamente se ha apostado por invertir tiempo y conocimientos en el desarrollo de dispositivos más cómodos y menos dolorosos, que consigan ayudar no solo a las personas que ya están diagnosticadas como hipertensas sino también a aquellas que desarrollan dicha enfermedad.

En este trabajo, se presentó una aplicación web, que con ayuda de una pulsera inteligente, es capaz de detectar y prevenir la hipertensión de un individuo aplicando técnicas de Aprendizaje Automático mediante el algoritmo de clasificación J48. Como trabajo a futuro se pretende abordar más enfermedades crónico-degenerativas como son la diabetes y hepatitis, mediante el uso de wearables con actualizaciones en sus tipos de lectura y funcionalidades, que presenten las certificaciones correspondientes y avaladas por las autoridades en materia de salud.

Se pretende integrar el envío de alarmas al paciente con el objetivo de llevar a cabo las recomendaciones médicas realizadas por la aplicación. Finalmente, se integrarán más APIs de pulseras inteligentes como: Fitbit™, Polar™, así como la implementación de diversos algoritmos de aprendizaje automático para mejorar la eficiencia de la aplicación.

Agradecimientos. Este trabajo de investigación fue patrocinado por el Consejo Nacional de Ciencia y Tecnología de México (CONACYT) y la Secretaría de Educación Pública (SEP) de México a través del programa PRODEP. Los autores también agradecen al Tecnológico Nacional de México (TecNM) por apoyar este proyecto.

Referencias

1. Piña, M., Araujo, G., Castillo, C.: Hipertensión arterial un problema de salud pública en México (2020)
2. Campos, I., Hernández, L., Pedroza, A., Medina, C., Barquera, S.: Hypertension in Mexican adults: Prevalence, diagnosis and type of treatment. *Ensanut MC 2016. Salud Publica Mex*, vol. 60, pp. 233–243 (2018)
3. Sazonov, E., Neuman, M. R.: *Wearable sensors: Fundamentals, implementation and applications* (2014)
4. Janjua, G., Guldenring, D., Finlay, D., McLaughlin, J.: Wireless chest wearable vital sign monitoring platform for hypertension. *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS.*, pp. 821–824 (2017)
5. Ni, H., Cho, S., Mankoff, J., Yang, J., Dey, A. K.: Automated recognition of hypertension through overnight continuous HRV monitoring. *J. Ambient Intell. Humaniz. Comput.*, vol. 9, pp. 2011–2023 (2018) doi: /10.1007/s12652-017-0471-y.

6. Afeni, B., Aruleba, T., Oloyede, I.: Hypertension prediction system using naive bayes classifier. *Journal of Advances in Mathematics and Computer Science*, vol. 24, no. 2, pp. 1–11 (2017) doi: 10.9734/JAMCS/2017/35610.
7. Ruiz-Fernandez, D., Marcos-Jorquera, D., Gilart-Iglesias, V., Vives-Boix, V., Ramírez-Navarro, J.: Empowerment of patients with hypertension through BPM, IoT and remote sensing. *Sensors*, vol. 17, no. 10, pp. 2273, (2017) doi: 10.3390/s17102273.
8. Irizarry, T., Allen, M., Suffoletto, B. P., Einhorn, J., Burke, L. E., Kamarck, T. W., Rollman, B. L., Muldoon, M. F.: Development and preliminary feasibility of an automated hypertension self-management system. *The American Journal of Medicine*, vol. 131, no. 9, pp. 1125.e1–1125.e8, (2018) doi: 10.1016/j.amjmed.2018.04.038.
9. Thangada, N. D., Garg, N., Pandey, A., Kumar, N.: The emerging role of mobile-health applications in the management of hypertension. *Current Cardiology Reports*, vol. 20 (2018) doi: 10.1007/s11886-018-1022-7.
10. Singh, N., Singh, P., Bhagat, D.: A rule extraction approach from support vector machines for diagnosing hypertension among diabetics. *Expert Syst. Appl.*, vol. 130, pp. 188–205 (2019)
11. Carvajal, O., Melin, P., Miramontes, I., Prado-Arechiga, G.: Optimal design of a general type-2 fuzzy classifier for the pulse level and its hardware implementation. *Eng. Appl. Artif. Intell.*, vol. 97, no. 104069 (2021)
12. Guzmán, J. C., Miramontes, I., Melin, P., Prado-Arechiga, G.: Optimal genetic design of type-1 and interval type-2 fuzzy systems for blood pressure level classification. *Axioms*, vol. 8, no. 1, (2019) doi: 10.3390/axioms8010008.
13. Kitt, J., Fox, R., Tucker, K. L., McManus, R. J.: New Approaches in Hypertension Management: a Review of Current and Developing Technologies and Their Potential Impact on Hypertension Care. *Curr. Hypertens. Rep.*, vol. 21 (2019)
14. Mankar, K., Gawande, P., Thakare, A. P.: Design Remote Healthcare Monitoring System for Hypertensive Patient Based On IOT. *Int. J. Electron. Commun. Soft Comput. Sci. Eng.*, vol. 5, pp. 10–15 (2020).
15. Wu, X., Yuan, X., Wang, W., Liu, K., Qin, Y., Sun, X., Ma, W., Zou, Y., Zhang, H., Zhou, X., et. al.: Value of a machine learning approach for predicting clinical outcomes in young patients with hypertension. *Hypertension*, pp. 1271–1278 (2020)
16. Kario, K.: Management of Hypertension in the Digital Era: Small Wearable Monitoring Devices for Remote Blood Pressure Monitoring. *Hypertension*, pp. 640–650 (2020)
17. Welykholowa, K., Hosanee, M., Chan, G., Cooper, R., Kyriacou, P. A., Zheng, D., Allen, J., Abbott, D., Menon, C., Lovell, N. H., et. al.: Multimodal Photoplethysmography-Based Approaches for Improved Detection of Hypertension. *J. Clin. Med.*, vol. 9, no. 1203 (2020)
18. Tsoi, K., Yiu, K., Lee, H., Cheng, H. M., Wang, T. D., Tay, J. C., Teo, B. W., Turana, Y., Soenarta, A. A., Sogunuru, G.P.: Applications of artificial intelligence for hypertension management. *J. Clin. Hypertens.*, vol. 23, pp. 568–574 (2021)
19. Zhang, Y., Yang, N., Si, G., Zhang, Y., Dong, Z., Huang, Y., Tan, X.: What matters the adherence with BP 24-hr self-monitoring wearable device among hypertensive patients? A population-based survey. *Transl. Behav. Med.*, vol. 10, pp. 1053–1063 (2021)
20. Ulianova, S.: Cardiovascular Disease dataset. | Kaggle (2022)
21. Administração Central do Sistema de Saúde (ACSS): Hypertension Program Activity - Data Europa EU (2021)

Juan Antonio Alor-Hernandez, Nancy Aracely Cruz Ramos, Giner Alor-Hernández, et al.

22. Khanal, M. K., Dhungan, R. R., Bhandari, P., Gurung, Y., Paudel, K. N.: Prevalence, associated factors, awareness, treatment, and control of hypertension: Findings from a cross sectional study conducted as a part of a community based intervention trial in Surkhet, Mid-western region of Nepal. *Plos Biology*, Submit Your Research, (2017) doi: 10.1371/journal.pone.0185806.

Análisis del estado de ánimo en personas con tendencia suicida mediante testores típicos

Daniel Alejandro Barajas Aranda, Aurora Torres Soto,
María Dolores Torres Soto

Universidad Autónoma de Aguascalientes,
México

{atorres, mdtorres}@correo.uaa.mx,
alengot@hotmail.com

Resumen. El uso de testores típicos para enfrentar problemas del mundo cotidiano es cada vez más frecuente, tal es el caso de este artículo, en donde se utilizan para ayudar a comprender el grave problema del suicidio que existe en Aguascalientes, México; un problema que va en aumento conforme transcurre los años. Este estudio se llevó a cabo con las bases de datos proporcionadas por el departamento de psicología de la Universidad Autónoma de Aguascalientes. Donde se han recopilados datos de personas con tendencia suicida. Este trabajo se centra en el análisis de síntomas asociados al estado de ánimo; tales como problemas para dormir, pérdida de peso, y problemas emocionales como la autoestima, entre otros. Como parte del análisis se obtuvo el conjunto de todos los testores típicos y el cálculo de su peso informacional. Los resultados sugieren que las personas con tendencia suicida exhiben sentimientos y estados delatores que podrían ser usados para intervenir de manera temprana en este grave problema.

Palabras clave: Testores típicos, suicidio, estado de ánimo, reconocimiento de patrones.

Analysis of Mood in People with Suicidal Tendency Using Typical Testors

Abstract. The use of typical testors to face problems of every day's world is increasingly frequent, such is the case of this article, where they are used to help in understanding the fundamental problem of suicide that exists in Aguascalientes, Mexico; a problem that is increasing as the years go by. This study was conducted with the databases provided by the psychology department of the Autonomous University of Aguascalientes. Where data on people with suicidal tendencies have been collected. This work focuses on the analysis of

symptoms associated with mood, such as sleep problems, weight loss, and emotional problems such as self-esteem, among others. As part of the analysis, the set of all typical testors and the calculation of their informational weight were obtained. The results suggest that people with suicidal tendencies exhibit telltale feelings and states that could be used to intervene early in this severe problem.

Keywords: Typical testors, suicide, mood, pattern recognition.

1. Introducción

Según datos del INEGI (Instituto Nacional de Estadística y Geografía) el número de suicidios en Aguascalientes está aumentando cada año [1], en el año 2012 se consumaron 113 suicidios, y a partir de ese año esa cifra no baja de 100. Para el año 2020 este número alcanzó una cifra histórica con 184 suicidios. De igual manera de acuerdo a Campos [2] el número de intentos de suicidio aumenta proporcionalmente, lo que ocasiona un grave problema, especialmente si vemos que por cada suicidio consumado, hay al menos 10 intentos [2].

Dentro de las principales consecuencias que deja el suicidio, están la afectación social y la familiar, especialmente por el hecho de que la gran mayoría de los suicidas se concentran en una edad entre 15 y 34 años (figura 1). En estas edades el impacto económico en la sociedad es muy fuerte debido a su capacidad productiva, aunado a esto solo un pequeño porcentaje de personas suicidas se encuentra sin desempeñar alguna actividad laboral [1].

Para identificar las características de personas con tendencia suicida para posibilitar una intervención temprana, se ha recopilado información asociada al estado de ánimo. Dentro de esta información podemos encontrar factores como: dormir sin descansar, dormir más de lo acostumbrado, pérdida de peso, sentimientos de ser mala persona, entre otras.

Mediante el uso de una herramienta computacional basada en CUDA [3], se ha obtenido el conjunto completo de testores típicos, y se han determinado los principales patrones relacionados con el estado de ánimo, que presentan las personas entre 15 y 34 años que atentan contra su vida en el estado de Aguascalientes, México.

Para indicar la frecuencia de la presencia de cada variable involucrada en el conjunto total de los testores típicos, se ha obtenido el peso informacional [4], eligiendo las variables que tienen un peso informacional mayor al 40% para su análisis.

1.1. Suicidio

La muerte es el cese de funciones biológicas, el cual según el protocolo médico se caracteriza por el término de las funciones cardio respiratorias y/o neuro-cerebrales de manera irreversible, ocurriendo una pérdida de constantes vitales por lo que el cuerpo no es capaz de sostenerse por sí mismo [2]. El suicidio (sui 'de sí mismo' y caedēre 'matar') [5], es el acto por el cual una persona atenta contra su vida de manera voluntaria.

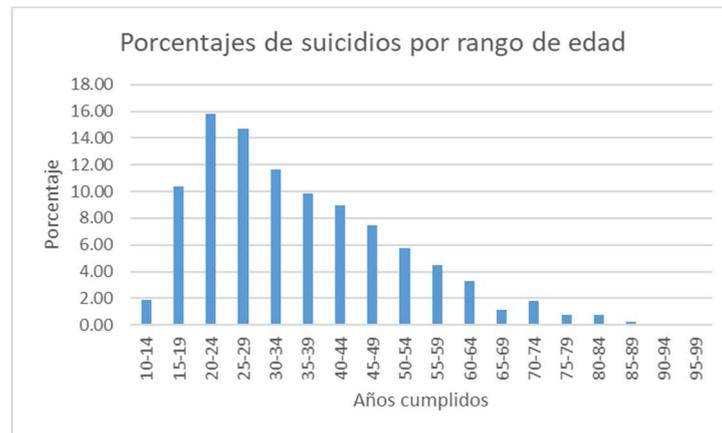


Fig. 1. Porcentajes de suicidio por rango de edad en Aguascalientes 2012-2020 [1].

De acuerdo con Barajas [6] los factores asociados al suicidio pueden categorizarse en diversas causas, las cuales se agrupan en alteraciones fisiológicas, que afectan a nivel biológico; psicológicas, las cuales agrupan alteraciones, trastornos y enfermedades mentales (entre ellos los relacionados al estado de ánimo); sociales, en las que se consideran las interacciones de la persona con su entorno cultural; y ambientales, las cuales son factores externos a las personas y están relacionadas con el entorno en el que habitan.

1.2. Reconocimiento de Patrones

El reconocimiento de patrones es una disciplina científica que busca la clasificación de objetos en un número de categorías o clases[7]. Estos objetos van desde señales, imágenes, sonidos y cualquier tipo de medida que se pueda requerir.

Dentro de las principales aplicaciones del reconocimiento de patrones se pueden mencionar: reconocimiento de voz, visión artificial, diagnóstico asistido por computadora y reconocimiento de caracteres. Esta disciplina se conoce desde los años 60's y se ha desarrollado gracias al avance tecnológico, logrando separarse del área de estadística para ser una disciplina autónoma [8].

La teoría de testores se formuló a mediados de los años 50's en la ex-uni6n de las rep6blicas socialistas sovi6ticas (URSS) como una de las direcciones cient6ficas independientes de la cibern6tica matemática [9]. En donde se utiliz6 por primera vez para encontrar desperfectos en circuitos el6ctricos.

Un testor es un conjunto de caracter6sticas capaces de distinguir entre clases, pues ning6n objeto de cierta clase puede ser confundido con alguno de otra clase [10]. Y un testor t6pico es aquel que contiene el n6mero m6nimo de caracter6sticas, por lo que al perder una de ellas deja de ser testor.



Fig. 2. Metodología.

2. Materiales y métodos

En la realización de este trabajo se llevaron a cabo diversas fases para crear una metodología que pueda ser replicada, los pasos que se siguieron para establecer la forma en cómo afecta el estado de ánimo en personas con tendencia suicida, se muestran en la figura 2 y se describen a continuación. La base de datos consta de 43 variables de las cuales 40 están asociadas al estado de ánimo.

Dentro de ellas se pueden mencionar los estados de ánimo como los sentimientos de ser mala persona, querer hacerse daño, etc., y los estados físicos como lo son la pérdida de peso, dormir sin descansar, dormir más de lo acostumbrado, entre otras. Estas variables tienen una escala numérica, basada en el número de días en los que se presentaron factores de estado de ánimo, durante las dos semanas anteriores al muestreo. Éste se llevó a cabo por el departamento de psicología de la Universidad Autónoma de Aguascalientes (UAA).

Esta base de datos se sometió a un preprocesamiento en el cual se eliminaron los registros repetidos o incompletos, dejando un total de 253 registros, donde 71 personas presentaban tendencia suicida, y el resto fueron del grupo de control. La base de datos se sometió a un análisis de testores típicos mediante un algoritmo de procesamiento en paralelo, en el cual se obtienen los testores típicos utilizando la arquitectura CUDA [3], obteniendo un total de 33125 testores típicos.

Partiendo de este conjunto total de testores típicos se calculó el peso informativo de cada variable, tomando en cuenta el porcentaje de aparición de estas en el total de casos. Enseguida se estableció el punto de corte de tal manera que se conserven un número adecuado de variables para el análisis. Posteriormente se crearon tablas de contingencia, realizando las comparaciones entre los porcentajes que presentan las personas con tendencia suicida y las que no la presentan.

3. Resultados

Del análisis de testores se obtuvo un total de 33125 testores típicos. Con éstos se calculó el peso informativo y se seleccionó como punto de corte el 40%; al observar la figura 3 se puede apreciar que las variables se agrupan en 4 principales rangos según su peso informativo, $0 < x < 20$, $20 \leq x < 40$, $40 \leq x < 60$, $60 \leq x < 80$.

Como se aprecia en dicha figura existe un gran número de variables por debajo del 40%, así que, a fin de elegir un punto de corte adecuado, se ha optado por tomar aquellas variables que están por encima del 40%, lo que involucra al 50% de las agrupaciones.

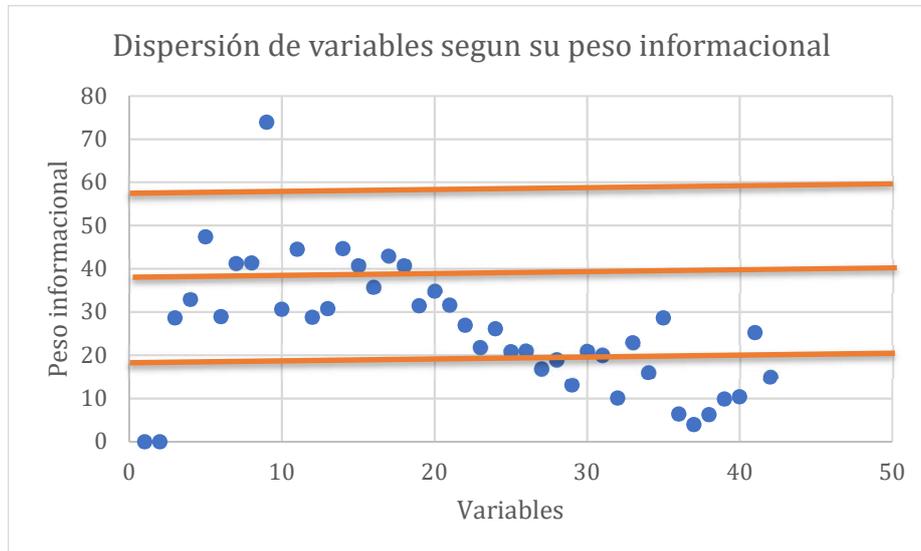


Fig. 3. Dispersión de variables según su peso informacional.

Tabla 1. Variables con peso informacional mayor al 40%.

Peso	Variable
47.443 %	Dormía sin descansar
41.221 %	No podía seguir adelante
41.387 %	Nada me hacía feliz
73.966 %	Sentí que era una mala persona
44.587 %	Dormía más de lo acostumbrado
44.738 %	Sentía deseos de estar muerto
40.702 %	Quería hacerme daño
42.942 %	Estaba a disgusto conmigo mismo
40.744 %	Perdí peso sin intentarlo

con el mayor peso informacional. Con la selección del punto de corte del 40% las variables resultantes son las que se presentan en la tabla 1.

En las siguientes figuras mostramos un análisis de porcentajes de algunas variables en donde se muestran los días en que las personas sintieron esa emoción en los últimos 14 días. En color azul se muestran las personas con tendencia suicida. Mientras que en color naranja son el grupo de control.

En la figura 4 se muestra la variable dormir sin descansar, el grupo de control manifiesta un 19% más el haberlo presentado de 1 a 2 días. Mientras que las personas con tendencia suicida manifiestan haberlo sentido al menos 5 días en mayor medida. Lo que nos muestra que las personas suicidas manifiestan una mayor alteración en los ciclos de sueño, pudiendo desencadenar otros síntomas derivados a esto.

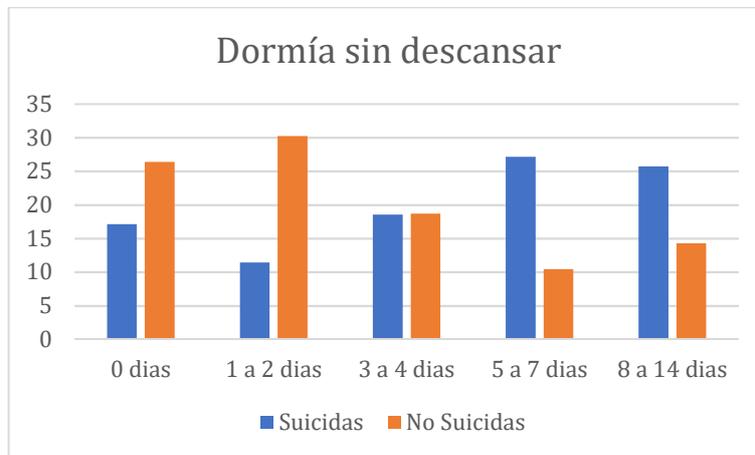


Fig. 4. Comparativa dormir sin descansar.



Fig. 5. Comparativa nada me hacía feliz.

En las siguientes figuras mostramos un análisis de porcentajes de algunas variables en donde se muestran los días en que las personas sintieron esa emoción en los últimos 14 días. En color azul se muestran las personas con tendencia suicida. Mientras que en color naranja son el grupo de control.

En la figura 4 se muestra la variable dormir sin descansar, el grupo de control manifiesta un 19% más el haberlo presentado de 1 a 2 días. Mientras que las personas con tendencia suicida manifiestan haberlo sentido al menos 5 días en mayor medida. Lo que nos muestra que las personas suicidas manifiestan una mayor alteración en los ciclos de sueño, pudiendo desencadenar otros síntomas derivados a esto.

Adicionalmente en la figura 5 las personas con tendencia suicida muestran una mayor prevalencia en sentir que nada los hace feliz. Tal es el caso de que el grupo de control manifiesta un 24% más el no haberse sentido así ningún día, mientras que más

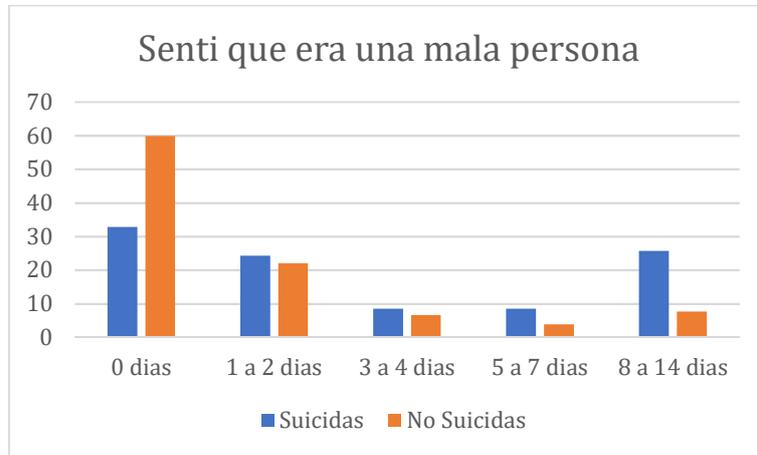


Fig. 6. Comparativo sentimiento de ser una mala persona.

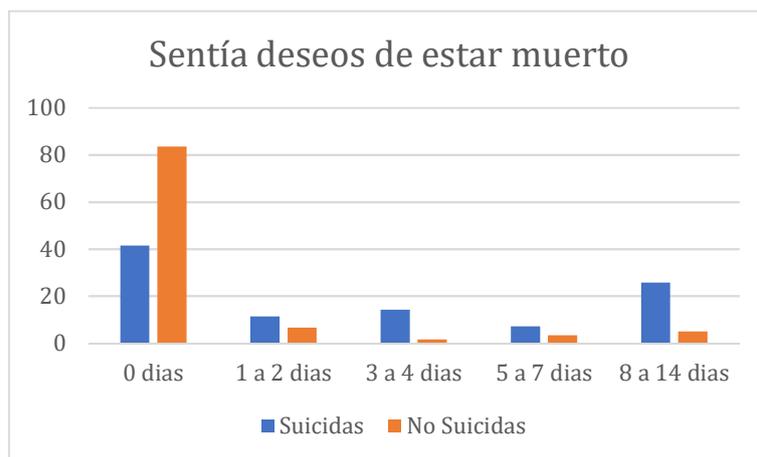


Fig. 7. Comparativo sentimiento del deseo de estar muerto.

del 30% de las personas con tendencia suicida tuvo esta sensación por más de 5 días. Lo cual nos muestra que las personas con tendencia suicida suelen tener mayores índices de depresión.

El sentimiento de ser una mala persona se manifiesta por más tiempo en personas con tendencia suicida. Y el no haberse sentido de esta manera se muestra un 27% más en el grupo de control (figura 6). Demostrando con esto que las personas suicidas muestran una tendencia marcada a sentirse inferiores con mayor regularidad.

De igual manera el sentir deseos de estar muerto se manifiesta por periodos más prolongados y en mayor medida en personas con tendencia suicida (figura 7).

Mientras que en el grupo de control el 83% manifiesta no haber sentido esta emoción. Variable muy importante por considerar, ya que nos indica que estas personas manifiestan los síntomas de querer suicidarse con antelación a cometer el acto, creando con esto una ventana de tiempo en la cual actuar.

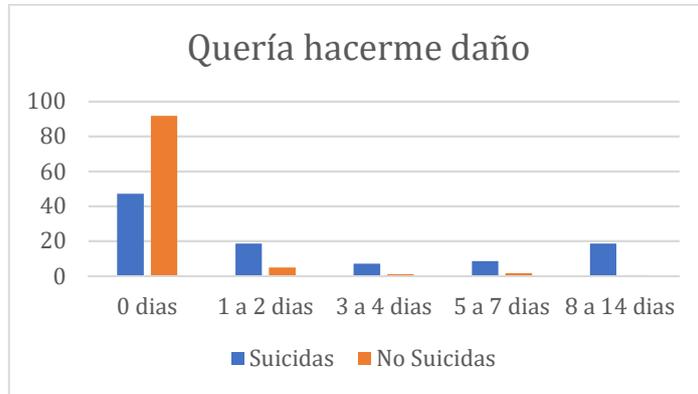


Fig. 8. Comparativa de querer hacerse daño.

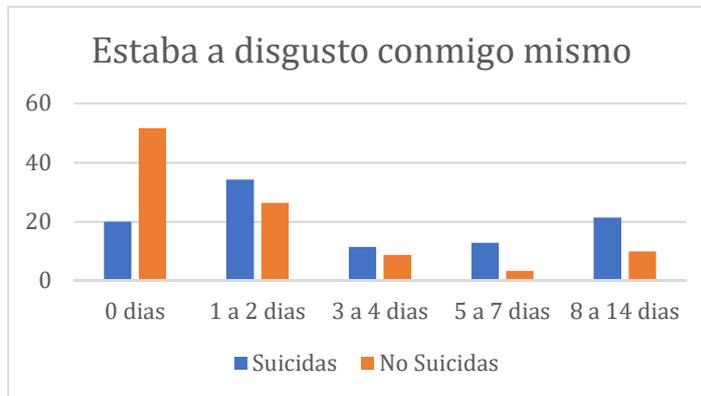


Fig. 9. Comparativa estar a disgusto con si mismo.



Fig. 10. Comparativa perder peso sin intentarlo

En la figura 8 se muestra que las personas sin tendencia suicida manifiestan el no haber sentido el querer hacerse daño en un 92%, mientras que las personas suicidas

tuvieron esta emoción un mayor número de días. Emoción que puede manifestarse en una acción física en la cual la persona se auto inflige, y al estar presente durante un largo tiempo da pie a poder identificarla prematuramente antes de que la lesión sea de riesgo.

Las personas con tendencia suicida manifiestan en mayor medida el sentirse a disgusto con ellos mismos, a diferencia del grupo de control, las cuales manifestaron un 32% más el no haber sentido esta emoción ningún día de acuerdo con la figura 9.

La pérdida de peso también es una característica que se muestra en mayor medida en personas con tendencia suicidas. Mostrando en la figura 10 un 22% más el no manifestarse en el grupo de control. Variable que es fácilmente reconocible y que afecta en gran medida a nivel sistémico a las personas, creando con esto una reacción en cadena que puede desencadenar otros síntomas asociados al suicidio.

4. Conclusiones

En este estudio se pudo observar que las personas con tendencia suicida manifiestan con mayor frecuencia y por tiempos más prolongados, síntomas asociados a un estado de ánimo malo. Tales como sentimientos de ser una mala persona, querer hacerse daño, estar a disgusto consigo mismo.

Aunado esto con mayores problemas relacionados con el ciclo de sueño (dormir más de lo acostumbrado, y el dormir sin descansar) y la pérdida de peso sin intentarlo, aspectos que se relacionan estrechamente con pensamientos suicidas. Se observó que la variable de ser una mala persona tiene un peso informacional muy alto.

Lo que indica que las personas con tendencia suicida experimentan problemas para auto valorarse. De igual manera son alarmantes la frecuencia con la que las personas con tendencia suicida presentan pensamientos de querer hacerse daño o incluso deseos de estar muertos, ya que esto nos indica que los pensamientos suicidas se manifiestan por mucho tiempo antes de que se lleve a cabo tal acción.

Esto nos muestra que se puede tener un buen margen de acción para detectar y actuar antes de que una persona cometa suicidio. El uso de los testores típicos ayudó a identificar las variables más relevantes, logrando que nuestra atención se centre solo en un grupo reducido de ellas.

Estas variables muestran grandes diferencias entre los valores de los grupos de estudio, pues el análisis de testores provee aquellas características con poder de discriminación para clasificar. El uso de los testores típicos puede aplicarse en diversas áreas, tal es el caso de este estudio donde fueron aplicados para estudiar la problemática del suicidio, pudiendo utilizar la metodología en otros problemas no solo relacionados al área de salud.

Referencias

1. Instituto Nacional de Estadística y Geografía Estadísticas Vitales Defunciones generales (2022)
2. Campos, N.: Diplomado en el Protocolo de Actuación (2016)

3. Barajas, D. A., Torres, A., Torres, M. D.: DBT an Algorithm Based on CUDA for Reducing the Time to Obtain Typical Testors. *Res Comput Sci.*, vol. 150, no. 9 (2021)
4. Torres M. D., Ponce de León, E. E., Ochoa, O. C. A., Torres, A., Díaz, E.: Mecanismos de aceleración en selección de características basada en el peso informacional de las variables para aprendizaje no supervisado. *Sistemas, Cibernética e Informática*, vol. 6, no. 2 (2009)
5. Definición de suicidio - Diccionario del español jurídico - RAE.
6. Barajas, D.: Identificación de Factores de Riesgo determinantes en el suicidio en Aguascalientes mediante la técnica de Testores Típicos. Universidad Autónoma de Aguascalientes (2017)
7. Watanabe, S.: *Pattern Recognition: Human and Mechanical*. Wiley, New York (1985)
8. Prandi, D., Gauthier, J. P.: *Pattern Recognition*. In: *SpringerBriefs in Mathematics* (2018)
9. Dmitriev, A. N., Zhuravlev, Y. I., Krendeleiev, F. P.: On mathematical principles of object and phenomena classification. *Discrete Analysis*, pp. 3–15 (1966)
10. Ruiz-Shulcloper, J. R., Guzmán-Arenas, A., Martínez, Trinidad, J. F.: *Enfoque lógico combinatorio al reconocimiento de patrones: Selección de variables y clasificación supervisada*. Instituto Politécnico Nacional, Mexico (1999)

Lógica tradicional para razonamiento no tradicional

J.-Martín Castro-Manzano

Universidad Popular Autónoma del Estado de Puebla,
Facultad de Filosofía,
México

josemartin.castro@upaep.mx

Resumen. En esta contribución mezclamos cuatro lógicas de términos para producir una lógica de términos sintética. La meta es ofrecer un marco lógico de sistemas terminísticos tradicionales para modelar razonamiento no-tradicional. Para alcanzar este objetivo esbozamos brevemente cuatro lógicas diseñadas para capturar cuatro aspectos del razonamiento del lenguaje natural (aserción, numeracidad, modalidad y relevancia); y posteriormente mezclamos dichas lógicas para producir una lógica sintética.

Palabras clave: Lógica de términos, lógica numérica, lógica modal, lógica relevante.

Traditional Logic for Non-Traditional Reasoning

Abstract. In this contribution we mix four term logics as to produce a synthetic logic of terms. The goal is to offer a logical framework of traditional terministic systems to model non-traditional reasoning. To achieve this goal we briefly outline four logics designed to capture four aspects of natural language reasoning (assertion, numeracy, modality, and relevance); and then we mix these logics in order to produce a synthetic logic.

Keywords: Term logic, numerical logic, modal logic, relevance logic.

1. Introducción

Podemos decir que, en cierto sentido, la lógica es una disciplina que se encarga de estudiar la relación de inferencia en el contexto de algún lenguaje natural [19] y que, para llevar a cabo tal empresa, tenemos a nuestra disposición sistemas formales de impronta Fregeana, esto es, sistemas lógicos de primer orden [11,17,34,3,18]; con todo, aun si esta narrativa estándar nos resulta común cuando enseñamos, investigamos o aplicamos lógica esta es la visión recibida de la lógica, después de todo, no hay necesidad de ser personas particularmente críticas como para notar que semejante visión de la lógica, aunque sin duda nos puede ser familiar, no por ello es natural [36,15,22,46,28,12].

En consecuencia, en un esfuerzo por entender el razonamiento en lenguaje natural desde una posición naturalista, Sommers abogó por una revisión de la lógica de términos tradicional desde la década de 1950. Su proyecto se desarrolló a través de tres ramas, en ontología, semántica y lógica (cfr. [37]), que se convirtieron, respectivamente, en una teoría de categorías, una teoría de la verdad y una teoría lógica conocida como Lógica de Términos y Futores [35,36,38,13,15,16].

Esta última teoría es una lógica que usa términos y funtores en lugar de elementos formales de primer orden como variables o cuantificadores (cfr. [32,30,23,36,37]). Siguiendo los principios de este proyecto, en esta contribución mezclamos cuatro lógicas de términos para producir una lógica de términos sintética.

La meta es ofrecer un marco lógico de sistemas de impronta tradicional para modelar razonamiento no-tradicional. Para alcanzar este objetivo esbozamos brevemente cuatro lógicas diseñadas para capturar cuatro aspectos del razonamiento en lenguaje natural (aserción, numeracidad, modalidad y relevancia); y luego mezclamos dichas lógicas para producir una lógica sintética.

2. Cuatro lógicas de términos

2.1. Lógica de términos asertórica

La silogística asertórica, la lógica en el centro de la lógica de términos tradicional, es una lógica de términos que hace buen uso de enunciados categóricos para capturar una noción básica de aserción. Un enunciado categórico es un enunciado compuesto por dos términos, una cantidad y una cualidad. Por lo general, decimos que un enunciado categórico es un enunciado de la forma:

$$\langle \text{Cantidad} \rangle \langle S \rangle \langle \text{Cualidad} \rangle \langle P \rangle.$$

donde Cantidad = {Todo, Algún}, Cualidad = {es, no es}, y S y P son términos-esquema. Desde el punto de vista de la Lógica (asertórica) de Términos y Futores de Sommers & Englebretsen (TFL^α , de ahora en adelante) [35,36,38,13,15,16], decimos:

Definición 1 (Enunciado categórico en TFL^α) Un enunciado categórico en TFL^α es un enunciado de la forma:

$$\pm S \pm P.$$

donde \pm son funtores, y S y P son términos-esquema. Dado este lenguaje (digamos, $\mathcal{L}_{TFL^\alpha} = \langle \mathcal{T}, \pm \rangle$, donde $\mathcal{T} = \{A, B, C, \dots\}$ es un conjunto de términos y \pm es una abreviatura de los funtores + o -), TFL^α ofrece una definición de validez de la siguiente manera [15, p.167]:

Definición 2 (Silogismo válido (en TFL^α)) Un silogismo es válido (en TFL^α) si y solo si:

1. La suma algebraica de las premisas es igual a la conclusión, y

Tabla 1. Una inferencia asertórica válida.

	Enunciado	TFL^α
1.	Toda persona es inteligente.	-P + I
2.	Toda persona lógica es persona.	-L + P
⊢	Toda persona lógica es inteligente.	-L + I

2. El número de conclusiones particulares (*viz.*, cero o uno) es igual al número de premisas particulares.¹

Y así, con esta lógica podemos modelar inferencias asertóricas como la que se muestra en la Tabla 1.

2.2. Lógica de términos numérica

La lógica de términos numérica de Murphree (TFL^ν)—que sirve como una extensión de la silogística numérica [43,44]—es una lógica de términos que trata de capturar razonamiento aritmético mediante el modelado de inferencias con cuantificadores numéricos [29]. En esta lógica, un enunciado numérico es un enunciado de la forma:

$$\langle \text{Cantidad} \rangle \langle n \rangle \langle S \rangle \langle \text{Cualidad} \rangle \langle P \rangle.$$

donde Cantidad = {Todo, Todo excepto, A lo mucho, Al menos, Algún}, $n \in \mathbb{R}^+$, Cualidad = {es, no es}, y S y P son términos-esquema. Formalmente, dado que TFL^ν es una extensión conservadora de TFL^α, decimos:

Definición 3 (Enunciado numérico en TFL^ν) Un enunciado numérico en TFL^ν es un enunciado de la forma:

$$\pm_n S \pm_\epsilon P.$$

donde \pm son funtores, $n, \epsilon > n \in \mathbb{R}^+$, y S y P son términos-esquema. En consecuencia, dado este lenguaje ($\mathcal{L}_{\text{TFL}^\nu} = \langle \mathcal{T}, \pm, \mathbb{R}^+ \rangle$), TFL^ν ofrece la siguiente noción de validez:

Definición 4 (Silogismo válido (en TFL^ν)) Un silogismo es válido (en TFL^ν) si y solo si:

1. La suma algebraica de las premisas es igual a la conclusión,
2. El número de conclusiones particulares (*viz.*, cero o uno) es igual al número de premisas particulares, y
3. (a) El valor de una conclusión universal es igual a la suma de los valores de las premisas universales, o bien (b) El valor de una conclusión particular es igual a la diferencia de la premisa universal menos la particular.²

Siguiendo nuestro patrón de exposición, y como ejemplo, consideremos la inferencia de la Tabla 2.

¹ Debemos mencionar que este enfoque no es solo capaz de representar la inferencia silogística, ya que también puede representar enunciados relacionales, singulares y compuestos con facilidad y claridad [13], pero para nuestros propósitos actuales, esta exposición será suficiente.

² Esta última condición es diferente de la lógica numérica de Szabolcsi, la cual requiere que el valor de las premisas sea igual o mayor que el valor de la conclusión (cfr. [40]).

Tabla 2. Una inferencia numérica válida.

	Enunciado	TFL^ν
1.	Todas excepto 11 personas son inteligentes.	$-_{11}P +_{\varepsilon} I$
2.	Al menos 30 personas lógicas son personas.	$+_{30}L +_{\varepsilon} P$
⊢	Al menos 19 personas lógicas son inteligentes.	$+_{19}L +_{\varepsilon} F$

2.3. Lógica de términos modal

La lógica de términos modal de Englebretsen (TFL^μ)—una versión formal de la silogística modal [1,2,27,24,21,20,39,14,33,41,25]—intenta capturar la modalidad extendiendo TFL^α con □ y ◇ [14,16]. Entonces, dado un término T, TFL^μ permite las siguientes combinaciones: +□ + T (es decir, □ + T), +□ − T (es decir, □ − T), −□ + T (es decir, −□T), −□ − T y, como siempre, el operador ◇ se define como −□−. Así, podemos decir que un enunciado modal *de dicto* es un enunciado de la forma:

$$\langle \text{Modalidad} \rangle (\langle \text{Cantidad} \rangle \langle S \rangle \langle \text{Cualidad} \rangle \langle P \rangle).$$

y un enunciado modal *de re* es un enunciado de la forma:

$$\langle \text{Cantidad} \rangle \langle S \rangle \langle \text{Cualidad} \rangle \langle \text{Modalidad} \rangle \langle P \rangle.$$

donde Modalidad = {◇, □}, Cantidad = {Todo, Algún}, Cualidad = {es, no es}, y S y P son términos-esquema. Así, formalmente:

Definición 5 (Enunciado modal en TFL^μ) Un enunciado modal en TFL^μ es un enunciado de una de las siguientes formas:

$$\mu(\pm S \pm P) | \pm S \pm P | \pm S \pm \mu P.$$

donde ± son funtores, μ es una modalidad y S y P son términos-esquema. Dado este lenguaje ($\mathcal{L}_{TFL^{\mu}} = \langle \mathcal{T}, \pm, \mathcal{M} \rangle$, donde $\mathcal{M} = \{ \diamond, \square \}$), tenemos la siguiente noción de validez:

Definición 6 (Silogismo válido (en TFL^μ)) Un silogismo es válido (en TFL^μ) si y solo si:

1. La suma algebraica de las premisas es igual a la conclusión,
2. El número de conclusiones particulares (*viz.*, cero o uno) es igual al número de premisas particulares,
4. La conclusión no es más fuerte que cualquier premisa,³ y
5. El número de premisas *de dicto*-◇ no es mayor que el número de conclusiones *de dicto*-◇.

Como ejemplo, consideremos la inferencia que se muestra en la Tabla 3.

³ Según [14], existe una transitividad o “fuerza” de los operadores modales de tal forma que: □T implica T□, T□ implica T, T implica T◇, y T◇ implica ◇T.

Tabla 3. Una inferencia modal válida.

	Enunciado	TFL ^μ
1.	Toda persona es necesariamente inteligente.	−P + □I
2.	Toda persona lógica es persona.	−L + P
⊢	Toda persona lógica es necesariamente inteligente.	−L + □I

2.4. Lógica de términos relevante

La lógica de términos relevante (TFL^ρ) es una extensión de TFL^α que captura una noción de relevancia siguiendo algunas ideas del sentido aristotélico de relevancia causal (cfr. [42,45]).

Esta lógica representa fragmentos de discurso complejo (en la medida en que incluyen al menos dos premisas y una conclusión) con modo y figura (porque importa el orden de los enunciados y los términos) en los que una conclusión distinta de las premisas (evitando así circularidad, *petitio principii*) necesariamente (y por lo tanto deductivamente) se sigue y depende de dichas premisas (evitando así la irrelevancia, *non causa ut causa*). En esta lógica decimos que un enunciado relevante es un enunciado de la forma:

$$\langle \text{Cantidad} \rangle \langle S \rangle \langle \text{Cualidad} \rangle \langle P \rangle \langle \text{Bandera} \rangle.$$

donde Cantidad = {Todo, Algún}, Cualidad = {es, no es}, S y P son términos-esquema, y Bandera = {p_i, c} para i ∈ {1, 2, 3, ...} es un conjunto de banderas (de premisa o conclusión). Entonces, formalmente, decimos:

Definición 7 (Enunciado relevante en TFL^ρ) Un enunciado relevante en TFL^ρ es un enunciado de la forma:

$$\pm S \pm P_f,$$

donde ± son funtores, S y P son términos-esquemas f es una bandera. Con este lenguaje ($\mathcal{L}_{\text{TFL}^\rho} = \langle \mathcal{T}, \pm, \mathcal{F} \rangle$, donde \mathcal{F} es un conjunto de banderas), TFL^ρ ofrece una noción de validez de la siguiente manera:

Definición 8 (Silogismo válido (en TFL^ρ)) Un silogismo es válido (en TFL^ρ) si y solo si:

1. La suma algebraica de las premisas es igual a la conclusión,
2. El número de conclusiones particulares (*viz.*, cero o uno) es igual al número de premisas particulares, y
6. Todas las banderas de las premisas se reclaman para llegar a la conclusión mientras que las banderas de la conclusión son diferentes a las banderas de las premisas.

Y así, con esta lógica podemos modelar inferencias relevantes como la que se muestra en la Tabla 4.

Tabla 4. Una inferencia relevante válida.

	Enunciado	TFL ^ρ
1.	Toda persona es inteligente.	-P + I _{p1}
2.	Toda persona lógica es persona.	-L + P _{p2}
⊢	Toda persona lógica es inteligente.	-L + I _c

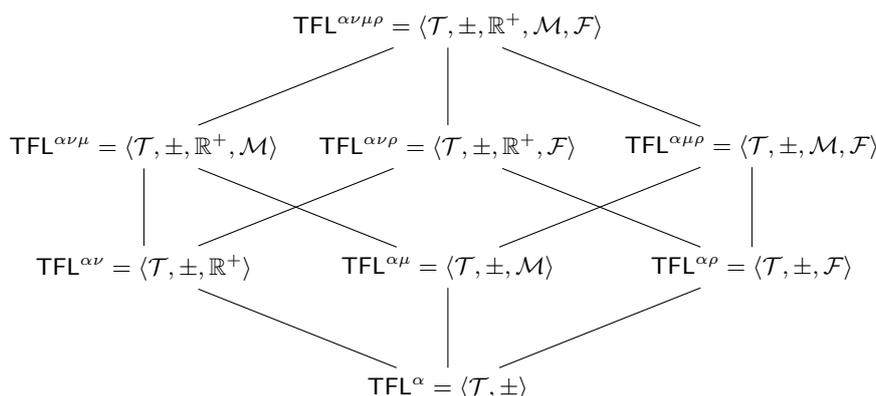


Fig. 1. Una familia de sistemas terminísticos.

3. Una lógica de términos sintética

Como podemos ver hasta este punto, estas lógicas intentan capturar diferentes aspectos básicos del razonamiento en lenguaje natural, a saber, la aserción (TFL^α), la numeracidad (TFL^ν), la modalidad (TFL^μ) y la relevancia causal (TFL^ρ), utilizando una sintaxis de términos; además, también es fácil ver que, dados los lenguajes y las bases deductivas de cada lógica, podemos mezclar estos sistemas mediante uniones (*joints*) e intersecciones (*meets*) de elementos lingüísticos y reglas (Figura 1).

Observemos, por tanto, que TFL^{αν} = TFL^ν, TFL^{αμ} = TFL^μ, y TFL^{αρ} = TFL^ρ. Luego, notemos que TFL^{ανμ} = TFL^{νμ}, TFL^{ανρ} = TFL^{νρ}, TFL^{αμρ} = TFL^{μρ}. Y finalmente, consideremos TFL^{ανμρ}, el sistema sintético de nuestro interés. Por lo tanto, dado el lenguaje $\mathcal{L}_{\text{TFL}^{\alpha\nu\mu\rho}} = \langle \mathcal{T}, \pm, \mathbb{R}^+, \mathcal{M}, \mathcal{F} \rangle$, decimos:

Definición 9 (Enunciado sintético en TFL^{ανμρ}) Un enunciado sintético en TFL^{ανμρ} es un enunciado de una de las siguientes formas:

$$\mu(\pm_n S \pm_\varepsilon P)_f | \pm_n S \pm_\varepsilon P_f | \pm_n S \pm_\varepsilon \mu P_f.$$

donde μ son modalidades, \pm son funtores, $n, \varepsilon \in \mathbb{R}^+$, f es una bandera y S y P son términos-esquema. Por lo tanto, siguiendo nuestro patrón de exposición, también decimos:

Definición 10 (Silogismo válido (en TFL^{ανμρ})) Un silogismo es válido (en TFL^{ανμρ}) si y solo si:

Tabla 5. Un silogismo sintético.

Enunciado	TFL ^{ανμρ}
1. Necesariamente todos excepto 2 A dan 4 B a algún C.	$\Box(-_2A + (+_\varepsilon G +_4 B +_\varepsilon C))_{0p_1}$
2. Al menos 5 D son necesariamente A.	$+_5D +_\varepsilon \Box A_{0p_2}$
3. Todo B es E.	$-_0B +_\varepsilon E_{0p_3}$
⊢ Posiblemente 3 D dan 4 E a algún posible C.	$\diamond(+_3D + (+_\varepsilon G +_4 E +_\varepsilon \diamond C))_{0c}$

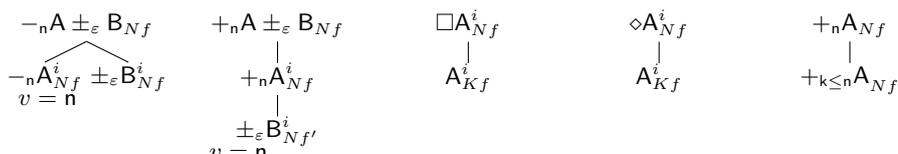


Diagrama 1.1. Reglas de expansión de TFL^{ανμρ}

1. La suma algebraica de las premisas es igual a la conclusión,
2. El número de conclusiones particulares (*viz.*, cero o uno) es igual al número de premisas particulares,
3. (a) El valor de una conclusión universal es igual a la suma de los valores de las premisas universales, o bien (b) El valor de una conclusión particular es igual a la diferencia de la premisa universal menos la particular,
4. La conclusión no es más fuerte que cualquier premisa,
5. El número de premisas *de dicto*- \diamond no es mayor que el número de conclusiones *de dicto*- \diamond , y
6. Todas las banderas de las premisas se reclaman para llegar a la conclusión, mientras que las banderas de la conclusión son diferentes a las banderas de las premisas.

Antes de ofrecer un ejemplo, notemos que podemos representar simultáneamente diferentes aspectos del razonamiento en lenguaje natural. El invariable acto de habla de aserción, ya sea la afirmación o la negación, es capturado por el mismo uso de términos y funtores; la numeracidad está representada por el uso de n (observemos que cuando $n = 0$ o $n = 1$, TFL^ν colapsa con TFL^α); las diferentes formas o modos de aserción se capturan mediante el uso de modalidades (notemos que cuando las modalidades están ausentes, TFL^μ colapsa con TFL^α); y la relevancia se captura mediante el uso de banderas de premisa o conclusión.

Ahora, como ejemplo, consideremos una inferencia con múltiples premisas y que toma en cuenta el fenómeno de aserción (incluso usando relaciones, más allá de la silogística básica), la numeracidad (tanto excepcional como no-excepcional), la modalidad (tanto *de dicto* como *de re*) y la relevancia causal: llamemos a este ejemplo “silogismo sintético” (Cuadro 5). Para evaluar la (in)validez de inferencias como la anterior, podemos producir un conjunto de reglas de expansión para TFL^{ανμρ} como en el Diagrama 1.1 (cfr. [10,31,7,8,4,6,5]). En breve, después de aplicar una regla introducimos un superíndice $i \in \{1, 2, 3, \dots\}$.

Para los enunciados cuyo término inicial es “-” (i.e. enunciados universales), el índice puede ser cualquiera natural; para enunciados cuyo término inicial es “+” (i.e. enunciados particulares), el índice tiene que ser un nuevo natural si aún no tienen un índice. También, después de aplicar una regla creamos un vector v para hacer un seguimiento del valor n (la última es una regla para ordenar términos atómicos con un “+” adjunto).

Además, para el caso modal introducimos un subíndice $K \in \{1, 2, 3, \dots\}$ y dejamos el superíndice fijo como está. Para enunciados cuyo operador inicial es \square (i.e. enunciados necesarios), el subíndice puede ser cualquier natural; para enunciados cuyo término inicial es \diamond (i.e. enunciados problemáticos), el subíndice tiene que ser un nuevo natural si aún no tienen un índice. Y por último, introducimos y mantenemos una bandera $f, f' \in \{p_i, c\}$ (p_i como premisa para $i \in \{1, 2, 3, \dots\}$, c como conclusión). Por supuesto, también contamos con las típicas reglas de rechazo: $-(\pm A) = \mp A$, $-(\pm A \pm B) = \mp A \mp B$, y $-(- - A - - A) = +(-A) + (-A)$.

Y así, como es usual, podemos decir que un *árbol* es un grafo conexo acíclico determinado por nodos y vértices. El nodo en la parte superior se llama *raíz*. Los nodos en la parte inferior se llaman *hojas*. Cualquier camino desde la raíz hacia abajo por una serie de vértices es una *rama*. Para probar la validez de una inferencia, construimos un árbol asociado a dicha inferencia produciendo una sola rama en cuyos nodos ocurran las premisas y el rechazo de la conclusión: esta es la *lista inicial*.

Luego aplicamos las reglas de expansión que nos permiten extender la lista inicial. Entonces, para este sistema sintético decimos que una rama está *abierta* si y solo si no hay términos de la forma $\pm A_{Nf}^i$ y $\mp A_{Nf}^i$ en ella; una rama es *semi-abierta* (o *semi-cerrada*) si y solo si hay términos de la forma $\pm A_{Nf}^i$ y $\mp A_{Nf}^i$; de lo contrario, es *cerrada*.

Una rama abierta se indica escribiendo ∞ al final de la misma; una rama semi-abierta (semi-cerrada) se indica escribiendo $\infty_{f,f}$ ($\infty_{f,f}$); y una rama cerrada, como siempre, se denota por $\perp_{f,f'}$. Consideremos, a modo de ejemplo, el árbol para la inferencia que se muestra en la Tabla 5: Diagrama 1.2. Con estas definiciones y consideraciones, podemos probar las siguientes proposiciones:

Teorema 1 (Completud para $\text{TFL}^{\alpha\nu\mu\rho}$) Sea Γ un conjunto de términos y T un término arbitrario. Entonces, $\Gamma \vdash \pm T$ en $\text{TFL}^{\alpha\nu\mu\rho}$ si y solo si existe un árbol completo y cerrado con $v = 0$ para $\Gamma \cup \{\mp T\} \vdash \perp_{f,f'}$.

Corolario 1 Sea Γ un conjunto de términos y T un término arbitrario. Entonces, $\Gamma \vdash \pm T$ en $\text{TFL}^{\alpha\nu\mu\rho}$ si y solo si existe un árbol semi-cerrado/semi-abierto y completo con $v = 0$ para $\Gamma \cup \{\mp T\} \vdash \perp$.

Con estas distinciones y estos resultados, podemos apreciar un marco lógico de sistemas terminísticos de impronta tradicional para modelar razonamiento no-tradicional (i.e., numérico, modal y relevante). En efecto, decimos que una inferencia es aristotélica o *propter quid* si y solo si cada rama del árbol asociado a la inferencia está cerrada y todas las banderas se acarrean al final de cada hoja; una inferencia es *non sequitur* si y solo si su árbol asociado tiene una rama abierta; de lo contrario, la inferencia es *clásica* (ya sea *quia* o *non causa ut causa*).

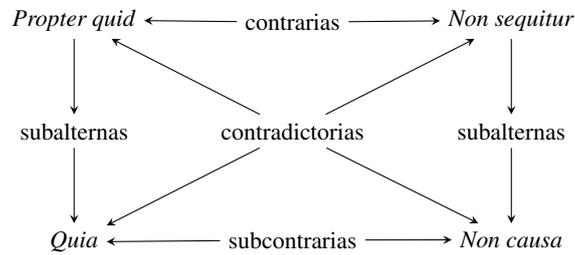


Fig. 2. Cuadrado de relaciones inferenciales.

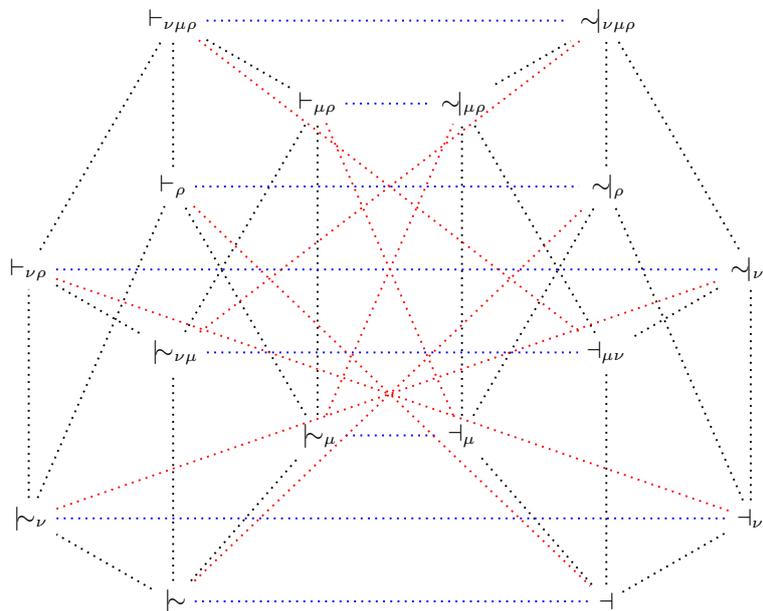


Fig. 3. Espacio de relaciones inferenciales. En negro, las subalternas; en azul, las contrarias; en rojo, las contradictorias. Las inferencias *propter quid* se representan con \vdash ; las inferencias *quia*, con \sim ; las *non causa*, con \dashv ; y las *non sequitur*, con \sim .

De manera similar, si tenemos una instancia de *verum ad* (i.e. $\phi \rightarrow (\psi \vee \neg\psi)$), que es *quia*, también podría ser *non causa*, pero no puede ser *non sequitur*. Asimismo, una *petitio* (i.e. $\phi \rightarrow \phi$) puede ser tanto *quia* como *non causa*, pero ciertamente no es *propter quid* ni *non sequitur*. Y finalmente, si tenemos una inferencia válida, tenemos una inferencia que es tanto *propter quid* como *quia*, pero no *non causa* ni *non sequitur*.

Con base en estos resultados y en la mezcla de lógicas, podemos establecer el siguiente esquema de inferencias: ya no un simple cuadrado, sino un hipercubo de relaciones inferenciales que nos permite observar cómo estas lógicas de impronta tradicional pueden usarse para modelar inferencias más interesantes (Figura 3).

4. Conclusiones

El razonamiento en lenguaje natural es un procedimiento inferencial complejo que puede incluir, más allá del acto de habla de aserción, información sobre numeracidad, modalidad y relevancia causal. Dada esta premisa, en este trabajo hemos combinado cuatro lógicas de términos de impronta tradicional para modelar inferencia no-tradicional (i.e. inferencia numérica, modal y relevante). Finalmente, para cerrar este trabajo, consideremos brevemente algunas posibles objeciones y comentarios con respecto al trabajo futuro.

Objeción 1. Esta propuesta es innecesaria. Si bien es cierto que esta propuesta puede considerarse innecesaria, esta objeción, así planteada, es muy deflacionaria: uno podría preguntarse cuál es la utilidad de cualquier esfuerzo científico. Pero incluso si nos centramos únicamente en la utilidad particular de nuestra propuesta, podemos señalar, al menos, un par de propósitos:

1. El estudio y desarrollo de sistemas como estos contribuyen a la investigación del razonamiento en lenguaje natural usando herramientas más allá de las lógicas de primer orden de matiz Fregeano.
2. Estos sistemas pueden desarrollarse aún más para promover el uso de paradigmas de programación no-clásicos para la programación lógica como en [26,9].

Objeción 2. Esta propuesta es muy compleja. También es cierto que, a primera vista, la propuesta parece compleja. Esta es una buena observación, pero no es una buena objeción. En efecto, sería bastante contra-intuitivo argumentar que no necesitamos estudiar o desarrollar lógicas de orden superior, lógicas híbridas o lógicas no-clásicas porque son más complejas que la clásica. El problema de esta objeción es que no reconoce la ganancia neta de los modelos complejos.

Por lo tanto, incluso si la combinación de estas lógicas parece aumentar la complejidad, ese no es un precio muy alto a pagar si consideramos los beneficios netos de sintetizar cuatro aspectos diferentes del razonamiento en lenguaje natural con un solo mecanismo inferencial.

Objeción 3. Esta propuesta no es inteligencia artificial. Es verdad que esta propuesta parece estar más cerca de la lógica filosófica que de la inteligencia artificial en tendencia, pero eso no implica que no forme parte de la inteligencia artificial.

Por un lado, las relaciones entre lógica e inteligencia artificial tienen una larga historia; y por otro lado, y más importante, las teorías lógicas muestran su nexo con la inteligencia artificial en la medida en que ofrecen una variedad de usos para la segunda: desde usos sencillos en los que la lógica figura como una herramienta de análisis como en la especificación de sistemas formales o la representación de conocimiento, como en este estudio, hasta usos más fuertes en los que la lógica funciona como herramienta de construcción como en el *model checking* o la programación lógica.

Finalmente, dados estos desafíos, nos gustaría mencionar dos líneas de investigación para trabajos futuros: en primer lugar, necesitamos comparar los beneficios netos de esta lógica sintética *vis-à-vis* lógicas Fregeanas; y en segundo lugar, necesitamos ofrecer detalles de implementación.

Agradecimientos. Nos gustaría agradecer a los revisores por sus valiosas observaciones y precisas correcciones. Este trabajo fue financiado por un Proyecto de Investigación UPAEP.

Referencias

1. Becker, A.: Die Aristotelische Theorie der Möglichkeitsschlüsse. Wissenschaftliche Buchgesellschaft (1968)
2. Bocheński, J.: Formale Logik. Karl Alber (1962)
3. Carnap, R.: Die alte und die neue logik. Erkenntnis, vol. 1, pp. 12–26 (1930)
4. Castro Manzano, J. M.: Silogística intermedia, términos y árboles. Tópicos, Revista De Filosofía, , no. 58, pp. 209–237 (2019)
5. Castro Manzano, J. M.: Distribution tableaux, distribution models. Axioms, vol. 9, no. 2 (2020) doi: 10.3390/axioms9020041.
6. Castro Manzano, J. M.: Un método de árboles para la silogística modal. Open Insight, Revista De Filosofía, , no. 58, pp. 209–237 (2020)
7. Castro Manzano, J. M., Reyes Cardenas, P. O.: Term functor logic tableaux. South American Journal of Logic, vol. 4, no. 1, pp. 9–50 (2018)
8. Castro Manzano, J. M.: Murphree’s numerical term logic tableaux. Electronic Notes in Theoretical Computer Science. Proceedings of the Eleventh and Twelfth Latin American Workshop on Logic/Languages, Algorithms and New Methods of Reasoning (LANMR), vol. 354, pp. 17–28 (2020) doi: 10.1016/j.entcs.2020.10.003.
9. Castro Manzano, J. M.: Traditional logic and computational thinking. Philosophies, vol. 6, no. 1 (2021) doi: 10.3390/philosophies6010012.
10. D’Agostino, M., Gabbay, D. M., Hähnle, R., Posegga, J.: Handbook of Tableau Methods. Springer (1999)
11. de Morgan, A.: On the syllogism, no. IV and on the logic of relations. Transactions of the Cambridge Philosophical Society, vol. 10, pp. 331 (1864)
12. Englebretsen, G.: Bare Facts and Naked Truths: A New Correspondence Theory of Truth. Taylor & Francis (2017)
13. Englebretsen, G.: The New Syllogistic. Peter Lang (1987)
14. Englebretsen, G.: Preliminary notes on a new modal syllogistic. Notre Dame J. Formal Logic, vol. 29, no. 3, pp. 381–395 (1988) doi: 10.1305/ndjfl/1093637935.
15. Englebretsen, G.: Something to Reckon with: The Logic of Terms. University of Ottawa Press (1996)
16. Englebretsen, G., Sayward, C.: Philosophical Logic: An Introduction to Advanced Topics. Bloomsbury Academic (2011)
17. Frege, G., Angelelli, I.: Begriffsschrift und andere Aufsätze. Wissenschaftliche Buchgesellschaft (1973)
18. Geach, P. T.: Reference and Generality: An Examination of Some Medieval and Modern Theories. Cornell University Press (1962)
19. Haack, S.: Philosophy of Logics. Cambridge University Press (1978)
20. Hintikka, J., Hintikka, K.: Time & Necessity: Studies in Aristotle’s Theory of Modality. Clarendon Press (1973)
21. Kneale, W., William Kneale, M., Kneale, W., Kneale, M., Conte, A., Press, O. U.: The Development of Logic. Clarendon Press (1962)
22. Kreeft, P., Dougherty, T.: Socratic Logic: A Logic Text Using Socratic Method, Platonic Questions & Aristotelian Principles. St. Augustine’s Press (2004)

23. Kuhn, S. T.: An axiomatization of predicate functor logic. *Notre Dame J. Formal Logic*, vol. 24, no. 2, pp. 233–241 (1983) doi: 10.1305/ndjfl/1093870313.
24. Łukasiewicz, J.: *Aristotle's syllogistic from the standpoint of modern formal logic*. Clarendon Press (1957)
25. Malink, M.: *Aristotle's Modal Syllogistic*. Harvard University Press (2013)
26. Massie, D.: *Computer Implementation of Term Functor Logic (TFL), Based on Directed Graph Representation of TFL* (2013)
27. McCall, S.: *Aristotle's Modal Syllogisms*. North-Holland Publishing Company (1963)
28. Moss, L.: Natural logic. In: Lappin, S., Fox, C. (eds.) *The Handbook of Contemporary Semantic Theory*. John Wiley & Sons (2015)
29. Murphree, W. A.: Numerical term logic. *Notre Dame J. Formal Logic*, vol. 39, no. 3, pp. 346–362 (1998) doi: 10.1305/ndjfl/1039182251.
30. Noah, A.: Predicate-functors and the limits of decidability in logic. *Notre Dame J. Formal Logic*, vol. 21, no. 4, pp. 701–707 (1980) doi: 10.1305/ndjfl/1093883255.
31. Priest, G.: *An Introduction to Non-Classical Logic: From If to Is*. Cambridge University Press (2008)
32. Quine, W. V. O.: Predicate functor logic. In: Fenstad, J. E. (ed.) *Proceedings of the Second Scandinavian Logic Symposium*. North-Holland (1971)
33. Rini, A. A.: Is there a modal syllogistic? *Notre Dame J. Formal Logic*, vol. 39, no. 4, pp. 554–572 (1998) doi: 10.1305/ndjfl/1039118870.
34. Russell, B.: *A Critical Exposition of the Philosophy of Leibniz: With an Appendix of Leading Passages*. Cambridge University Press
35. Sommers, F.: On a fregean dogma. In: Lakatos, I. (ed.) *Problems in the Philosophy of Mathematics*, vol. 47, pp. 47–81. Elsevier (1967)
36. Sommers, F.: *The Logic of Natural Language*. Clarendon Press; Oxford: New York: Oxford University Press (1982)
37. Sommers, F.: Intellectual autobiography. In: Oderberg, D. S. (ed.) *The Old New Logic: Essays on the Philosophy of Fred Sommers*, pp. 1–24. Bradford book (2005)
38. Sommers, F., Englebretsen, G.: *An Invitation to Formal Reasoning: The Logic of Terms*. Ashgate (2000)
39. Striker, G.: Assertoric vs modal syllogistic. *Ancient Philosophy*, vol. 14, pp. 39–51 (1994)
40. Szabolcsi, L., Englebretsen, G.: *Numerical Term Logic*. Edwin Mellen Press (2008)
41. Thom, P.: *The Logic of Essentialism: An Interpretation of Aristotle's Modal Syllogistic*. Springer Netherlands (2012)
42. Thom, P.: *Logic and Ontology in the Syllogistic of Robert Kilwardby*. Brill (2007)
43. Thompson, B.: Syllogisms using “few”, “many”, and “most”. *Notre Dame Journal of Formal Logic*, vol. 23, no. 1, pp. 75–84 (1982) doi: 10.1305/ndjfl/1093883568.
44. Thompson, B.: Syllogisms with statistical quantifiers. *Notre Dame Journal of Formal Logic*, vol. 27, no. 1, pp. 93–103 (1986) doi: 10.1305/ndjfl/1093636527.
45. Woods, J.: *Aristotle's Earlier Logic*. College Publications (2014)
46. Woods, J.: Logic Naturalized, pp. 403–432. Springer International Publishing, Cham (2016)

Implementación de modelos de red neuronal profundos en GPUs embebidas para la clasificación de secuencias de fibrilación auricular

Christian García-Aquino¹, Dante Mújica-Vargas¹,
Juan Gabriel González-Serna¹, Manuel Matuz-Cruz²

¹ Tecnológico Nacional de México,
Centro Nacional de Investigación y Desarrollo Tecnológico,
México

² Tecnológico Nacional de México,
México

m21ce012@cenidet.tecnm.mx

Resumen. En este artículo se implementaron arquitecturas basadas en aprendizaje profundo en tarjetas embebidas. La experimentación se realizó en una tarjeta embebida Nvidia Jetson Nano y se emplearon las señales ECG de la base de datos CinC Challenge 2017 de PhysioNet, de las cuales, se extrajeron características de Tiempo-Frecuencia para el entrenamiento de los modelos. Los modelos de aprendizaje profundo que se emplearon para las tareas de clasificación fue una CNN SqueezeNet y una RNN del tipo BiLSTM. Para cuantificar el rendimiento de clasificación de los modelos, se obtuvo la exactitud y el error promedio por cada clase. En donde se sugiere que el modelo convolucional tiene un menor costo computacional, sin embargo, en tareas de clasificación se sugiere que el modelo recurrente tuvo un mayor rendimiento y una mejor propagación del error frente al modelo convolucional.

Palabras clave: Jetson nano, physionet, squeezeNet, BiLSTM, tiempo-frecuencia.

Implementation of Deep Neural Network Models on Embedded GPUs for Classification of Atrial Fibrillation Sequences

Abstract. In this article, architectures based on deep learning in embedded cards were implemented. The experimentation was carried out on an embedded Nvidia Jetson Nano card and ECG signals from the PhysioNet CinC Challenge 2017 database were used, from which Time-Frequency characteristics were extracted for model training.

The deep learning models that were used for the classification tasks were a SqueezeNet CNN and a BiLSTM type RNN. To quantify the classification performance of the models, the accuracy and average error for each class were obtained. Where it is suggested that the convolutional model has a lower computational cost, however, in classification tasks it is suggested that the recursive model had better performance and better error propagation compared to the convolutional model.

Keywords: Jetson nano, physioNet, squeezenet, biLSTM, time-frequency.

1. Introducción

Las Redes Neuronales Artificiales son sistemas de computación basados en un modelo simplificado del cerebro humano. Sus capacidades de clasificación y propiedades como la generalización, tolerancia a fallas y aprendizaje, los hacen atractivos para una basta cantidad de aplicaciones que difícilmente se resuelven con equipos de computo convencionales [1], [2], [3] y [4]. Además, se utilizan en diversos campos del conocimiento [5], incurriendo a la necesidad de sistemas informáticos cada vez más potentes.

Como es bien sabido, obtener estos sistemas informáticos para ejecutar redes neuronales potentes no es barato, por lo que las alternativas económicas son esenciales para el desarrollo y la investigación. En la literatura, existen enfoques sobre tarjetas de bajo costo, mencionando algunos relevantes en relación al trabajo en cuestión. En [6] emplearon modelos previamente entrenados para clasificar pelotas en escenarios, demostrando tiempos de entrenamiento reducidos, mejor precisión y menor sobreajuste a diferencia de un modelo entrenado desde 0³.

En [7] se realizó un sistema de reconocimiento de objetos basado en la red neuronal YOLO, implementado en una tarjeta Jetson TK1, consiguiendo detectar y reconocer objetos a 5 FPS. En [8] se propuso la red de detección de cruces peatonales (CDNet) basada en YOLOv5 para la detección rápida y precisa de cruces peatonales, implementada en el dispositivo Jetson Nano con una velocidad de detección de 33 FPS. En [9], se propuso un sistema de detección de carril basado en redes CNN Encoder-Decoder y LSTM, se implementó en una tarjeta NVIDIA Jetson Xavier NX obteniendo puntuaciones mayores a 90 % en distintas métricas.

Como se puede ver a través de esta sección, se han generado desarrollos empleando tarjetas de desarrollo de bajo costo. Tales dispositivos, han sido de utilidad para la implementación de diferentes modelos neuronales a fin de tener un sistema portátil y de bajo consumo. Sin embargo, en los trabajos mencionados no se puntualiza un análisis del costo computacional en tareas de

³ Desde 0 se refiere a un modelo que no ha sido entrenado en lo absoluto o que no ha recibido una transferencia de aprendizaje.

clasificación, aunado a que no suponen solución alguna orientada a la clasificación de fibrilación auricular (FA). Por lo tanto, el objetivo de esta investigación es implementar redes neuronales en tarjetas embebidas para realizar la clasificación de secuencias de FA, partiendo del empleo de la tarjeta embebida NVIDIA Jetson Nano. Enfocándose en realizar un análisis de rendimiento en los modelos neuronales, que con el uso de características específicas lograr un costo computacional bajo y un alto rendimiento de clasificación.

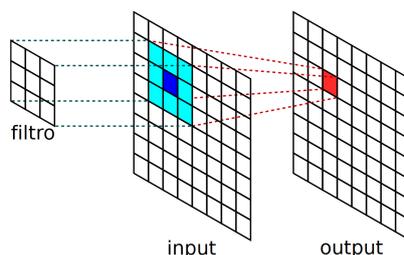


Fig. 1. Proceso de convolución [11].

El resto de este documento está organizado de la siguiente manera. En la sección II, se tiene una breve introducción sobre las Redes Neuronales y las características extraídas de los ECG 's utilizados durante la experimentación. El método de implementación de los modelos de red neuronal se indica en la sección III. Los resultados obtenidos de la experimentación y un análisis comparativo se presentan en la sección IV. Las conclusiones se mencionan en la sección final y se describe el trabajo futuro.

2. Marco teórico

2.1. Redes neuronales

Redes neuronales convolucionales. Las Redes Neuronales Convolucionales (CNN) son, en esencia, redes neuronales que emplean la operación de convolución como una de sus capas. Las CNN se han aplicado a problemas en los que los datos de entrada sobre los que se van a realizar las predicciones tienen una cuadrícula conocida como una serie temporal o una imagen [10]. En el caso de las imágenes, la operación de convolución se da intuitivamente por una entrada $I(m, n)$ y un Kernel $K(a, b)$ la operación de convolución se expresa como:

$$\text{Conv}(t) = \sum_a \sum_b I(a, b) * K(m - a, n - b), \quad (1)$$

donde m, n y a, b son las dimensiones de una imagen y del kernel de convolución respectivamente. Apreciándose gráficamente el proceso interno que realiza la capa de convolución de una CNN en la Figura 1.

Redes neuronales recurrentes. Las Redes Neuronales Recurrentes (RNN) se emplean para analizar datos que cambian con el tiempo [12]. Para permitir a la red memorizar y acceder a los historiales de entrada, se introducen las conexiones recurrentes para predecir el paso de tiempo actual y transfiriendo esa predicción al siguiente paso como una entrada.

De acuerdo con la Figura 2, un modelo RNN tiene la misma estructura que los modelos clásicos de RNA's, contando con una capa de entrada, n capas ocultas y una de salida, sin olvidar el parámetro t correspondiente al tiempo, siendo x_{t-1} , x_t y x_{t+1} las entradas del modelo RNN en diferentes instantes de tiempo.

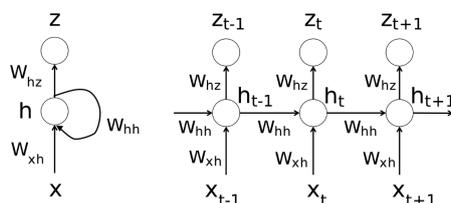


Fig. 2. Estructura básica de una RNN [12].

En la forma más básica de una RNN, se encuentra su función de aprendizaje en la ecuación 2 para las capas ocultas y en la ecuación 3 para la capa de salida:

$$h[t] = f(W_x^h(x[t] + b_h) + W_h^h(h[t - 1] + b_h)), \quad (2)$$

$$y[t] = g(W_h^o(h[t] + b_o)). \quad (3)$$

2.2. Transformada de Fourier de tiempo corto

La Transformada de Fourier de Tiempo Corto (STFT) se encarga de analizar señales no estacionarias a través de la Transformada de Fourier (TF). En donde la STFT consiste en dividir la señal en pequeños segmentos de tiempo de tal manera que se pueda asumir que para cada segmento la señal es estacionaria, y así calcular la TF en cada porción de la señal, la cuál es tomada como una ventana que se desliza a lo largo del eje del tiempo, resultando en una representación de dos dimensiones de la señal [13]. Matemáticamente, se escribe como:

$$STFTx(t) = X(\tau, \omega) = \int_{-\infty}^{\infty} x(t)w(t - \tau) e^{-j\omega t} dt, \quad (4)$$

donde $w(t)$ es la función ventana de tipo Hann o colina Gaussiana inicializada en 0 y $x(t)$ es la señal de entrada a transformar, $X(\tau, \omega)$ es en esencia la TF de $x(t)$, $w(t - \tau)$ es una función compleja que representa la fase y la magnitud de la señal sobre el tiempo y la frecuencia.

Concurrentemente la fase instantánea es empleada junto con el eje del tiempo τ y el eje de la frecuencia ω para suprimir cualquier discontinuidad por salto en

la fase resultante en la STFT. El índice de tiempo τ es normalmente considerado como un tiempo “lento” y usualmente no se expresa con tan alta resolución como con el tiempo t .

2.3. Transformada wavelet continua

En [14] se introdujo la Transformada Wavelet Continua (CWT) como una técnica alternativa a la STFT para suprimir el problema de resolución ocasionado por la yuxtaposición de los datos. En relación a eso, la CWT se realiza de manera comparable al análisis STFT, debido a que la señal es multiplicada por una función, en este contexto denominada “Wavelet”, siendo análoga a la función de ventaneo de la STFT, y la transformada se calcula separadamente para distintos segmentos de la señal en el dominio del tiempo [15]. Escribiendo la CWT como se describe en la Fig. 3.

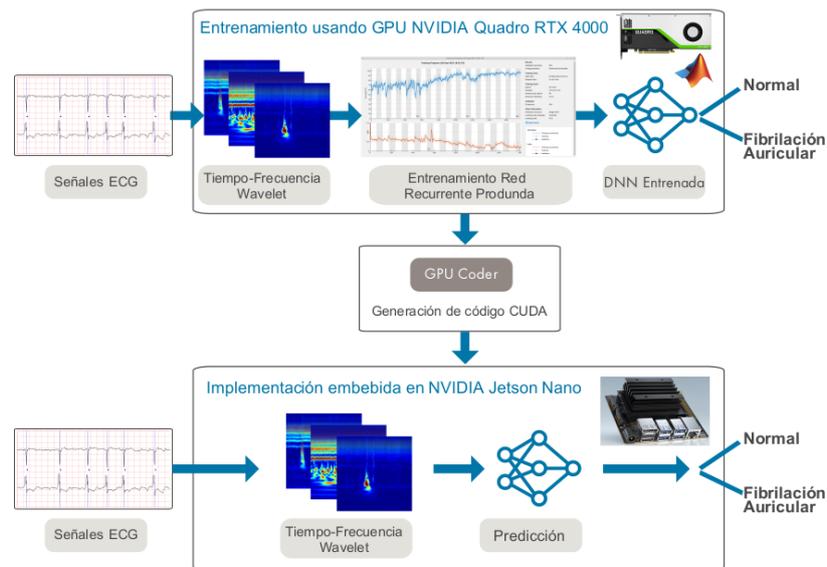


Fig. 3. Implementación.

En donde $f(t)$ es la señal a la cual se aplica la transformada, $\Psi_{u,s}^*(t)$ es la asociación compleja de la función Wavelet $\Psi_{u,s}(t)$ y la señal transformada W , es una función de dos variables, siendo u y s , los parámetros de tiempo y escala respectivamente, que irán cambiando su valor conforme a las iteraciones dentro del análisis:

$$W f(u, s) = \int_{-\infty}^{+\infty} f(t) \Psi_{u,s}^*(t) dt. \quad (5)$$

3. Implementación

Es conveniente analizar el proceso de implementación de los modelos de aprendizaje profundo para la clasificación de las secuencias de FA. Comenzando por la preparación de las señales ECG con las que posteriormente se extraerán las características de Tiempo-Frecuencia tanto para el modelo recurrente como para el modelo convolucional. Asimismo, se realiza el entrenamiento de los modelos con dichas características y una vez entrenadas, embeber los modelos en una tarjeta electrónica de alto rendimiento para realizar las experimentaciones. Una descripción de lo anterior puede apreciarse en la Figura 3.

3.1. Señales ECG

Para esta investigación se empleó la base de datos CinC 2017 de PhysioNet, el cual contiene señales de ECG muestreadas a 300 Hz, organizadas en cuatro clases diferentes: Normal (N), AFib (A), Otro ritmo (O) y Grabación ruidosa, mismos que se consideraron las primeras dos clases para realizar experimentos de clasificación. Se pueden ver más detalles del conjunto de datos en [16].

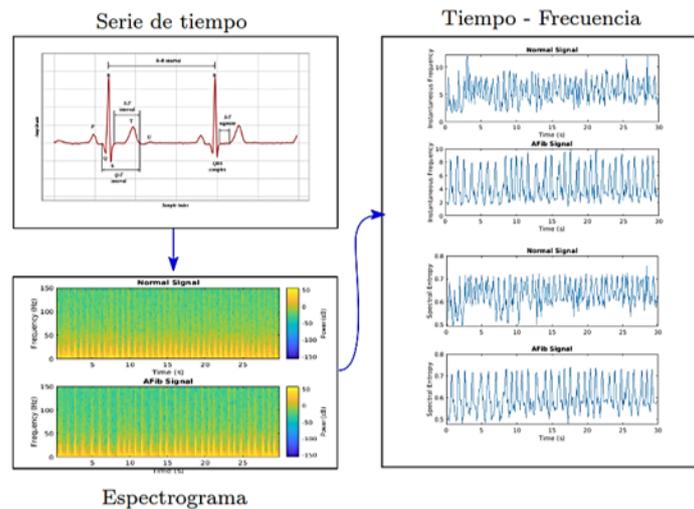


Fig. 4. Extracción de momentos tiempo-frecuencia.

3.2. Entrenamiento de los modelos basados en aprendizaje profundo

Momentos de tiempo-frecuencia. A fin de incrementar el rendimiento de la arquitectura neuronal, es conveniente utilizar características con mayor discriminación. En el área del Procesamiento Digital de Señales (DSP), es común el análisis de éstas en los dominios de frecuencia y tiempo-frecuencia; para ello se deben graficar los espectrogramas de los ECG de latido normal y con FA.

En la Figura 4, se muestran los espectros para señales de muestras en la parte inferior izquierda.

Para calcular el espectrograma, se considera la resolución espectral con la longitud total de la señal. Si es posible, se debe calcular un periodograma de toda la señal utilizando una ventana de Kaiser. De no ser posible, se debe calcular un solo periodograma modificado en un período de tiempo razonable, la función debe calcular un periodograma de Welch dividiendo la señal en segmentos superpuestos, tomando muestras de cada segmento mediante una ventana de Kaiser y promediando los periodogramas de los segmentos.

Una vez obtenidos los espectrogramas, se extraen los momentos de Tiempo-Frecuencia conocidos como la Frecuencia Instantánea y la Entropía Espectral. Empleando el primero para estimar la frecuencia dependiente del tiempo de una señal. La función calcula un espectrograma usando TF de corta duración a lo largo de 255 ventanas de tiempo y las salidas corresponden a los centros de las mismas.

El segundo momento es la Entropía Espectral encargada de medir qué tan plano es el espectro de una señal. Una señal con un espectro puntiagudo, como una suma de sinusoides, tiene una entropía espectral baja. Una señal con un espectro plano, como el ruido blanco, tiene una alta entropía espectral.

Modelo BiLSTM. Por otra parte, se empleó una Red Long Short-Term Memory (LSTM), adecuada para estudiar secuencias de datos y así aprender las dependencias a largo plazo.

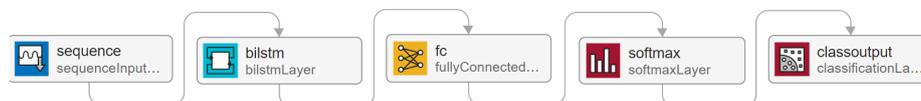


Fig. 5. Red BiLSTM.

La capa LSTM puede mirar la secuencia de tiempo en la dirección de avance, mientras que la capa de LSTM-Bidireccional puede mirar tanto en la dirección de avance como en la de retroceso. En este sentido, la arquitectura neuronal utiliza una capa LSTM-Bidireccional.

Esta arquitectura se puede reinterpretar como un modelo híbrido, ya que considera el procesamiento de una serie de tiempo a través de una red recurrente y una fase de clasificación dada por una red totalmente conectada del tipo Perceptrón Multicapa, con una capa final del tipo Softmax para clasificación multiclase, pudiendo observar la arquitectura completa en la Figura 5.

Representaciones de tiempo-frecuencia. Para las tareas de clasificación de señales ECG empleando CNN, se debe hacer una serie de adecuaciones. En contraste con la arquitectura BiLSTM, no es posible trabajar con series de tiempo.

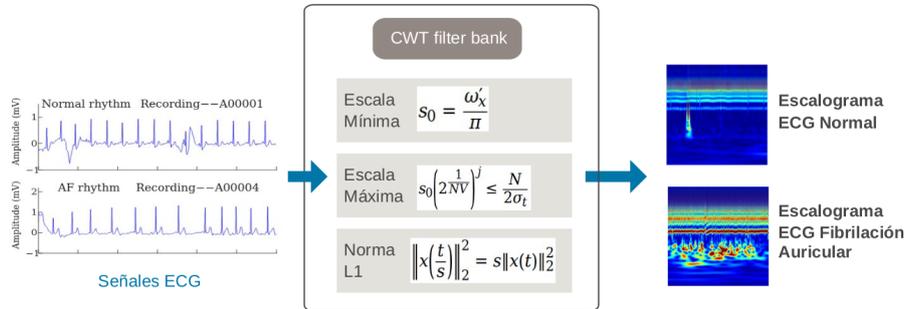


Fig. 6. Extracción de escalogramas.

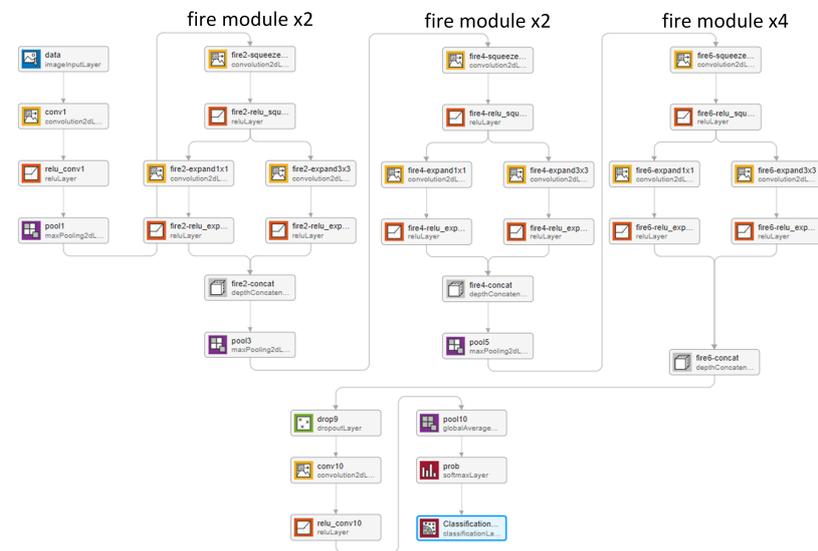


Fig. 7. Modelo SqueezeNet.

Por lo tanto, las señales ECG deben ser transformadas en una representación 2D, utilizando Wavelets, en particular Bancos de Filtros (FB) de Transformación Wavelets Continuas (CWT).

El precálculo del (FB) es el método preferido cuando se obtiene el CWT de muchas señales usando los mismos parámetros, considerando varias series de tiempo ECG. Las escalas mínima y máxima se determinan automáticamente en función de la propagación de energía de la wavelet. Si x tiene un valor real, WT es una matriz 2D donde cada lado corresponde a una escala. El tamaño de la columna de WT es igual a la longitud de x .

Si x tiene un valor complejo, WT es una matriz 3-D, donde la primera página es el CWT para las escalas positivas y la segunda página es el CWT para las

escalas negativas. La función CWT usa la normalización L1, con lo cual, si se tienen componentes oscilatorios de igual amplitud en sus datos a diferentes escalas, se tendrán la misma magnitud en el CWT.

Para ejemplificar esta transformación se puede observar la Figura 6. La imagen permite ilustrar los procesos que se realizan con una serie de tiempo sana y otra con la presencia de fibrilación auricular.

Modelo SqueezeNet. SqueezeNet (Fig. 7) es una CNN profunda originalmente diseñada para clasificar imágenes en 1000 categorías. En el presente trabajo de investigación, se reutilizó esta arquitectura de red CNN para clasificar las señales de ECG en función de las imágenes del CWT de los datos de series de tiempo. Cada capa de la arquitectura de una CNN se considera un filtro.

Las capas iniciales identifican características más comunes de las imágenes, como manchas, bordes y colores. Las capas posteriores se centran en características más específicas para diferenciar categorías. SqueezeNet está capacitada para clasificar imágenes en 1000 categorías de objetos. Para utilizar esta red en el problema de clasificación de señales ECG, resulta obligatorio volver a entrenar la red.

En las CNN se puede explotar el concepto de transferencia de conocimiento, lo cual sugiere que para reentrenar una red en otra tarea de clasificación basta con ajustar las últimas capas (fase de predicción), ya que las capas de convolución sirven para extraer características, y éstas son genéricas para cualquier tipo de imagen. Por lo que se analizaron las últimas 6 capas y se realizó la modificación de la capa “conv10” de 1×1 con una nueva capa convolucional con el número de filtros proporcionalmente al número de clases con los que se trabajó.

Entrenamiento de los modelos. Dado que las arquitecturas basadas en Aprendizaje Profundo requieren grandes cantidades de información en la fase de entrenamiento, el tiempo requerido para esta tarea puede requerir una gran cantidad de horas. Fue necesario solventar esta problemática con tarjetas aceleradoras, para este proyecto se empleó una GPU Quadro RTX 4000 con 2304 núcleos CUDA. Con este dispositivo se realizaron las fases de entrenamiento y validación de las redes.

En donde el modelo BiLSTM requirió un tiempo de entrenamiento de 2 minutos con un total de 30 épocas y 1770 iteraciones, mientras que el modelo convolucional requirió un tiempo de entrenamiento de 41 segundos con un total de 40 épocas y 320 iteraciones.

Implementación en la tarjeta embebida. Para comunicarse con el hardware embebido de NVIDIA, se creó un objeto de conexión de hardware. Posterior a la creación de la conexión de la tarjeta embebida con MATLAB, se hizo uso de GPU Coder para la generación de código CUDA. Finalmente, una vez que el procedimiento fue exitoso, todos los archivos generados son copiados en la tarjeta electrónica, resaltando el archivo con extensión elf, ya que es el ejecutable de todo el sistema de clasificación desarrollado.

Tabla 1. Exactitud obtenida por clase y modelo.

Muestra	ecgN RNN	ecgAF RNN	ecgN CNN	ecgAF CNN
1	0.910	0.851	0.893	0.848
2	0.893	0.997	0.869	0.979
3	0.943	0.959	0.906	0.952
4	0.920	0.935	0.901	0.934
5	0.975	0.961	0.907	0.923
6	0.923	0.912	0.908	0.895
7	0.889	0.918	0.871	0.880
8	0.899	0.994	0.886	0.963
9	0.905	0.854	0.887	0.854
10	0.942	0.895	0.907	0.868
11	0.897	0.904	0.872	0.876
12	0.957	0.971	0.884	0.945
13	0.932	0.872	0.897	0.850
14	0.904	0.837	0.889	0.828
15	0.967	0.869	0.895	0.838
16	0.989	0.845	0.911	0.835
17	0.938	0.878	0.903	0.863
18	0.917	0.920	0.909	0.885
19	0.926	0.968	0.905	0.933
20	0.935	0.913	0.903	0.897
Promedio	0.928	0.913	0.895	0.892

4. Experimentación y resultados

En este experimento se evaluó el rendimiento de ambos modelos neuronales en términos de la métrica de Exactitud, así como del Error en la predicción de las clases. Puesto que los modelos fueron adecuados para una clasificación binaria, solo se consideran las clases: a) Ritmo Normal (N) y b) Ritmo con FA (AF). Considerando 20 señales de muestra para cada clase. Para lo cuál, se asumen las siguientes nomenclaturas:

- **ecgN_RNN y ecgAF_RNN:** Representa la predicción de la Red Neuronal Recurrente entrenada con los momentos de Tiempo-Frecuencia para la clase ritmo cardíaco normal y fibrilación auricular.
- **ecgN_CNN y ecgAF_CNN:** Representa la predicción de la Red Neuronal Convolutiva entrenada con los escalogramas de Tiempo-Frecuencia para la clase ritmo cardíaco normal y fibrilación auricular.

En la Tabla 1 se presenta de manera resumida la exactitud de predicción de ambos modelos neuronales. Se destaca una eficiencia promedio de la red neuronal recurrente del 92.8% para la clase ritmo cardíaco normal y un 91.3% para la clase ritmo cardíaco con fibrilación auricular. Mientras que para el modelo convolutivo un 89.5% y 89.2%, respectivamente.

Gráficamente estos resultados pueden ser visualizados en la Figura 8. Es posible observar aspectos como los valores promedio, mínimo, máximo, así como la dispersión de los resultados. De manera rápida se puede visualizar que ambos modelos neuronales presentaron una menor varianza en los resultados para la clase ritmo cardíaco sano, en comparación del ritmo cardíaco con fibrilación auricular.

De manera complementaria, en la Tabla 2 se presenta el Error en la predicción de los modelos neuronales profundos. Los valores promedio sugieren una menor

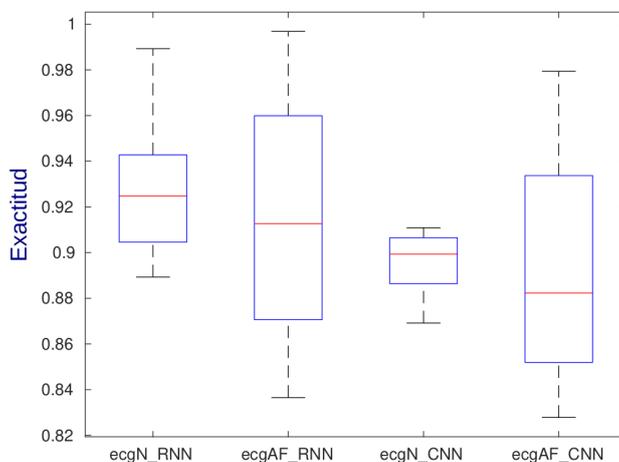


Fig. 8. Exactitud promedio por clase y modelo.

Tabla 2. Error obtenido por clase y modelo.

Muestra	ecgN RNN	ecgAF RNN	ecgN CNN	ecgAF CNN
1	0.090	0.149	0.107	0.152
2	0.107	0.003	0.131	0.021
3	0.057	0.041	0.094	0.048
4	0.080	0.065	0.099	0.066
5	0.025	0.039	0.093	0.077
6	0.077	0.088	0.092	0.105
7	0.111	0.082	0.129	0.120
8	0.101	0.006	0.114	0.037
9	0.095	0.146	0.113	0.146
10	0.058	0.105	0.093	0.132
11	0.103	0.096	0.128	0.124
12	0.043	0.029	0.116	0.055
13	0.068	0.128	0.103	0.150
14	0.096	0.163	0.111	0.172
15	0.033	0.131	0.105	0.162
16	0.011	0.155	0.089	0.165
17	0.062	0.122	0.097	0.137
18	0.083	0.080	0.091	0.115
19	0.074	0.032	0.095	0.067
20	0.065	0.087	0.097	0.103
Promedio	0.072	0.087	0.105	0.108

de predicción realizada por la RNN, en comparación con el modelo convolucional. De igual manera, las tendencias gráficas se pueden visualizar en la Figura 9.

Tiempo de procesamiento. El segundo experimento nos permitió conocer el tiempo de procesamiento promedio requerido en la fase de predicción de ambos modelos, a partir de las 20 instancias de muestra. Se consideraron 3 dispositivos distintos:

- **CPU** Intel Core i7-8750 con 12 núcleos a 4.1GHz.
- **GPU** Nvidia Quadro RTX 4000 con 2304 núcleos CUDA.
- **Tarjeta Embebida** Nvidia Jetson Nano con 128 núcleos CUDA.

Tabla 3. Costo computacional.

Dispositivo	Red Neuronal Recurrente	Red Neuronal Convolutacional
CPU	0.769 s	0.128 s
GPU	0.061 s	0.010 s
Tarjeta Embebida	21.487 s	15.093 s

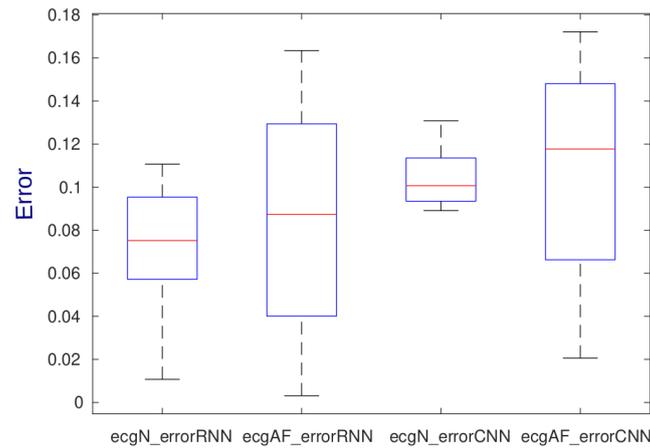


Fig. 9. Error promedio por clase y modelo.

En la Tabla 3 se presenta de manera resumida el costo computacional medido en segundos de los 3 dispositivos utilizados en la experimentación, en donde la información es indiferente a una clase en particular. El modelo recurrente requiere mayor tiempo de procesamiento, independientemente del dispositivo utilizado. Esto se debe a que este tipo de arquitecturas trabaja con información en el tiempo, lo cual es un aspecto importante, ya que permite garantizar un mejor rendimiento.

Por ende, fue de interés para esta investigación saber cuál era el tiempo de procesamiento de estos modelos neuronales en un dispositivo embebido de alto rendimiento. Pudiendo observar de la tabla que en ambos casos es superior a los 15 segundos para realizar una predicción, este tiempo se considera aceptable por la cantidad de procesamiento que se realiza en esta fase.

5. Conclusiones

Los resultados obtenidos en este experimento resaltan aspectos que influyen en el rendimiento, ya que emplear series de tiempo en crudo no garantiza un rendimiento de predicción aceptable, lo cual no es útil al realizar prototipos de manera física. Por ende, el uso del procesamiento digital de señales es esencial en las tareas del procesamiento de series de tiempo de las señales ECG.

Como se observó, se consideraron dos variantes de modelos neuronales. Para la arquitectura recurrente, se obtuvieron espectrogramas de frecuencia para ambos tipos de señales; sin embargo, estas eran imágenes las cuales no podían ser utilizadas por la red recurrente, así que a partir de los espectrogramas se obtuvieron los momentos Tiempo-Frecuencia (Frecuencia Instantánea y Entropía Espectral).

Estas características son series de tiempo que concentran la información más importantes de las señales ECG y permitieron acelerar el entrenamiento de la red recurrente, e incrementar su rendimiento a más del 92%. Por otra parte, para la arquitectura convolucional bastó con transformar la señales ECG en Escalogramas de Frecuencia, esta transformación 1D en 2D permitió utilizar esta característica para hacer el entrenamiento de la red. Garantizando un rendimiento superior al 89%.

Asumiendo que se podría obtener un mejor rendimiento si se recurrieran a técnicas como el aumento información, para mejorar su rendimiento, aunado a un mayor estudio de características significativas posibles que se puedan extraer de datos crudos de los ECG. Como trabajo futuro se tiene la tarea de mejorar los tiempos de procesamiento a fin de concebir un sistema que tenga la capacidad de trabajar los más cercano al tiempo real, de manera portátil y de manera remota.

Agradecimientos. Este trabajo fue apoyado por el Tecnológico Nacional de México/CENIDET a través del proyecto “Delimitación de masas sólidas malignas en mamografías mediante un algoritmo de nodos conectados con el menor ángulo polar”, así como por el CONACYT.

Referencias

1. Curran, K., Li, X., McCaughley, N.: Neural network face detection. *The Imaging Science Journal*, vol. 53, no. 2, pp. 105–115 (2005)
2. Patil, V., Shimpi, S.: Handwritten english character recognition using neural network. *Elixir Comput Sci Eng*, vol. 41, pp. 5587–5591 (2011)
3. Bartlett, P. L.: The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. *IEEE transactions on Information Theory*, vol. 44, no. 2, pp. 525–536 (1998)
4. Åkesson, B. M., Toivonen, H. T.: A neural network model predictive controller. *Journal of Process Control*, vol. 16, no. 9, pp. 937–946 (2006)
5. Timotheou, S.: The random neural network: A survey. *The computer journal*, vol. 53, no. 3, pp. 251–267 (2010)
6. Holz, D., Genter, K., Saad, M.: RoboCup 2018: Robot world cup XXII. *Lecture Notes in Computer Science* (2019)
7. Reyes, E., Gómez, C., Norambuena, E., Ruiz-del-Solar, J.: Near real-time object recognition for pepper based on deep neural networks running on a backpack. In: Holz, D., Genter, K., Saad, M., von Stryk, RoboCup 2018: Robot World Cup XXII. *RoboCup 2018. Lecture Notes in Computer Science*, vol 11374. Springer, Cham (2019)

8. Zhang, Z. D., Tan, M. L., Lan, Z. C., et al.: CDNet: A real-time and robust crosswalk detection network on Jetson nano based on YOLOv5. *Neural Comput and Applic* (2022)
9. Kortli, Y., Gabsi, S., Lew, F. C., Voon, L. Y., Jridi, M., Merzougui, M., Atri, M.: Deep embedded hybrid CNN–LSTM network for lane detection on NVIDIA Jetson Xavier NX. *Knowledge-Based Systems*, vol. 240 (2022)
10. Ketkar, N.: Convolutional neural networks. *Deep Learning with Python*. pp. 63–78 (2017)
11. Redolfi, J.: *Aplicación en agricultura de precisión de esquemas actuales de reconocimiento visual* (2018)
12. Vogt, N.: CNNs, LSTMs, and attention networks for pathology detection in medical data (2019)
13. Smith, J. O.: *Mathematics of the discrete Fourier transform (DFT)*. W3K Publishing (2007)
14. Morlet, J., Arens, G., Fourgeau, E., Giard, D.: Wave propagation and sampling theory—part II: sampling theory and complex waves. *GEOPHYSICS*, vol. 47, no. 2, pp. 222–236 (1982)
15. Bolós, V.J., Benítez, R.: The wavelet scalogram in the study of time series. In: *Advances in Differential Equations and Applications*. SEMA SIMAI Springer Series, vol. 4. Springer, Cham (2014)
16. Clifford, G. D., Liu, C., Moody, B., Li-wei, H. L., Silva, I., Li, Q., Johnson, A. E., Mark R. G.: AF classification from a short single lead ECG recording: The PhysioNet/computing in cardiology challenge 2017. In: *2017 Computing in Cardiology (CinC)* pp. 1–4 (2017)

Uso de técnicas de Web Scraping para obtención automática de bases de datos en la Web

Rogelio Mijangos-Espinosa, Alicia Martínez-Rebollar,
Hugo Estrada-Esquivel, Yasmín Hernández-Pérez

¹Tecnológico Nacional de México
Departamento de Ciencias Computacionales,
México

{m20ce066, alicia.mr, hugo.ee, yasmin.hp}@cenidet.tecnm.mx

Resumen. Actualmente, la Web es el canal de comunicación más utilizado a nivel mundial. Millones de usuarios utilizan esta red para transmitir sus datos a diferentes personas, empresas, dependencias gubernamentales, instituciones educativas y de investigación. La Web permite la distribución de información de manera continua e ilimitada, lo cual implica que ésta requiera ser localizada, recolectada, revisada para poder ser utilizada con fines de investigación. Sin embargo, este enorme flujo de datos que se produce cada día en la Web hace muy lenta y complicada la captura, oportuna y sistemática, de la información que se produce en forma periódica, así como la identificación de nueva información colocada en la Web. En este sentido se requiere de sistemas a los cuales se pueda delegar las tareas de búsqueda y recolección de información. En este artículo se presenta una herramienta software que implementa técnicas de *Web Scraping* para la búsqueda, selección y descarga de bases de datos alojadas en la web, creando un compendio de datos que puede ser utilizado por técnicas de análisis de información. El sistema propuesto se ha evaluado con los procesos de búsqueda, descarga y almacenamiento de información sobre el COVID-19 en México.

Palabras clave: Web scraping, extracción de información, big data, COVID-19.

Use of Web Scraping Techniques for Automatic Capturing of Databases Located in the Web

Abstract. Currently, the Web is the most widely used communication channel worldwide. Millions of people use this network to transmit their data to different people, companies, government agencies, educational and, and also research institutions. The Web enables the unlimited and continuous distribution of

information, which implies that it needs to be located, collected, reviewed in order to be used for research purposes. However, this enormous flow of data that is produced every day on the Web makes it very difficult the timely and systematic capturing of information that is produced periodically, as well as the identification of new information published on the Web. In this context, software systems are needed to which the tasks of searching and collecting information can be delegated. This paper presents a software tool that implements Web Scraping techniques for searching, selecting and downloading databases located on the Web, which enables to create a data repository that can be used by information analysis techniques. The proposed system has been evaluated with the processes of searching, downloading and storing of information about the COVID-19 in Mexico.

Keywords. Web scraping, information extraction, big data, COVID-19.

1. Introducción

En la actualidad la Web representa la fuente de información más utilizada en el ámbito académico, científico e industrial. La cantidad de información contenida en la Web es impresionante, y comprende desde documentación formal y estructurada de muy diversos temas, hasta contenidos informales generados directamente por usuarios a través de medios sociales. De esta forma la cantidad de información almacenada en la Web crece día con día a medida que más usuarios cuentan con mecanismos de acceso a internet.

De esta manera, la Web se ha convertido en la principal fuente de información para las personas que realizan investigaciones que requieren información actualizada. La recolección de información de forma oportuna, correcta y completa requiere de algoritmos inteligentes que ayuden en el proceso automático de captura y almacenamiento de información.

Uno de los ejemplos más notables de la necesidad de contar con mecanismos de ayuda para el proceso de captura de información es la actual pandemia de COVID-19, la cual da origen a una enorme cantidad de información difundida y actualizada constantemente por dependencias de gobierno, instituciones públicas e instituciones de salud de todo el mundo [1]. La recuperación de información en la Web no es algo sencillo, especialmente cuando se tiene información con características Big data, es decir, con grandes cúmulos de información de muy diferentes fuentes de datos.

Las cantidades de datos masivos que se generan en la Web, el constante crecimiento y la actualización (muchas veces en tiempo real) vuelve complicada la tarea de obtener buena información estructurada en los tiempos para realizar análisis con oportunidad. Esta situación obliga la revisión constante de los contenidos de sitios Web para mantener la información actualizada.

Sin embargo, la revisión de múltiples sitios Web con técnicas convencionales merma la cantidad de información que se puede recuperar manualmente [2]. Este artículo presenta un sistema de recuperación de información que automatiza la recolección de

información de archivos localizados en la Web. El sistema propuesto utiliza técnicas de Scraping para rastrear y almacenar información contenida en bases de datos disponibles en sitios Web de forma pública.

Las descargas de información tienen como punto de inicio un grupo de direcciones URL y rutas XPath que permite rastrear la información dentro de la página Web. Una vez localizada la información requerida, ésta es recuperada y almacenada en una base de datos local, para estar disponible para el usuario.

Este artículo se encuentra organizado de la siguiente manera, la Sección 2 presenta el marco teórico y los trabajos relacionados. La Sección 3, muestra una descripción de la propuesta del sistema de recuperación. La Sección 4, presenta el algoritmo de recuperación de información. La Sección 5, describe las pruebas realizadas al algoritmo de recuperación de información. La Sección 6 presenta la discusión. Finalmente, la Sección 7 presenta las conclusiones y trabajos futuros.

2. Trabajos relacionados

Las técnicas de Web Scraping se utilizan para extraer información de sitios Web de manera automática mediante el análisis y manipulación de la estructura HTML. El Web Scraping rastrea, recupera y estructura información incrustada en etiquetas HTML de las páginas Web. Esto es una tarea compleja debido a los diferentes tipos de datos que se encuentran en los sitios Web [3].

La recuperación y almacenamiento unificado de los contenidos de diferentes sitios Web sobre un mismo dominio permite al usuario un rápido acceso a la información requerida, dando una mayor ventaja a la obtención de conocimiento encontrado en la Web. [4]. La búsqueda de trabajos relacionados permitió analizar los siguientes trabajos de investigación: El trabajo presentado por Zhao [5] muestra la descripción básica de un sistema de *Web Scraping*, mostrando el flujo de los procesos, la utilidad de las tareas y las áreas de aplicación de un *Scraping*.

El documento se divide en dos partes, la primera muestra la forma de realizar conexiones con diferentes fuentes de información. La segunda parte describe la forma de extraer información de estas fuentes. En el trabajo de Zhao se utilizaron herramientas de apoyo como Urllib2 y Selenium para realizar conexiones estables. Los datos encontrados pueden ser visualizados de forma previa a la extracción y la extracción y análisis de los datos se realiza de forma local.

El artículo puntualiza los problemas legales, derechos de autor, condiciones de servicio como restricciones y limitaciones en un Web Scraping. Rodríguez [6] realizó un análisis de 18 sistemas que ocupaban técnicas Web Scraping y que se encontraban en el mercado en el año 2019. Los sistemas fueron comparados de acuerdo con los servicios que cada uno ofrecía y los clasificaron en tres tipos: recursivos, buscadores, e-commerce. Rodríguez aplicó librerías de Java para scrapers recursivos.

Los formatos JSON se utilizaron para la transferencia de archivos, realizando pruebas con estas características se obtuvieron aproximadamente 100 mil resultados. El autor aplicó patrones de análisis al tipo e-commerce permitiéndoles extraer los precios de productos y servicios. El trabajo de investigación realizado por Uzun [7] analiza la

forma tradicional de hacer *Web Scraping* presentando diferentes aspectos como tiempo, metodología, accesos, extracciones, entre otros aspectos.

Uzun describe el diseño e implementación de un nuevo enfoque para *Web Scraping* denominado UzunExt, en el cual los métodos en cadena son la base para la extracción de datos mejorando los tiempos de respuesta. El enfoque propuesto es explicado en dos partes: En la primera parte se presenta las acciones de rastreo y extracción de datos. En la segunda parte, el autor aplica un análisis y procesamiento a los datos antes de ser almacenados. Uzun realizó experimentos de funcionalidad a una muestra de 100 sitios Web que contenían datos en diferentes idiomas y formatos.

Los resultados fueron favorables al utilizar el enfoque propuesto obteniendo 300 páginas y 4 patrones. El trabajo de investigación de Ram Sharan [8] presenta un análisis de aplicaciones tradicionales y aplicaciones basadas en la nube. El artículo establece que algunos de los problemas a los cuales se enfrentan los sistemas de *Web Scraping* son: restricciones de acceso, captcha, volúmenes enormes en los archivos, distribución heterogénea y la fiabilidad de los datos recuperados. El autor propuso la utilización de servicios Web proporcionados por Amazon, Elastic compute cloud y DynamoDB.

También se utilizaron los servicios Web que ofrece Selenium para el control del sistema de *Web Scraping*. Las pruebas las realizaron en 50 sitios Web pertenecientes a Amazon, Google shopping y otros sitios de comercio, desplegados en instancias de 20 y 30 máquinas utilizando un nodo o un nodo con varios hilos. Los sistemas basados en la nube arrojaron mejores resultados mostrando datos confiables, una reducción de costos significativa y mantenimientos más económicos.

El trabajo presentado por Glez-Peña [9] aborda la revisión de las fortalezas y limitaciones de las herramientas *Web Scraping* encontrando que los problemas de conexión con las bases de datos, la denegación de servicios Web y la demanda de datos masivos son algunas de las problemáticas comunes de estos sistemas. Glez-Peña presenta tres aspectos importantes para el análisis de las herramientas: el tiempo, la capacidad y la comunicación. Las librerías libcurl, Apache HttpClient, Jsoup, htmlcleaner, BeautifulSoup, scrapy, Web-Harvet se seleccionaron como herramientas de soporte debido a la compatibilidad que ofrecían.

3. Propuesta de un sistema de recuperación de información

Las técnicas de *Web Scraping* se utilizan para extraer información de sitios Web de manera automática mediante el análisis y manipulación de la estructura HTML. El *Web Scraping* rastrea, recupera y estructura información incrustada en etiquetas HTML de las páginas Web, lo cual es complejo [3] debido a los diferentes tipos de datos que se encuentran en los sitios Web.

La descarga y estructuración de estos contenidos da una comprensión mayor y una ventaja en la obtención de conocimiento de valor tomando en cuenta que los contenidos tienen volúmenes altos, actualizaciones veloces y una variedad de contenidos que demandan soluciones eficientes para su procesamiento. En este artículo presentamos nuestra propuesta de un sistema para recuperación de información (*Web Scraping*), el cual está organizado en 4 etapas (Figura 1):

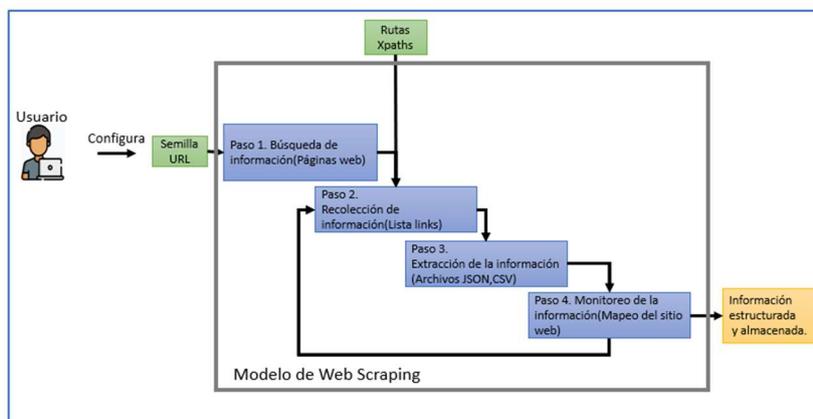


Fig. 1. Vista general del método propuesto.

Tabla 1. Ejemplo de configuración del Proyecto COVID CONACYT.

Nombre Proyecto	Dominio de búsqueda	URL semilla	Ruta XPath
COVID	datos.covid-	https://datos.covid-	//*[@id="bPvM1"]/a/span [2]
CONACYT	19.conacyt.mx	19.conacyt.mx/#DownZCSV	//*[@id="bPvM2"]/a/span [2]

- Búsqueda de información.
- Recolección de información del sitio Web.
- Descarga de información del sitio Web.
- Monitoreo de información del sitio Web.

Con el objetivo de validar nuestra propuesta hemos elegido un caso de estudio para la recuperación de información para COVID-19, la cual es una enfermedad respiratoria que se ha propagado por todo el mundo desde el 2019. Las dependencias de salud han sido los encargados de publicar la cantidad de casos confirmados, casos de defunciones, casos recuperados y la ubicación de estos de manera constante a través de los diferentes medios y canales de transmisión de información [10].

Uno de los medios más utilizados para compartir información del COVID-19 son las páginas Web de carácter gubernamental y de los servicios de salud, creando así múltiples fuentes de información que pueden ser consultados por investigadores para realizar diferentes estudios científicos referentes al COVID-19. El sistema de información propuesto permite la recolección de bases de datos publicadas en la Web que contienen información sobre el COVID-19 en México.

La información que se busca recolectar son casos positivos, casos de defunciones, casos negativos, casos sospechosos y la ubicación de estos índices, con el fin de generar un conjunto de datos unificado que pueda ponerse a disposición de investigadores que lo requieran para efectos de análisis de la información. Las fuentes de información

utilizadas para este caso de estudio son sitios de dependencias gubernamentales, así como de dependencias de salud, las cuales se listan a continuación:

- IMSS: <https://coronavirus.gob.mx/>.
- INSABI: <https://datos.gob.mx> .
- CONACYT: <https://datos.covid-19.conacyt.mx>.
- UNAM: <https://covid19.ciga.unam.mx/>.
- Instituto Politécnico Nacional: <https://www.ipn.mx> .
- Universidad de Hopkins: <https://data.humdata.org>.
- Apple: <https://covid19.apple.com>.

4. Algoritmo de recuperación de información

El algoritmo de recuperación de información tiene como objetivo la búsqueda y recuperación de información de bases de datos almacenados en páginas Web. La descarga de los datos servirá como base para la creación de una base de datos homogénea que se pone a disposición de los usuarios del sistema para el análisis de datos utilizando, por ejemplo, técnicas de minería de datos. El algoritmo recuperación de información requiere una configuración inicial que se realiza la primera vez que se desea ejecutar el algoritmo. Dicha configuración requiere los siguientes datos:

- el nombre del proyecto de búsqueda.
- el dominio de búsqueda.
- la URL donde se realizará el proceso de obtención de datos.
- las rutas Xpath.

Las direcciones XPath son rutas utilizadas dentro de un documento XML para seleccionar nodos o conjunto de nodos que permiten navegar a través de ello [11]. Estas rutas Xpath permitirán encontrar los elementos dentro del XML recuperado y ejecutar interacciones con el sitio Web emulando las actividades realizadas por una persona dentro del sitio Web cuando se descarga un dato.

Adicionalmente se requiere indicar el tiempo que se estará ejecutando el sistema de *Web Scraping*. La tabla 1 muestra un ejemplo de esta información. En este ejemplo, el algoritmo, estará recuperando información de manera automática de los dominios especificados por el usuario durante el periodo indicado.

4.1. Búsqueda de información del sitio Web

La primera etapa del algoritmo de búsqueda de información del sitio Web tiene dos tareas principales: a) validar la existencia de la URL semilla proporcionada por el usuario, y b) iniciar un navegador Web de forma automática en el cual se realizará procesos de interacción con el sitio Web que permita ver el contenido oculto y recuperar las estructuras HTML de las URLs ingresadas por el usuario.

Cada una de las URLs debe pertenecer al mismo dominio para ser un conjunto de URLs válidas para la extracción. El siguiente código corresponde al ingreso de URLs

al sistema, la cual puede recibir una única URL o un conjunto de ellas siempre y cuando estas tengan el mismo dominio:

```
start_urls = [URL Semilla]

options = webdriver.ChromeOptions()

options.add_argument('--start-maximized')

options.add_argument('--disable-extensions')
```

En esta etapa el algoritmo inicia el requerimiento de acceso a los servidores de cada una de las URLs ingresadas, validando la existencia del sitio Web. Si la URL proporcionada no existe, se envía una notificación al usuario de “URL no encontrada”. En caso contrario, el algoritmo extrae una copia del sitio Web (su contenido HTML) en un archivo con extensión .XML y ejecuta un navegador Web de forma automática con características que permite eliminar extensiones o complementos que se tengan instalados en el navegador. Todo esto se lleva a cabo por cada URL proporcionada por el usuario. A continuación, se muestra el uso de una ruta *XPath* del sitio Web que permite realizar un clic en un botón de la página desencadenando una interacción:

```
WebDriverWait(driver, 20).until(

    EC.element_to_be_clickable((By.XPATH,

        '//*[@id="bPvM2"]/a')) \

    click())
```

Finalmente, cada archivo .XML recuperado es transformado en un archivo JSON, lo que permitirá la comunicación y envío de los datos a través de la librería SELENIUM [12]. A continuación, se muestra el requerimiento y ejecución del navegador:

```
def extraer_datos():

    driver = webdriver.Chrome("./chromedriver.exe", op

        tions=options)

    # inicialiar el navegador

    for url in start_urls:

        driver.get(url)
```

4.2. Recolección de información del sitio Web

La segunda etapa del algoritmo tiene dos tareas principales: a) analizar e interactuar con el contenido de cada XML utilizando las direcciones *XPath* y b) identificar los enlaces de los archivos que se desean descargar.

En esta segunda etapa se toman las rutas *Xpath* para interactuar con el elemento que apuntan esas rutas aplicando un clic o selección de valores hasta encontrar una etiqueta

de dirección tipo $\langle a \rangle$, identificando el enlace de direccionamiento asociado a esa etiqueta y guardándolo dentro de una lista de links de descarga.

Es importante hacer notar que el sistema debe replicar las acciones que realizaría un humano al intentar acceder a la información de un sitio, por ejemplo, desplegando y seleccionando opciones de diversos menús desplegables. A continuación, se muestra cómo se pasa una ruta *XPath* dentro de una función que permite recuperar los enlaces de direccionamiento de un archivo descargable:

```
Dato = WebDriverWait(driver, 20).until(
    EC.element_to_be_clickable((By.XPATH,
    '//*[@id="bPvM2"]/a'))) \
    .getattribute("href")
    Listalink.append(dato)
```

Al final de este proceso se contará con una lista de las direcciones de almacenamiento de los archivos que se pueden descargar.

4.3. Descarga de información del sitio Web

La tercera etapa del algoritmo tiene como objetivo realizar las descargas de la lista de links identificado en la etapa anterior. La descarga se lleva a cabo, realizando una iteración en la lista de links recuperados y ejecutando un “clic” de forma automática sobre el enlace en turno, esto ejecutará un evento de descarga del archivo. Este proceso se repite por cada uno de los links almacenados en la lista de archivos a recuperar. A continuación, se muestra el código del algoritmo:

```
Def descargaArchivos(listalink):
    S=0
    For d in listalink:
        r= requests.get(d, allow_redirects=TRUE)
        file_name = 'ruta del archivo + un nombre + .csv'
        output = open(file_name, 'wb')
        output.write(r.content)
        output.close()
        s+1
    print(file_name)
```

4.4. Monitoreo de la información del sitio Web

La cuarta etapa del algoritmo tiene dos tareas principales: a) revisar de manera constante los sitios Web de los cuales se haya extraído información y b) detectar actualizaciones en el contenido del sitio Web que requieren un nuevo proceso de descarga de información. El monitoreo debe ser configurado por el usuario utilizando una de las dos siguientes opciones del sistema:

En la primera opción el usuario configura el periodo de tiempo en el cual el sistema ejecutara la descarga de los datos, el cual puede ser por minuto, horas o días. De esta forma, se ejecutará el proceso de recolección de información cuando se cumpla el periodo de tiempo establecido. El código asociado a este proceso es el siguiente:

```
Schedule.every(1).minutes.do(extraer_datos) #Cada 1 minuto ejecuta la
función de extraer_datos

While True:

    Schedule.run_pending()

    Time.sleep(1)
```

La segunda opción, el sistema realizará un mapeo del sitio Web, el cual sirve como punto de comparación para la revisión constante del sitio Web en busca de un cambio o actualización del sitio. Cuando se detecta un cambio el sistema se ejecuta nuevamente el proceso de descarga de información. A continuación, se muestra el código de la configuración por mapeo de sitio Web. En caso que en algún momento la información sea borrada, se informará al usuario del cambio o inconsistencia de los datos y se solicitará una nueva configuración:

```
Response = urlopen(url).read()

currentHash = hashlib.sha224(response).hexdigest()

time.sleep(30)

response = urlopen(url).read()

newHash = hashlib.sha224(response).hexdigest()

if newHash == currentHash:

    continue
```

5. Pruebas al algoritmo Web Scraping

Las pruebas realizadas al algoritmo *WebScraping* están enfocadas a la estrategia de búsqueda del caso de estudio COVID-19. La información que se busca recolectar son casos positivos, casos de defunciones, casos negativos, casos sospechosos y la ubicación de estos índices, con el fin de generar un conjunto de datos unificado que pueda ponerse a disposición de investigadores que lo requieran. Las pruebas realizadas fueron sobre cinco dominios (datos.covid-19.conacyt.mx, data.humdata.org, datos.gob.mx, covid19.apple.com y <https://serendipia.digital>), con un total de 8 rutas *Xpath*. La ejecución de la búsqueda se realizó el día 25 de marzo del 2022.

Tabla 2. Configuración de las pruebas realizadas al algoritmo *WebScraping*.

Nombre Proyecto	Dominio de búsqueda	URL semilla	Ruta XPath
COVID CONACYT	datos.covid-19.conacyt.mx	https://datos.covid-19.conacyt.mx/#DownZCSV	//*[@id="bPvM1"]/a/span[2] //*[@id="bPvM2"]/a/span[2]
Human data	data.humdata.org	https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases	//*[@id="data-resources-0"]/div/ul/li[1]/div[3]/a[1] //*[@id="data-resources-0"]/div/ul/li[2]/div[3]/a[1] //*[@id="data-resources-0"]/div/ul/li[3]/div[3]/a[1]
datosGob	datos.gob.mx	https://datos.gob.mx/busca/dataset/informacion-referente-a-casos-covid-19-en-mexico	//*[@id="datasets-list"]/div/div[3]/ul/div[1]/a
Covid19apple	covid19.apple.com	https://covid19.apple.com/mobility	//*[@id="download-card"]/div[2]/a
Serendipia	https://serendipia.digital	https://serendipia.digital/covid19-mx/datos-abiertos-sobre-casos-de-coronavirus-covid-19-en-mexico/	// *[@id="datos_abiertos_table"]/table/tbody/tr[1+str(i)]/td[3]/a

5.1. Configuración del Web Scraping

La configuración de los cinco dominios utilizados para las pruebas se muestra en la Tabla 2. En esta tabla se muestra el nombre del proyecto, el dominio de búsqueda, la URL semilla utilizada y sus correspondientes rutas *Xpath*.

5.2. Búsqueda de información del sitio Web

La búsqueda de información del sitio Web se llevó a cabo validando la existencia de la URL Semilla. No se encontró error en ninguna de las URLs del caso de estudio, En todos los casos fue posible establecer una conexión con el sitio Web, se recuperó la estructura HTML del sitio Web y se ejecutó un navegador Web de forma automática cargando todos los elementos de la página Web abierta.

5.3. Recolección de información

En esta etapa, el algoritmo toma como entradas las rutas *Xpath* e inicia un rastreo de estos elementos dentro del archivo .XML realizando interacciones con la página Web de forma automática sobre los elementos a los cuales apuntan los *XPath*. Una vez encontrado el enlace donde se almacenaba el archivo, se copia la ruta agregándola a la

```
[ 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-220131.csv',
'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-sospechosos-220131.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-220130.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-sospechosos-220130.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-220129.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-sospechosos-220121.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-220120.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-sospechosos-220120.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-220119.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-sospechosos-220119.csv', 'http://archivos.serendipiadata.com/data_covid/covid-19-mexico-220118.csv']
```

Fig. 2. Vista parcial de la lista de links encontrados dentro de una estructura HTML.

Tabla 3. Vista parcial de los datos encontrados en el Proyecto COVID CONACYT.

URL semilla	Página Web	Descripción	Archivos recuperados
https://datos.covid-19.conacyt.mx/#DownZCSV	Casos diarios por estado + nacional	Casos confirmados Casos sospechosos Casos negativos Casos defunciones	Casos_Diarios_Estado_Nacional Confirmados_20220309
			Casos_Diarios_Estado_Nacional Sospechosos_20220309
			Casos_Diarios_Estado_Nacional Negativos_20220309
			Casos_Diarios_Estado_Nacional Defunciones_20220309
			Casos_Diarios_Municipio_Confirmado_20220309
			Casos_Diarios_Municipio_Sospechosos_20220309
			Casos_Diarios_Municipio_Negativos_0220309
			Casos_Diarios_Municipio_Defuncione_20220309

lista de archivos descargables. Este proceso se realiza de forma automática para cada uno de los elementos que se recuperaron, obteniendo al final una lista de links de direccionamiento donde se encontraban alojados los archivo a recuperar.

A continuación, Se presenta un ejemplo parcial de la lista de links encontrados dentro de la estructura HTML de un sitio Web.

5.4. Descarga de información del sitio Web

La recuperación de la información se realizó ocupando la lista de direcciones almacenadas en el paso anterior, la cual contiene las direcciones de los archivos

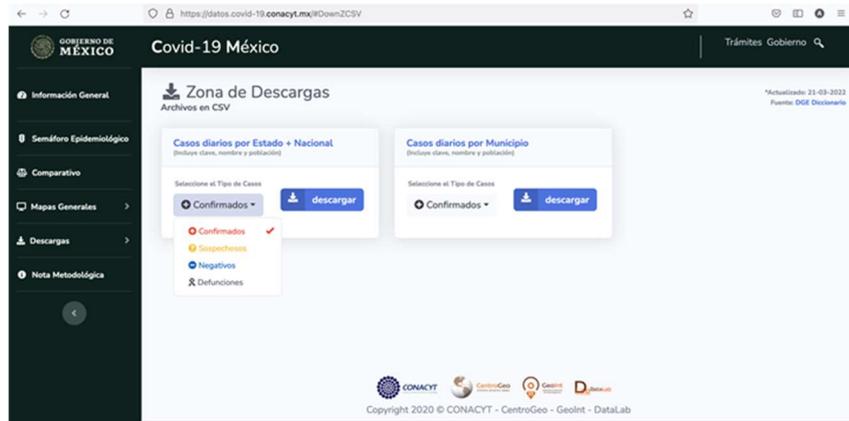


Fig. 3. Página Web con sus archivos para ser descargados por el Web Scraping.

descargables de cada sitio Web revisado. El sistema inicia el proceso de descarga ejecutando una solicitud de acceso a cada uno de los links recuperados, realizando un proceso de lectura y descarga del archivo almacenado en cada uno de los links. Este proceso se realiza por cada archivo de la lista de links de descarga de cada uno de los proyectos. En la Tabla 3, podemos observar el número de archivos descargados del día: 25-02-2022.

Los datos mostrados de manera parcial tienen ejemplos de los nombres de los archivos recuperados, una descripción y la ruta de donde se descargaron. Los archivos recuperados de la búsqueda del proyecto CONACYT fueron 12 archivos en formatos CSV. Los datos completos de la descarga de los cinco proyectos configurados se pueden consultar en la siguiente url¹. Los archivos mostrados en la tabla son los recolectados en un solo día, por lo cual en páginas como la del CONACYT con actualizaciones cada 24 horas se puede recuperar un total de 56 archivos en un periodo de 7 días.

La página Web de la URL semilla: <https://datos.covid-19.conacyt.mx/#DownZCSV> presentada en la figura anterior muestra una zona de descargas en la que un usuario debe seleccionar las diferentes opciones para la descarga de casos confirmados, sospechosos, negativos y defunciones (Figura 3).

5.5. Monitoreo de la información del sitio Web

La etapa de monitoreo de la información del Sitio Web se realiza para cada proyecto de *Web Scraping* configurado. En la Tabla 2 se observan los nombres de los proyectos configurados para la descarga de datos COVID 19 con sus respectivos dominios. Este proceso de monitoreo se lleva a cabo una vez que el sistema de recuperación ha descargado la información requerida, monitoreando los cambios que existen en el sitio Web de la información.

¹ <https://drive.google.com/file/d/10tN0gheKToix1d0jCRiPFOhD07U7hXTTo/view?usp=sharing>

Cuando el sistema detecta que existen cambios de la información descargada, entonces se descarga nuevamente la nueva información generada. Para el ejemplo presentado del COVID-19, el monitoreo se lleva a cabo cada 24 horas a partir de la primera extracción. Un análisis previo de los sitios Web permitió detectar los periodos de cambio de información dentro de los dominios utilizados.

6. Discusión

El caso de estudio del COVID-19 abordado permitió cuantificar y evaluar los resultados obtenidos en la utilización del *Web Scraping*. Este caso de estudio permitió generar diversas preguntas de investigación: ¿qué tipo de información se busca recuperar?, ¿cuáles son los formatos permitidos para realizar posteriores análisis de datos?, ¿la información obtenida sirve realmente como punto de partida para investigaciones?, ¿el *Web Scraping* permitió una descarga de información constante a un menor tiempo y recursos?, estas preguntas tuvieron respuestas interesantes para nuestro proyecto de investigación.

Las fuentes de información Web son la materia prima de este proyecto. Son estas URLs las que permiten la aplicación del *Web Scraping* obteniendo información verdaderamente relevante y de utilidad, por ello, la selección de las fuentes de información y las URL correctas son de vital importancia para la configuración de un nuevo proceso de *Web Scraping*. Los dominios de los sitios Web utilizados para las pruebas del sistema propuesto permitieron recuperar información de utilidad. Cada dominio contenía información sobre el covid-19 en México y en diferentes partes del mundo, con información en índices como casos positivos, sospechosos, negativos y defunciones.

Los 5 dominios analizados permitieron probar las técnicas *Web Scraping* en diferentes situaciones y estructuras de programación Web, cada dominio requirió una configuración inicial de proyecto permitiendo ver el desempeño del *Web Scraping* en diferentes situaciones. Los formatos de la información disponibles en estos sitios con datos COVID-19 son de tipo CSV. Este formato es el más utilizado por las diferentes dependencias para el informe de la evolución de esta pandemia, gracias a su facilidad de difusión, formato sencillo y claro. La información en este tipo de formatos es fácilmente manipulable y puede ser consumida por diferentes sistemas o algoritmos.

El tiempo que se requiere para la descarga manual de información de sitios Web con datos gigantes puede ser alta. La disminución de esfuerzo-horas-hombre delegando el trabajo de recolección de datos a un sistema computacional con reglas y métodos definidos aseguran que la información que se recolecta sea correcta según la configuración de parámetros y características definidas. Los proyectos de *Web Scraping* configurados recuperaron una cantidad de archivos considerable en cuatro horas. Las extracciones a los cinco dominios ingresados permitieron una recolección y descarga de 1050 archivos con información del COVID-19.

Estos archivos se encontraban almacenados dentro de enlaces de descarga en formatos CSV con estado público por lo cual son de libre acceso y pueden ser utilizados de forma abierta en diferentes proyectos sirviendo como base para investigaciones.

El tamaño de la información recuperada fue de 35 GB de información como datos crudos sin ningún tipo de pre-procesamiento.

7. Conclusiones y trabajo a futuro

El sistema de recuperación de información propuesto en este artículo permitió la recuperación de datos de utilidad para posteriores análisis de minería de datos. El sistema permite delegar la actividad de recolección continua de datos a partir de la Web para temas tan dinámicos como es el COVID-19. El sistema propuesto es de utilidad en entornos donde la información cambia en múltiples ocasiones a la largo del día. El sistema monitorea de forma continua el contenido de las páginas Web para detectar cambios y realizar el proceso de descarga de información.

Las 5 etapas del Web Scraping propuesto permite obtener la información rápidamente debido a las rutas XPath que apuntan directamente a la información que requiere el usuario. Los trabajos de Web Scraping como el presentado por Zhao [5] permiten que el sistema analice dentro de una página información relevante con respecto a una instrucción inicial. Mientras que nuestro Web Scraping realiza una recolección de información partiendo de una ruta XPath asegurando que la información que se recupera es realmente lo que necesita el usuario según sus configuraciones.

El sistema valida la existencia de dicha información y recuperándola según un periodo de tiempo establecido. Este Web Scraping se puede clasificar como un Scraper recursivo según la clasificación de Rodríguez [6], recuperando datos almacenados con una estructura en diferentes fechas. A diferencia de otros trabajos de Web Scraping como el presentado por Uzun [7] que utiliza patrones de recolección. Los trabajos futuros del trabajo de investigación están centrados en el mapeo automático del sistema propuesto con herramientas de minería de datos que tomen como entrada los datos recuperados por el sistema.

Referencias

1. Hernandez, A., Suarez, G., Sánchez, K., Toscano, V. M., Martínez, V., Sanchez-Perez, H.: A Web Scraping Methodology for Bypassing Twitter API Restrictions. (2018) doi: 10.48550/arXiv.1803.09875.
2. Vargas, A.: Herramienta para recopilar la información en las noticias publicadas en los sitios Web de internet. Serie Científica de la Universidad de las Ciencias Informáticas, vol. 10, no. 5, pp.14–24 (2017)
3. Suganya, R., Devi, D., Manjula, Siddharth, R. K.: An Efficient Approach for Web Indexing of Big Data through Hyperlinks in Web Crawling (2015)
4. Moreno, J. P.: Una aproximación a big data. Revista de De-recho UNED, no. 14, pp. 471–506 (2014)
5. Zhao, B.: Web Scraping. Encyclopedia de big data, pp. 1–3, Seattle: Springer (2017) doi: 10.1007/978-3-319-32001-4_483-1.
6. Villanueva, U. J.: Investigación y Desarrollo de Técnicas de Scraping. Universidad de Alcalá, Escuela Politécnica Superior (2019)

7. Uzun, E.: A Novel Web Scraping Approach Using the Additional Information Obtained from Web Pages. *IEEE Access*, vol. 8, pp. 61726–61740 (2020) doi: 10.1109/ACCESS.2020.2984503.
8. Ram, C., Santosh, P., Sadhu, B., Subarna, S.: Cloud based Web scraping for big data applications. In: *IEEE International Conference on Smart Cloud*, pp. 138–143 (2017) doi: 10.1109/SmartCloud.2017.28.
9. Glez-Peña, D., Lourenço, A., López, H., Reboiro, M., Jato, F., Riverola, F.: Web Scraping technologies in an-API world. *Briefings in Bioinformatics*, vol. 15, no. 5, pp. 788–797 (2014) doi: 10.1093/bib/bbt026.
10. Garrido, A.: Estrategia general de búsqueda de información. *asociación de enfermería española*. pp. 30–32 (2005)
11. w3schools. w3schools. https://www.w3schools.com/xml/xpath_intro.asp.
12. Selenium. https://www.selenium.dev/documentation/webdriver/getting_started.

Servicio para consumir datos de transporte público en tiempo real utilizando la plataforma FIWARE

Luis Reyes-González, Alicia Martínez-Rebollar,
Hugo Estrada-Esquivel, Yasmin Hernández-Pérez

¹Tecnológico Nacional de México/CENIDET,
Departamento de Ciencias Computacionales,
México

{m21ce024, alicia.mr, hugo.ee, yasmin.hp}@cenidet.tecnm.mx

Resumen. Hoy en día, las grandes ciudades producen una enorme cantidad de información proveniente de múltiples servicios públicos y privados. Una de las fuentes de datos que ha tenido especial interés para el desarrollo de Ciudades Inteligentes es la movilidad urbana. La movilidad ha sido uno de los factores que más ha favorecido el crecimiento económico, pero también ha generado múltiples problemas en las ciudades. Por esta razón, la movilidad también ha sido objeto de investigación por parte de muchos grupos a nivel internacional. Uno de los temas relevantes es el análisis de datos que se generan en estas grandes ciudades en tiempo real desde diferentes agentes y bases de datos. Sin embargo, estos datos se producen en formatos muy diferentes y usualmente sin considerar estándares, lo cual dificulta que los investigadores puedan usar los datos para desarrollar soluciones de movilidad inteligente. En este trabajo de investigación se presenta un servicio de software que permita consumir datos de transporte público, los cuales se encuentran dispersos en diferentes espacios de almacenamiento, estandarizarlos con modelos de datos y finalmente almacenarlos en la nube de FIWARE. De esta forma, desarrolladores y organizaciones interesadas por datos de movilidad podrán consultar el API de la nube de FIWARE y consumir los datos de movilidad ya estandarizados. Estos datos permitirán generar nuevo conocimiento o alguna aplicación que satisfaga necesidades concretas de movilidad.

Palabras clave: Modelo de datos, FIWARE, movilidad urbana, ciudades inteligentes, API.

Service for Consuming Public Transport Data in Real-Time Using the FIWARE Platform

Abstract. Currently, the big cities produce an enormous amount of information coming from several public and private services. Urban mobility is one of the

relevant data sources in the construction of Smart Cities. The mobility is one of the factors that better promote the economic growth, but it has also generated multiple issues in cities. For this reason, mobility has also been the subject of research by many groups internationally. One of the relevant issues is the analysis of data generated, in real-time, in these large cities from different agents and databases. However, these data are produced in very different formats and usually without considering standards. This makes it difficult for researchers in the field of mobility to use them for developing smart mobility solutions. In this research work, a software service is presented that allows consuming public transport data, which are located in different storage spaces, standardizing them with data models and finally storing them in the FIWARE cloud. In this way, developers and organizations interested in mobility data will be able to consult the API of the FIWARE cloud to consume standardized mobility data. These data will allow the generation of new knowledge or an application that satisfies specific mobility needs.

Keywords: Data model, FIWARE, urban mobility, smart cities, API.

1. Introducción

Actualmente, las grandes ciudades son consideradas como las principales fuentes económicas del país. Esta consideración se debe a que en las ciudades se pueden encontrar el mayor número de comercios, aglomeraciones empresariales y servicios, tales como, transporte, tecnología, comunicación vial, telecomunicaciones, etc. [1]. La movilidad en las ciudades es uno de los factores que más ha favorecido el crecimiento económico [2]. Sin embargo, esta misma movilidad se ha convertido en unos de los problemas principales para los habitantes de las ciudades, debido a los problemas que se generan al no contar con un buen sistema de predicción del tráfico, para prevenir aglomeraciones o hacer un mejor uso del transporte público [3].

Uno de los grandes problemas de los nuevos sistemas de movilidad urbana impulsados por enfoques como el Internet de las Cosas es que la información se encuentra aislada en múltiples espacios de almacenamiento, y bajo diferentes modelos de representación [4, 5, 5, 6]. Estos modelos de representación no toman en cuenta el uso de formatos estándares que permitan la reutilización de los datos almacenados.

Actualmente se cuenta con diferentes plataformas especializadas de Internet de las cosas (IoT, por sus siglas en inglés Internet of Things), que cuentan con estándares y protocolos para el manejo y procesamiento de información de IoT a gran escala, con el objetivo de crear y desplegar aplicaciones de IoT inteligentes y administrables [8][9] [10]. Una de estas plataformas de IoT es FIWARE¹.

Estas plataformas de IoT brindan la posibilidad de construir aplicaciones de movilidad inteligente que almacenen sus datos en la nube. Sin embargo, es necesario construir mecanismos que permitan el acceso a los datos, su estandarización y almacenamiento en una nube que permita la reutilización de los datos por

¹ https://fiware-training.readthedocs.io/es_MX/latest/

desarrolladores externos. Este artículo tiene como objetivo presentar un servicio software que permita consumir datos de transporte público en tiempo real y estandarizarlos en la plataforma de FIWARE [11] para su uso en aplicaciones que satisfagan alguna necesidad de movilidad urbana.

En la actualidad, los sistemas de transporte público de grandes ciudades (ej. Ciudad de México), con ayuda de los agentes que conforman la Internet de las Cosas proveen información de algunas de sus unidades de transporte [12]. En este artículo se utiliza los datos provistos por el Sistema de Corredores de Transporte Público de Pasajeros de la Ciudad de México, conocido como Metrobús, los cuales se transforman al modelo datos “*vehicle*” del estándar NGSI [12] y se envían a la nube de FIWARE para proporcionar información estandarizada que pueda ser utilizada para generar aplicaciones inteligentes que ayuden a la mejora del servicio de transporte público.

2. Trabajos relacionados

La movilidad urbana es un tema de gran interés y ha sido abordada desde diferentes perspectivas. En esta sección se presentan algunos de los trabajos más representativos en el área de sistemas de información para movilidad en ciudades. En [13] se analizó la movilidad como uno de los principales riesgos de enfermedades crónicas, al analizar el sedentarismo de las personas. En este trabajo sólo se analizan datos en tiempo real de movilidad para el estudio del sedentarismo, sin embargo, a diferencia del enfoque presentado en este artículo los datos no se estandarizan ni se almacenan un único repositorio.

Otros trabajos de investigación han identificado como problemática la movilidad urbana debido a que el tráfico que se genera en las zonas metropolitanas afecta directamente a la calidad de vida de la sociedad y tiene un alto impacto económico. En [15] los autores proponen el uso de un modelo de agrupación para visualizar la estructura y las características de la movilidad urbana en la ciudad de Sao Paulo, Brasil.

Este modelo, permite encontrar patrones estructurales en la movilidad urbana y revela nuevos conocimientos sobre los sistemas de transporte. Los autores afirman que estos datos de movilidad urbana pueden ayudar a respaldar las actividades de toma de decisiones para contrarrestar esta problemática. En esta propuesta se utilizan los datos obtenidos de la encuesta Origen-Destino de la Compañía de metro de la Ciudad Metropolitana de Sao Paulo. Utilizando el Framework CUBu.

En este artículo no se aborda la estandarización de los datos de movilidad, la cual es un aspecto relevante en nuestra propuesta. En [16] se analiza la movilidad desde el punto de vista del Sistema de Transporte Público de la ciudad de Singapur. El objetivo de esta propuesta es desarrollar una interfaz de análisis visual para presentar y explorar la movilidad de los pasajeros en un sistema de transporte público.

- 1) Proponen realizar una integración de una solución de tres módulos de visualización.
- 2) Vista de mapa asíncrona para información geográfica
- 3) Vista de mapa de flujo isotime para la comparación y manipulación de información temporal efectiva

- 4) Vista de viajes pares (Origen-Destino) para el análisis visual detallada de los factores de movilidad a lo largo de rutas pares.

La información de este estudio se obtuvo de las tarjetas personalizadas RFID de los pasajeros. El sistema lector de tarjetas registra cada acción de entrada y salida de terminales. La principal diferencia con nuestro enfoque es que utiliza datos históricos y no en tiempo real. Otra propuesta en el análisis de movilidad en el transporte público es la propuesta en [17]. En esta propuesta se propone una solución de software que involucra los datos de movilidad de transporte público del Ayuntamiento de Valencia.

Un aspecto importante de esta solución es la carga masiva de datos GTFS a la plataforma de datos de FIWARE. Como método de prueba, el autor utiliza fichero GeoJSON y con ayuda de la plataforma², consiguió ver la representación gráfica de cada parada de autobuses de la Empresa Municipal de Transporte de Valencia. La diferencia con nuestra propuesta es que en el trabajo no se utilizan datos en tiempo real y solo utiliza modelos GTFS al no considerar datos provenientes de vehículos.

Otras propuestas se han dado a partir de la pandemia del SARS-COV-19 en donde la movilidad humana ha cobrado mayor impacto [18], así como la movilidad en el transcurso del tiempo a partir de esta pandemia [19]. En este caso las fuentes son diferentes que las del trabajo presentado en este artículo, ya que su fuente son datos de redes sociales y no los datos de tiempo real de vehículos o tráfico.

3. Vista general del servicio propuesto

El consumo de los datos del transporte público y su estandarización requiere del análisis de los datos con técnicas de Internet de las Cosas. En este trabajo de investigación se propone el uso de la plataforma FIWARE, la cual permite desarrollar aplicaciones de software relacionadas con Internet de las Cosas. Esta plataforma ofrece componentes abiertos para realizar análisis de grandes volúmenes de datos y la provisión de métricas en tiempo real, manipulación de información de contexto, análisis de eventos en tiempo real, recopilación de información de sensores y acción sobre actuadores [11].

FIWARE ha sido diseñado como una plataforma abierta y estándar, basada en código abierto, para fomentar la creación de los estándares necesarios para desarrollar servicios y aplicaciones inteligentes en diferentes dominios, como ciudades inteligentes, logística, energía, agricultura, industria inteligente, etc. [20]. El caso de estudio utilizado en esta investigación ha sido los metrobuses de la Ciudad de México, los cuales comparten datos acerca de la ubicación en tiempo real de sus unidades de transporte en formato General Transit Feed Specification in Real Time (GTFS-RT)[12].

El formato GTFS-RT es un formato exclusivo para representar servicios de transporte público, por ejemplo, posición de las unidades, información sobre el viaje realizado y alertas de los estatus del viaje o unidad [21]. El servicio propuesto (MeTRoBus IoT) hace uso de la tecnología FIWARE para estandarizar la información generada por las fuentes de datos, aplicando estándares como el NGSI que permite

² <http://geojson.io/>

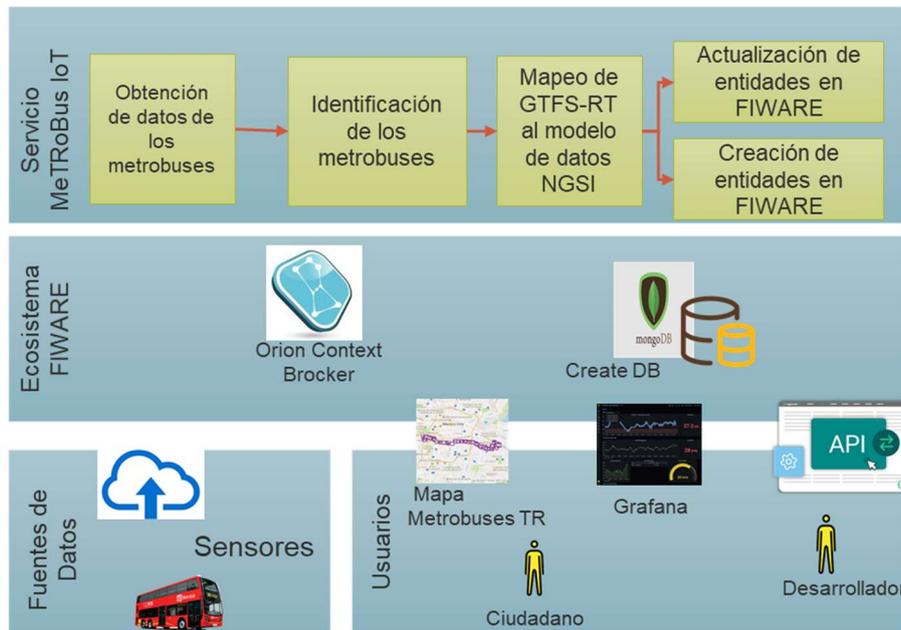


Fig. 1. Vista general de la solución propuesta.

representar objetos de la vida real en modelos de datos genéricos que puedan ser enviados al ecosistema de FIWARE. El componente Orion Context Broker está preparado para almacenar los últimos valores obtenidos por los sensores de los metrobuses [22]. Por su parte, el componente Quantumleap se utiliza para almacenar los históricos de los datos que se han capturado a lo largo del tiempo [23].

Los datos podrán ser utilizados por desarrolladores que deseen utilizar la información generada por el servicio de transporte de la CDMX. La Figura 1 muestra la vista general de la solución propuesta en este artículo, mostrando las tareas de recolección de datos, identificación de metrobuses, mapeo de datos al estándar NGSI, el envío de datos a la nube de FIWARE y su correspondiente visualización.

4. Servicio MeTroBus IoT

El servicio MeTroBus IoT para consumir datos de transporte público se encuentra organizado en 5 tareas principales:

- Obtención de los datos de los metrobuses de la Ciudad de México: se realizan peticiones *request* para la obtención de datos de los metrobuses en formato GTFS-RT del servicio provisto por la Secretaría de Movilidad de la Ciudad de México.
- Identificación de los metrobuses: la consulta de los datos de metrobuses genera una lista enorme con todos los datos de todos los metrobuses, por lo cual se

desarrolló un método para recorrer toda la lista de los datos y conservar los atributos necesarios para este proyecto de investigación.

- Mapeo de GTFS-RT a NGSI: con ayuda de los datos de la identificación de los autobuses y sus respectivos atributos se construyen los JSON para la creación y la actualización de entidades.
- Envío de datos a la plataforma FIWARE: se generan las peticiones POST o PATCH para crear o actualizar entidades en la plataforma FIWARE. Estas peticiones contienen el JSON con los datos GTFS-RT mapeados al modelo de datos “vehicle” del estándar NGSI.
- Visualización de datos: permite mostrar los datos en tableros de control o en mapas en los cuales se pueda analizar la movilidad de los autobuses.

El algoritmo que implementa el servicio del MeTRoBus IoT se muestra en la Figura 2. En las siguientes secciones se brinda mayor detalle de cada uno de estos componentes que conforman el servicio MeTRoBus IoT.

4.1 Obtención de los datos de los autobuses

Los datos de movilidad utilizados en esta investigación corresponden con los datos de localización de los autobuses de la CDMX, los cuales se encuentran en el formato Transit Feed Specification in Real Time (GTFS-RT).

La función *consumeGTFS()* nos permite consumir estos datos es la cual se encarga de realizar peticiones *request* (GET) al sitio donde se encuentran alojados los datos, y como respuesta de esta petición se obtiene los datos de ubicación de todos los autobuses de la Ciudad de México. Esta función se presenta en la Figura 3.

El resultado de esta función *consumeGTFS()* es la lista de cada uno de los autobuses de la CDMX (más de 750 autobuses en total).

La figura 4 muestra un ejemplo de la información de un solo autobús de la lista retornada por la función. Sin embargo, es importante comentar que la lista retornada por la función contiene la información de los más de 750 autobuses.

4.2 Identificación de los autobuses

La identificación de la información de cada uno de los autobuses se lleva recorriendo la lista retornada por la función *consume GTFS*. La información obtenida de cada autobús es almacenada en variables para poder mapearse al modelo de datos NGSI. La información GTFS-RT obtenida es la siguiente:

- Position-latitude: brinda la latitud de la ubicación del autobús.
- Position-longitude: brinda la longitud de la ubicación del autobús.
- Position-bearing: brinda la dirección hacia dónde va el autobús (el valor en grados).

Servicio para consumir datos de transporte público en tiempo real utilizando la plataforma FIWARE

Algoritmo ServicioMeTroBusIoT

```
Mientras TRUE Hacer
    //OBTENCIÓN DE DATOS DE LOS METROBUSES
    //Esta función se encarga de realizar consultar request al sitio de
    //datos de los metrobuses y retorna la lista de los datos de los metrobuses
    feed<-consumeGTFS ()
    //IDENTIFICACIÓN DE METROBUSES
    //Se genera el método para que recorra toda la lista de datos de los
    //metrobuses y seleccione los atributos necesarios
    Para i<-0 Hasta tamañoDelFeed Con Paso 1 Hacer
        //Selección de atributos
        datosMetrobusNGSI<-feed.entidad[i]
        //MAPEO DE GTFS-RT AL MODELO DE DATOS NGSI "Creación de entidad"
        //Se mapea los datos GTFS-RT a un JSON de la entidad "vehicle" del
        //estándar NGSI para la creación de entidades
        payloadCreacion<-llenaJSONcreacionEntidad(datosMetrobusNGSI)
        //CREACIÓN DE ENTIDADES
        //Se realiza la petición a los servicios de FIWARE para la creación
        //de la entidad
        estatusResquestPost <- crearEntidad(payloadCreacion)
        Si estatusResquestPost=existe Entonces
            //MAPEO DE GTFS-RT AL MODELO DE DATOS NGSI "Actualización de entidad"
            //Se mapea los datos GTFS-RT a un JSON de la entidad "vehicle" del
            //estándar NGSI para la actualización de entidades
            payloadActualizacion<-llenaJSONactualizacion(datosMetrobusNGSI)
            //ACTUALIZACIÓN DE ENTIDADES
            //Se realiza la petición a los servicios de FIWARE para la
            //actualización de la entidad
            estatusRequestPach<-actualizacionEntidad(payloadActualizacion)
        Fin Si
    Fin Para
    //Espera 30 segundos para repetir el ciclo
    Esperar 30 Segundos
Fin Mientras
Fin Algoritmo
```

Fig. 2. Algoritmo del Servicio MeTRoBus IoT propuesto.

- Position-odometer: brinda el valor de odómetro del metrobús.
- Position-speed: brinda la velocidad del metrobús.
- Current_Status: brinda el estatus del metrobús.
- Timestamp: brinda la hora y fecha de cuando se realiza la consulta.
- Vehicle-label: brinda el número económico de la unidad.

```
header {
  gtfs_realtime_version: "1"
  incrementality: FULL_DATASET
  timestamp: 1645115965915
}
entity {
  id: "1"
  vehicle {
    trip {
      trip_id: "8127602"
      start_time: "09:59:00"
      start_date: "20220217"
      schedule_relationship: SCHEDULED
      route_id: "437"
    }
    position {
      latitude: 19.413000106811523
      longitude: -99.11280059814453
      bearing: 2.0
      odometer: 0.0
      speed: 7.5
    }
  }
  current_stop_sequence: 31
  current_status: IN_TRANSIT_TO
  timestamp: 1645115983
  congestion_level: UNKNOWN_CONGESTION_LEVEL
  stop_id: "540"
  vehicle {
    id: "1285"
    label: "218"
  }
}
```

Fig. 4. Ejemplo de un segmento de la lista de datos de los autobuses.

El mapeo de los datos obtenidos de alguna fuente de datos debe ser estandarizada para poder ser consumida por cualquier aplicación o desarrollador que desee utilizarla. En el caso de estudio de esta investigación, la información de los autobuses se encuentra en el formato GTFS-RT, por lo que se requiere mapearse al modelo de datos “vehicle” definido para la plataforma FIWARE. La Tabla 1 muestra como son mapeadas las variables del modelo GTFS-RT al modelo de datos de datos “vehicle” especificado en el estándar NGSI de FIWARE.

La plataforma FIWARE permite conservar el dato más actual obtenido y a partir de estos datos generar una base de datos histórica. Debido a este comportamiento, fue necesario generar una estructura JSON exclusiva para la actualización de entidades.

4.3 Creación y actualización de entidades en FIWARE

Los métodos que utiliza la plataforma FIWARE para poder consultar, guardar o actualizar entidades se basan en peticiones *request*. Los datos obtenidos del sistema de autobuses y que han sido estandarizados utilizando un modelo de datos debe enviarse a la nube de FIWARE a través del componente Orion Context Broker (OCB). Una vez que los datos se encuentran en el OCB pueden ser consultados utilizando servicios REST. Los métodos para la creación y actualización de entidades son las siguientes:

- GET: consulta de información.
- POST: creación de entidades.
- PATCH: actualización de ciertos atributos de la entidad.

Table 1. Variables del modelo GTFS-RT mapeadas a las variables del modelo NGSI de FIWARE.

Datos GTFS-RT		Entidad Vehicle (NGSI)	
Variable	Descripción	Variable	Descripción
speed	Representa la velocidad que tiene el vehículo representado en m/s.	speed	Representa la velocidad que tiene el vehículo. Esta información se requiere en km/h.
odometer	Indica la distancia total en metros recorrida por el vehículo desde su producción inicial.	mileageFromOdometer	Indica la distancia total en kilómetros recorrida por el vehículo desde su producción inicial.
bearing	Representa la orientación en grados del vehículo. Son valores en decimal donde 0 es norte y 90 es este.	heading	Representa la orientación de grados del vehículo. Son valores decimales donde 0 es norte y gira en sentido de las agujas del reloj.
latitud	Representa la ubicación del vehículo. Brinda dos valores decimales separados, un valor para latitud y otro para longitud.	location	Representa la ubicación del vehículo.
longitude			Requiere los datos en un punto GeoJSON donde contiene los grados latitud y longitud.
current_status	Representa el estado del vehículo, desde el punto de vista ser servicio prestado. Posibles estados: INCOMING_AT: el vehículo está a punto de llegar a la parada. STOPPED_AT: el vehículo está detenido en la parada. IN_TRANSIT_TO: el vehículo salió de la parada anterior y está en tránsito.	servicesStatus	Representa el estado del vehículo. Estado que más se adapta: onRoute: El vehículo está realizando una misión. Se pueden agregar modificadores separados por comas para indicar qué misión está entregando actualmente el vehículo. Ejemplo: "onRoute, STOPPED_AT".
timestamp	Indica el tiempo en que se adquiere la información en formato timestamp.	timestamp	Indica el tiempo en que se adquiere la información en formato ISO 8601.
label	Representa la identificación del vehículo en el contexto de una flota de vehículos	fleetVehicleId	Representa la identificación del vehículo en el contexto de una flota de vehículos

Una vez que se cuenta con la especificación JSON de la entidad “*vehicle*” del estándar NGSI con los datos de los metrobuses se procede a realizar la petición POST o PACH ya sea para crear la entidad o actualizarla. En la figura 5 se puede observar el método que se utiliza para generar estas peticiones. Para poder identificar en qué momento crear o actualizar una entidad, se toma como referencia la respuesta del *request*. Estos son algunas respuestas:

- Request timed out: el servidor al cual se le hace la petición no contesta.
- 400: no se encuentra la URL solicitada.
- 200: petición GET bien realizada.
- 201: petición POST, creación de entidad de forma correcta.
- 422: petición POST, entidad no creada, la entidad ya existe.
- 204: petición PACH, actualización de entidad de forma correcta.

La lógica de nuestro método para la creación o actualización de entidades es: “Crear la entidad en la liga³ con los datos del metrobús en turno, si la respuesta del request es igual a 422 entonces no debe crearse sino actualizarse la entidad con la liga⁴ con los mismos datos del metrobús en turno”.

La creación de entidades permite que los valores obtenidos de los metrobuses se almacenen en el componente Orion Context Broker (OCB) de la plataforma FIWARE y puedan por tanto ser consultados por otros desarrolladores.

Para visualizar los datos en mapas se desarrolló de una aplicación web que consume datos del Orion Context Broker. Esta aplicación web es capaz de representar por lo menos un metrobús en un espacio geográfico de un mapa con sus datos tiempo real. Tomando en cuenta que los datos de los metrobuses se actualizan en el OCB cada 30 segundos, la aplicación web se automatiza para que pueda consumir datos nuevos en este intervalo de tiempo y representarlos en el mapa. Con la automatización de la aplicación web se podrá observar el recorrido que el metrobús realiza en tiempo real.

4.4 Visualización de los datos

En este proyecto de investigación se proponen dos formas para la visualizar los datos guardados en la plataforma FIWARE: tableros de control y mapas. En el caso de los tableros de control, la herramienta Quantumleap de FIWARE contempla el componente software Grafana para la visualización de datos históricos de forma gráfica. Para lograr la visualización se crea la conexión entre Grafana y CrateDB. Con ayuda de esta conexión se generan las consultas SQL necesarias para poder visualizar datos en un dashboard con diferentes gráficas o tablas.

³ <http://localhost:1026/v2/entities/>

⁴ <http://localhost:1026/v2/entities/vehicle:MetroBus:NUMERODEUNIDAD/attrs>

```
#URL para creación de entidades
urlCreacionEntidad = "http://" + str(ipServer) + ":1026/v2/entities/"
payloadCreacionEntidad=llenaJSONcreacionEntidad(idVehicle,
                                                labelVehicle,
                                                latitudeVehicle,
                                                longitudeVehicle,
                                                timeConsulta,
                                                statusVehicle,
                                                speedVehicle,
                                                bearingVehicle,
                                                odometerVehicle)

estatusResquestPOST = requests.post(url= urlCreacionEntidad,
                                   data=payloadCreacionEntidad,
                                   headers= headers_string)

#imprime el estado del POST
print("Estado POST: {}".format(estatusResquestPOST.status_code))

if(estatusResquestPOST.status_code==422):
    #URL para actualización de entidades
    urlActualizacionEntidad = "http://" + str(ipServer) + ":1026/v2/entities/vehicle:MetroBus:" + str(labelVehicle) + "/attrs"
    payloadActualizacionEntidad=llenaJSONactualizacion(
                                                labelVehicle,
                                                latitudeVehicle,
                                                longitudeVehicle,
                                                timeConsulta,
                                                statusVehicle,
                                                speedVehicle,
                                                bearingVehicle,
                                                odometerVehicle)

    estatusResquestPACH = requests.patch(url= urlActualizacionEntidad,
                                       data=payloadActualizacionEntidad,
                                       headers= headers_string)

#imprime el estado del PACH
print("Estado PACH: {}".format(estatusResquestPACH.status_code))
```

Fig. 4. Funciones para llenar el JSON de creación o actualización de entidades.



Fig. 5. Visualización del recorrido del metrobús número 9407 en el mapa.



Fig. 6. Visualización del recorrido del metrobús número 1208 en el mapa.

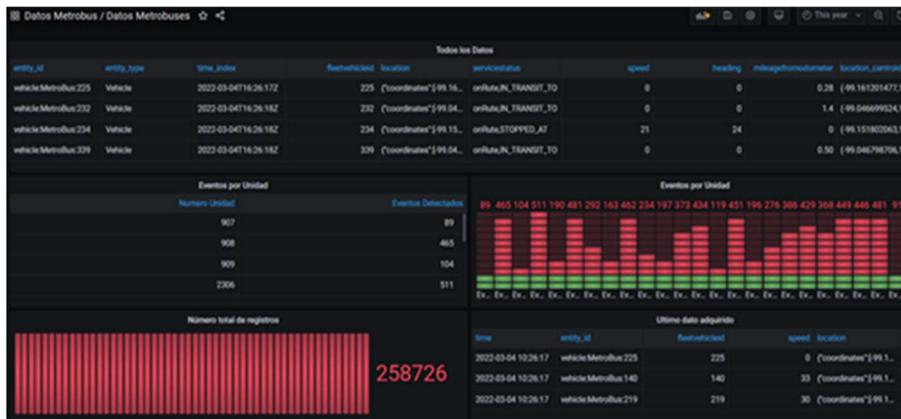


Fig. 7. Visualización de datos históricos en Grafana.

5. Pruebas al servicio MeTRoBus IoT

El servicio MeTRoBus IoT se encuentra automatizado para consumir, identificar, mapear y crear o actualizar entidades en FIWARE cada 30 segundos de los aproximadamente 750 metrobuses de la Ciudad de México que cuentan con monitoreo GPS. Como resultado del proceso, nuestra nube de FIWARE contiene datos en tiempo real y datos históricos de los metrobuses.

Esta información puede ser utilizada para el análisis de la movilidad en la ciudad. La plataforma FIWARE brinda APIs de acceso rápido a los datos, por lo que la mejor forma de validar el funcionamiento del servicio MeTRoBus IoT es visualizando de forma gráfica los datos alojados en la plataforma FIWARE. Por esa razón, se desarrolló una aplicación web que representa la posición de metrobuses en un mapa.

Esta aplicación consume los datos en tiempo real de los metrobuses de la Ciudad de México que están alojados en el Orion Context Broker de la plataforma FIWARE. En la figura 6 se muestra la trayectoria que recorrió el metrobús número 9407 en un tiempo de monitoreo de 10 minutos. La figura 7 muestra el recorrido del metrobús número 1208, con un tiempo de monitoreo de 13 minutos.

Como ejemplo adicional de la ejecución del servicio propuesto, se genera una grabación que muestra el funcionamiento del servicio MeTRoBus IoT. En la siguiente liga⁵ se puede apreciar que, mientras el servicio MeTRoBus IoT se encuentra en ejecución, la posición de los metrobuses cambia en el mapa. A diferencia de las primeras pruebas mostradas en el video, en esta ocasión no se muestra el recorrido, sino solo la posición los metrobuses en tiempo real.

Para la visualización del histórico de los datos de los metrobuses se utiliza la herramienta Grafana. En la Figura 8 se observa uno de los paneles de control con esta

⁵ <https://www.youtube.com/watch?v=x8WW5Z6pwYU>

herramienta para el proyecto presentado en este artículo. Cada una de estas tablas o gráficas se generaron a partir de consultas tipo SQL.

6. Conclusiones y trabajo a futuro

El servicio MeTRoBus IoT contribuye a la estandarización de entidades de dispositivos IoT para datos de movilidad urbana. El servicio propuesto permite obtener datos del servicio de Metrobuses de la CDMX y estandarizarlos con un modelo de datos. Estos datos ya estandarizados son alojados en la plataforma FIWARE la cual brinda APIs de acceso rápido a datos para su posterior uso por parte de desarrolladores.

Las APIs generadas en este proyecto facilitan el acceso a los datos de movilidad con el objetivo de generar nuevo conocimiento que ayude a la planificación de la movilidad urbana o a la necesidad de generar aplicaciones que satisfagan alguna necesidad de movilidad urbana. La herramienta Grafana permite tener una visualización de los datos de una forma simple e intuitivo para el usuario. Esta visualización también facilita el análisis de los datos de movilidad. Como trabajos futuros se pretende consumir los datos de la plataforma HEREMaps.

Esta plataforma brinda información acerca de la congestión vehicular de la Ciudad de México en tiempo real. Estos datos también tendrán un proceso de identificación de segmentos de ciudades con sus respectivos atributos para mapear los datos al modelo de datos “*TrafficFlowObserved*” provisto por FIWARE. Los datos de tráfico capturados a partir de HEREMaps también deberán estar alojados en la plataforma FIWARE.

La información de los metrobuses CDMX y de HEREMaps permitirá contar con datos, en tiempo real y datos históricos, de la movilidad en la Ciudad de México en un formato estandarizado. Este espacio de datos de movilidad podrá ser de utilidad para la planificación de movilidad urbana en la ciudad.

Referencias

1. REDEUSLAC Economía Urbana. (2022)
2. Montezuma, R.: Ciudad y transporte: la movilidad urbana. (2003)
3. Colchado Flores, I. C. F.: La movilidad urbana en la Ciudad de México: un problema complejo. Centro de Ciencias de la Complejidad (2017)
4. Pública, A. D. D. I.: Portal de Datos Abiertos de la CDMX. Portal de Datos Abiertos de la CDMX (2021)
5. Google. Referencia de GTFS Real-time | Transporte en tiempo real |. Google Developers. (2021)
6. FIWARE. Modelos de datos - Aprende FIWARE en español. Aprendiendo FIWARE (2020)
7. HERE Maps. Documentation, Code Examples and API References. HERE Developer. (2022)
8. KaaIoT. Enterprise IoT Platform with Free Plan | Kaa. Kaa IoT Platform (2022)
9. Bustamante, A. L.: Thinger.io plataforma. Thinger.io (2020)
10. de Panfilis, G.: About FIWARE. (2022)

11. FIWARE. La plataforma FIWARE - Aprende FIWARE en español (2022)
12. Metrobuses CDMX. Datos Abiertos. metrobús (2022)
13. FIWARE. Vehicle - FIWARE DataModels. (2019)
14. Wang, Y., König, L. M., Reiterer, H.: A smartphone app to support sedentary behavior change by visualizing personal mobility patterns and action planning (SedVis): Development and pilot study. *JMIR Formative Research*, vol. 5, no. 1 (2021)
15. Martins, T. G., Lago, N., De Souza, H. A., Santana, E. F. Z., Telea, A., Kon, F.: Visualizing the structure of urban mobility with bundling: A case study of the city of São Paulo. *Anais do Workshop de Computação Urbana. Sociedade Brasileira de Computação – SBC* (2020)
16. Zeng, W., Fu, C.-W., Arisona, S. M., Erath, A., Qu, H.: Visualizing mobility of public transportation system. *IEEE Transactions on Visualization and Computer Graphics*, vol. 20, no. 12, pp. 1833–1842 (2014)
17. Alemany, J. L. P.: Creación de un repositorio de contenidos para transporte público urbano con Fiware. *Universitat Politècnica de València Escuela Técnica Superior de Ingeniería Geodésica, Cartográfica y Topográfica, Valencia* (2019)
18. Huang, X., Li, Z., Jiang, Y., Ye, X., Deng, C., Zhang, J., Li, X.: The characteristics of multi-source mobility datasets and how they reveal the luxury nature of social distancing in the U.S. during the COVID-19 pandemic (2020)
19. Jeffrey, B., Walters, C. E., Ainslie, K. E. C., Eales, O., Ciavarella, C., Bhatia, S., Riley, S.: Anonymised and aggregated crowd level mobility data from mobile phones suggests that initial compliance with COVID-19 social distancing interventions was high and geographically consistent across the UK. *Wellcome Open Research*, vol. 5, no. 170 (2020)
20. Telefónica. FIWARE, el estándar que necesita el IoT (2017)
21. Google. Descripción general de GTFS Realtime | Transporte en tiempo real. *Google Developers* (2022)
22. FIWARE. Orion Context Broker (OCB) - Aprende FIWARE en español (2020)
23. FIWARE. QuantumLeap – QuantumLeap (2020)

Clasificación de haces de luz para el uso de energía por medio de redes neuronales

Alexi Vasquez Rodriguez, Leticia Flores Pulido,
Perfecto Malaquias Quintero, Carlos Perez Corona,
Francisco Javier Albores Velazco

Universidad Autónoma de Tlaxcala,
Facultad de Ciencias Básicas, Ingeniería y Tecnología,
México

{20181214, leticia.flores.p, perfectomalaquias.quintero.f,
carlos.perezc, franciscojavier.albores.v}@uatx.mx

Resumen. A pesar de los avances científicos en el área de la óptica aún existen fenómenos físicos de los cuales no se han investigado. En este caso de las redes neuronales en esta área no han sido del todo explotadas, los modelos de redes neuronales pueden facilitar la investigación de estas fuentes de luz. El propósito de este proyecto es crear una red neuronal, que, con base a características numéricas de las diversas fuentes de luz, se pueda clasificar eficazmente. La técnica utilizada para esta clasificación es Backpropagation. Esta es una técnica de propagación de errores hacia atrás, está programada en Python y en cada iteración se ingresarán patrones de entrada que serán los valores a clasificar. Se identificó que los resultados obtenidos se basan en la configuración de la red neuronal, esto se depende en gran medida de la forma en la que se conforme una arquitectura de red neuronal a la medida del problema.

Palabras clave: Redes neuronales, python, fuentes de luz, clasificación, inteligencia artificial.

Classification of Light Beams for Energy Use Using Neural Networks

Abstract. Despite scientific progress in optics area, there are still physical phenomena that have not been investigated. Since neural networks in this area have not been fully used, neural network models can make easier the research of these light sources. The purpose of this project is to create a neural network, which, based on numerical characteristics of the different light sources, can be effectively classified. The technique used for this classification is backpropagation. This is a backward error propagation technique programmed in Python where each iteration input patterns will be entered to be the values to classify. It was identified that the results obtained are based on the configuration

of the neural network, this depends on the way in which a neural network architecture tailored to the problem is formed.

Keywords: Neural networks, python, light sources, classification, artificial intelligence.

1. Introducción

Los haces de luz están compuestos por diferentes características, una de ellas es su longitud de onda, esta es una de las características en la que nos basamos para esta clasificación, esta onda contiene un campo eléctrico y un campo magnético, existen diferentes emisores de haces de luz, algunos de ellos son visibles y se puede determinar el color por la longitud de onda. Las redes neuronales son capaces de identificar y extraer las características que son realmente relevantes y deben utilizarse para resolver un problema, algo que escapa de las posibilidades de otras técnicas de aprendizaje automático [1].

Las redes neuronales nos ayudan a la ejecución de tareas complejas y en general específicas, nos ayudan a la clasificación de estos haces de luz ya que cuando se manda a una fuente de luz para su identificación este puede verse afectado por la luminosidad natural y es posible que arroje resultados erróneos. Estas redes neuronales están entrenadas con diferentes tipos de fuentes, este aprendizaje permite la reducción de operaciones que debería hacer para la identificación de las fuentes de luz.

2. Trabajos relacionados

En el trabajo realizado por Ramy [10], se presentó un proyecto en el que se clasifican diferentes fuentes de luz con eficiencia cuántica unitaria como fuentes de energía constante y aquellas con una eficiencia cuántica baja. Para la clasificación de una fuente de luz, la luz debe de crearse por conversión de un tipo de energía diferente. En el trabajo realizado por Ma et al. [9], se observó las características de las condiciones de luz, las condiciones de luz en interiores se describen habitualmente mediante niveles de iluminancia.

Estos mismos niveles de iluminancia, pueden ser generados por diferentes tipos de fuentes de luz, que irradian diferentes componentes. Este método fue evaluado para una serie de fuentes de luz comunes en interiores y bajo diferentes condiciones. En el artículo de Hloušek [4], se demostró que la precisión del DRNF (Detectores de resolución del número de fotones) informado mediante la medición de estadísticas de fotones para muchos estados diferentes de la luz, de los cuales se muestran 25 estados, que cubren varios números de fotones y g^2 valores. La obtención de fotones requiere mediciones repetidas utilizando un detector de resolución de número de fotones, algunos de los parámetros importantes para este son el rango, la velocidad y la precisión.

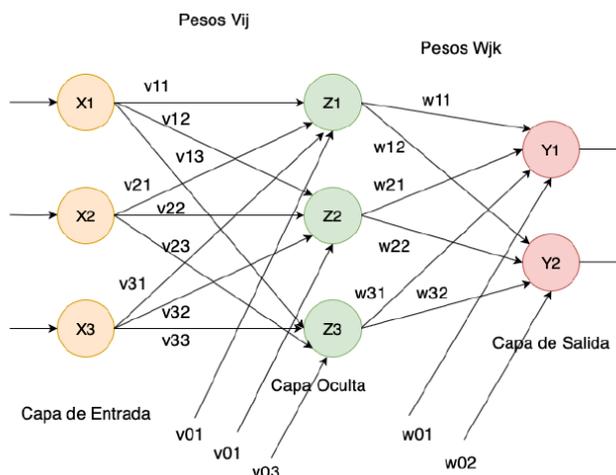


Fig. 1. Arquitectura básica de la red neuronal perceptrón multicapa. (Figura tomada de [7,8]).

En el trabajo hecho por Cohen [6], se mostró uno de los métodos de detección particulares, a saber, la teledetección láser, la detección y el alcance de la luz, está en continuo desarrollo, se han utilizado detectores monofónicos y detectores de resolución numérica de fotones para este fin. En el trabajo hecho por Kudyshev [5], se observó que el nano-ensamblaje determinista puede permitir la integración de dispositivos fotónicos cuánticos únicos en el chip. En este trabajo se implementó una clasificación supervisada basada en el aprendizaje automático de los emisores cuánticos como “simples” o “no simples” basándonos en sus datos de autocorrelación dispersos. En el trabajo realizado por Villegas [11], desarrollaron una mezcla de partículas, las cuales llevan mucho tiempo desempeñando un papel fundamental en muchas ramas de la ciencia y la tecnología, asistida por aprendizaje automático. Implementando varias redes neuronales conectadas en serie, cada una de las cuales realiza una predicción específica de las características del campo inicial.

3. Métodos y materiales

3.1. Algoritmo Backpropagation

La técnica Backpropagation, definido así por regresión de errores hacia atrás, es una técnica de redes neuronales que calcula el error de cada neurona de la última capa y lo propaga hacia atrás para saber el porcentaje de errores en toda la red. En la Figura 1, se muestra la arquitectura básica de la red neuronal perceptrón multicapa.

Esta arquitectura contiene una capa de entrada, una capa oculta, y una capa de salida, cada una de ellas interconectadas por flechas que representan los pesos

de las conexiones entre estas neuronas. Para los cálculos de esta red neuronal se necesitan las siguientes fórmulas tomadas [7], [8]. Para el cálculo de la capa oculta se necesita la siguiente ecuación (1), con su función de activación (2):

$$Z_j = v0_j + \sum x_i * v_{ij}, \quad (1)$$

$$Z_j \text{activada} = \frac{1}{1 + \exp(-Z_j)}. \quad (2)$$

Para el cálculo de la capa de salida se necesita la siguiente ecuación (3), con su función de activación (4):

$$Y_k = w0_k + \sum z_j * w_{jk}, \quad (3)$$

$$Y_k \text{activada} = \frac{1}{1 + \exp(-Y_k)}, \quad (4)$$

donde en la expresión (1), $v0_j$ y v_{ij} son los pesos de la capa de entrada a las capas ocultas, x es la entrada de los datos, y Z_j es la el conjunto de las neuronas ocultas. En la expresión (2), Z_j activada nos indicará si cada neurona oculta aprendió o no cada patrón de entrada. Por otro lado, en la expresión (3), tenemos que Y_k es la salida de clasificación, $w0_k$ son los pesos bias de la capa oculta a la capa de salida, z_j son las neuronas ocultas, y w_{jk} son los pesos de la capa oculta a la capa de salida.

En la expresión (4), Y_k activada nos indica si la neurona de salida aprendió o no el patrón de entrenamiento. En la expresión (5), se calcula el valor del error de salida (y). Cálculo del error de salida (y) e incremento de los pesos (w). Para calcular el error de salida se necesita la siguiente ecuación (5), donde t_k es la salida deseada del patrón 1, en este caso $t_1 = 1$:

$$\delta_k = (t_k - Y_k \text{activada}) * (Y_k \text{activada}) * [1 - Y_k \text{activada}]. \quad (5)$$

Para calcular los incremento de los pesos w que inciden en y necesitamos la siguiente ecuación (6), donde se hace el uso de un parámetro de aprendizaje alfa:

$$\Delta w_{jk} = \alpha * \delta_k * z_j \text{act}. \quad (6)$$

Para el cálculo del incremento de los bias se usa la siguiente ecuación (7):

$$\Delta w0_j = \alpha * \delta_j. \quad (7)$$

Cálculo del error oculto, activación e incremento de los pesos y bias v . Para este cálculo se sumará en cada unidad los errores ocultos delta que se obtuvieron en el paso anterior, usando la siguiente expresión:

$$\delta(z_j) = \sum \delta_j * w_{jk}. \quad (8)$$

Para la activación del error oculto se usa la siguiente expresión:

$$\delta(z_{j \text{act}}) = \delta z_j * z_{j \text{act}} * (1 - z_{j \text{act}}). \quad (9)$$

Para el cálculo del incremento de los peso v , se usa la siguiente ecuación:

$$\Delta v_{ij} = \alpha * \delta_j * x_i. \quad (10)$$

Para el cálculo del incremento de los bias se usa la siguiente ecuación:

$$\Delta v_{ij} = \alpha * \delta_j. \quad (11)$$

Finalmente para la actualización de los pesos w , v , bias w y bias v se usa la siguiente fórmula para todos los casos:

$$w_{ij} \text{ nuevo} = w_{ij} + \Delta w_{ij}. \quad (12)$$

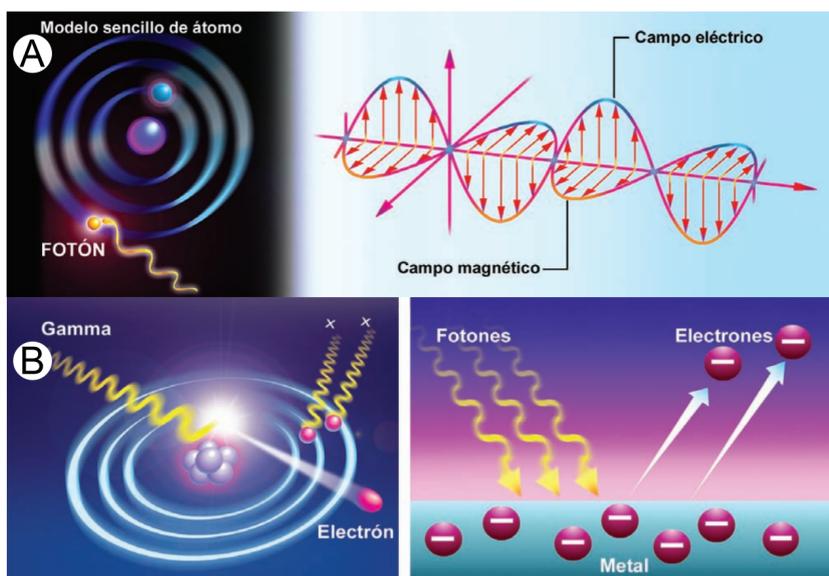


Fig. 2. Naturaleza de la luz y efecto fotoeléctrico. (Figura tomada de Bravo [2]).

3.2. Haces de luz

Los haces de luz conocidos por partículas u ondas de luz, en el libro de Hecht [3], habla sobre la definición de que la luz es un flujo constante de partículas submicroscópicas llamadas fotones. La luz es el resultado de un proceso cuántico que ocurre cuando un electrón se excita y pasa a un estado de energía alta y después la pierde. En la Figura 2 tomada de Bravo [2], la naturaleza de la luz.

- a) Se observa que la luz consiste en ondas electromagnéticas que se diferencian solo en su frecuencia. Esta onda electromagnética está compuesta

por dos perturbaciones: la primera es un campo eléctrico que vibra perpendicularmente a un campo magnético, también podemos observar el efecto fotoeléctrico.

- b) Este fenómeno consiste en que los electrones de la superficie de algunos metales se desprenden cuando se hacen incidir la luz sobre ellos.

4. Implementación

En la Figura 3 se observan los módulos que se desarrollarán las siguientes partes: Haces de luz: identificar diferentes haces de luz generados por fuentes naturales o artificiales tomando los datos de sus características de cada una de ellas. Técnica Backpropagation: se implementó la técnica Backpropagation, el objetivo es ajustar los pesos de una red neuronal en base a la tasa de error obtenida anteriormente, este ajuste de los errores garantiza una tasa de error más baja.

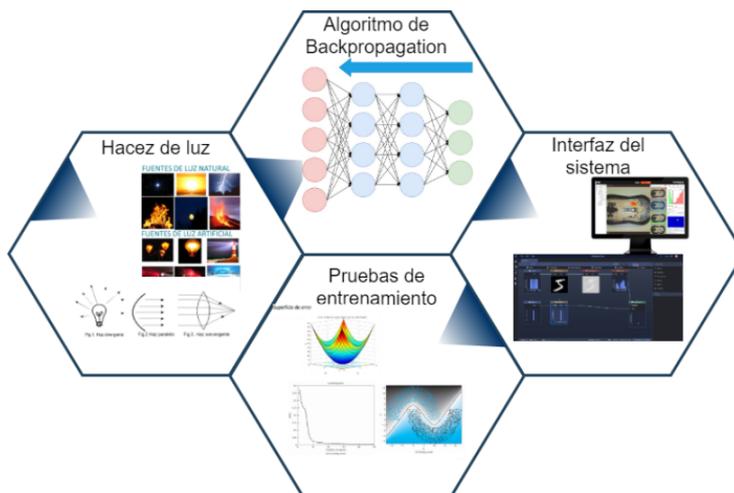


Fig. 3. Diagrama general del sistema de clasificación de haces de luz por medio de redes neuronales.

Prueba de entrenamiento: se analizan los resultados del entrenamiento en la cual se basa la técnica de Backpropagation. Interfaz del sistema: se desarrolló una interfaz para el usuario final, observaremos los resultados arrojados y el porcentaje de clasificación de estos haces de luz como se observa en la Figura 4.

En la Figura 4 se muestra la interfaz implementada para este proyecto. Se muestra la interfaz inicial, esta contiene principalmente un menú de opciones, en el menú de “Archivo” y “Ayuda”, en el apartado “Archivo” veremos 3 opciones

las cuales son: Abrir archivos entradas, Abrir archivo salidas y Salir, en la parte de “Ayuda” solo tendremos una opción de “Información” general del programa.

Dentro de la ventana tenemos los elementos con los que funciona la red neuronal, 1) tenemos la arquitectura propuesta, 2) ventana para visualizar los datos que ingresamos, 3) un botón de gráfica si se desea graficar los datos de entrada, 4) tenemos dos cuadros de entrada, uno se usará para ingresar el valor de Alpha y el siguiente para el número de las épocas y tenemos el botón de Backpropagation que nos ayudará a iniciar el entrenamiento de la red neuronal, 5) tenemos la gráfica de los resultados de salida, en la gráfica podemos dar clic en ella para generar otra ventana y verla más detalladamente, 6) se muestra la matriz de confusión de igual manera podemos dar clic para verlo más a detalle, 7) observamos algunas etiquetas las cuales nos muestran el porcentaje de la clasificación realizada y debajo de ellas un botón para observar gráficamente el cambio de los pesos que se hicieron durante el proceso.

5. Análisis de resultados

En la siguiente Tabla 1 se muestran los mejores resultados de esta clasificación en cada una de sus pruebas, se usaron 3 diferentes tipos de arquitecturas, la primera contiene 3 neuronas en la capa oculta, la segunda contiene solo 2 neuronas ocultas, y por último una arquitectura con 4 neuronas en la capa oculta.

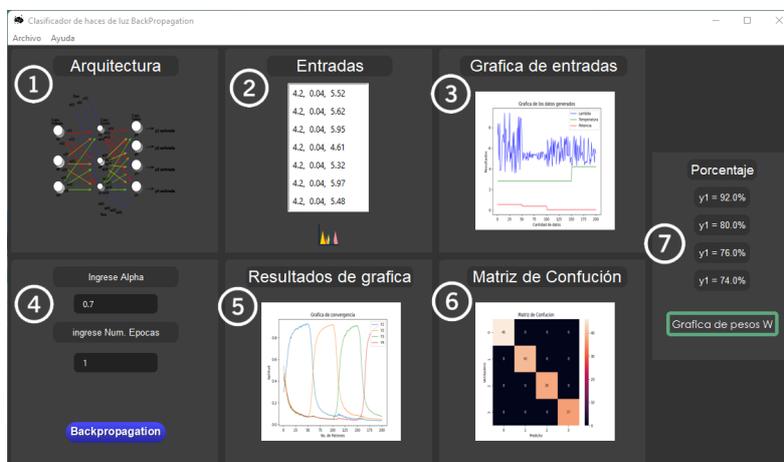


Fig. 4. Diagrama general del sistema de Clasificación de haces de luz por medio de redes neuronales.

Cada una de estas arquitecturas se probaron con 3 diferentes valores para Alpha y el número de épocas, la primera configuración consistió en Alpha = 0.3 con un número de épocas = 10, la segunda configuración con un Alpha = 0.6 y

un número de épocas = 100, y la tercera configuración con un Alpha = 0.9 y un número de épocas = 1000.

En la Figura 5 se muestran los mejores resultados obtenidos por estas tres arquitecturas, cada una de ellas con diferente configuración, la arquitectura 1 se configuró con el parámetro de Alpha de 0.3 y 10 épocas, la arquitectura 2 con un Alpha de 0.6 y 100 épocas y finalmente la arquitectura 3 con un Alpha de 0.3 y 10 épocas. Se muestran los resultados con matrices de confusión para la mejor observación de estas clasificaciones. La parte A), corresponde a la arquitectura 1, la parte B), corresponde a la arquitectura 2, y la parte C), corresponde a la arquitectura 3.

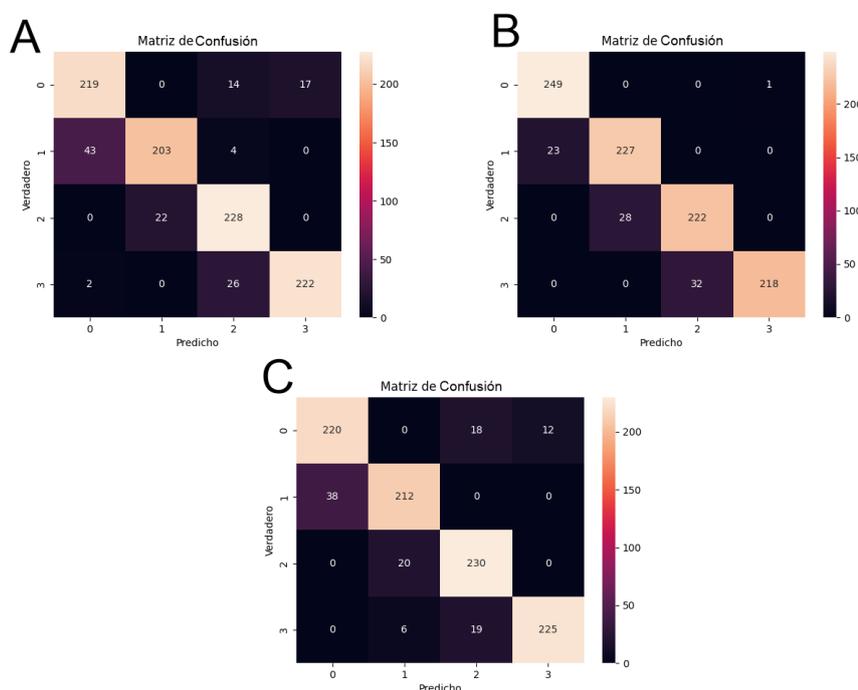


Fig. 5. Matrices de resultados para la mejor configuración del modelo utilizado.

6. Conclusiones

En este proyecto se hizo la clasificación de 4 diferentes tipos de haces de luz, cada uno de ellos con características numéricas en diferentes rangos. Las características que se tomaron fueron de longitud de onda, temperatura y potencia. Algunas de estas características se asemejan entre las fuentes de luz, como fue la longitud de onda. Una de las complicaciones al clasificar estas fuentes

de luz fue que algunas características tienen valores semejantes entre las fuentes y esto provoca una mala clasificación del modelo de red neuronal.

Por otra parte, la técnica Backpropagation es una técnica robusta y eficaz, usando algunos valores en los parámetros del modelo, usando una buena configuración de los parámetros podemos llegar a una perfecta clasificación de estos haces de luz. Usando este modelo con programación en Python, y además, creando el algoritmo desde cero podemos entender a mayores rasgos, que las redes neuronales se pueden implementar en problemas específicos, pero no todas las redes neuronales se adaptan a diferentes problemas.

Tabla 1. Tabla de resultados de todas las pruebas hechas.

Resultado de Pruebas					
Pruebas	Y1	Y2	Y3	Y4	Promedio
Prueba 1	87.6 %	81.2 %	91.2 %	88.8 %	87.2 %
Prueba 2	98.4 %	15.6 %	89.2 %	96.8 %	75 %
Prueba 3	100 %	0.0 %	0.0 %	100 %	50 %
Prueba 4	100 %	86.4 %	84.0 %	81.6 %	88 %
Prueba 5	99.6 %	90.8 %	88.8 %	87.2 %	91.6 %
Prueba 6	99.2 %	92.4 %	90.8 %	89.6 %	93 %
Prueba 7	88.0 %	84.8 %	92.0 %	90.0 %	88.7 %
Prueba 8	90.0 %	80.8 %	95.6 %	47.6 %	78.5 %
Prueba 9	100 %	0.0 %	0.0 %	100 %	50 %

Así se pudo observar que las topologías que mejor arrojaban el resultado eran las que tenían menos neuronas ocultas, esto podría deberse a que se cae en el sobre entrenamiento al agregar muchas neuronas en la capa oculta, si agregamos muchos nodos ocultos el valor de eficiencia del modelo se reduce bastante como se mostró en las pruebas.

Referencias

1. Berzal, F.: Redes neuronales & deep learning, Entrenamiento de redes neuronales artificiales. Spanish Edition (2018)
2. Bravo, A., de la Torre Bravo, A., Córdova, D.: Procedimientos endoscópicos en gastroenterología (2009)
3. Hecht, E.: Optica. Addison Wesley Longman (Pearson Education) (2000)
4. Hloušek, J., Dudka, M., Straka, I., Ježek, M.: Accurate detection of arbitrary photon statistics. Phys. Rev. Lett. 123, 153604 (2019)
5. Kudyshev, Z. A., Bogdanov, S. I., Isacsson, T., Kildishev, A. V., Boltasseva, A., Shalaev, V.M.: Rapid classification of quantum sources enabled by machine learning. Adv. Quantum Technol, vol. 3 (2020)

6. Cohen, L., Matekole, E .S., Sher, Y., Istrati, D., Eisenberg, H .S., Dowling, J.P.: Thresholded quantum LIDAR : Exploiting photon-number-resolving detection. In: OSA Quantum 2.0 Conference, OSA Technical Digest, Optical Society of America (2019)
7. Flores, L.: Antología de redes neuronales. Universidad Autónoma de Tlaxcala, México (2019)
8. Fausett, L. V.: Fundamentals of neural networks: Architectures, algorithms, and applications. Prentice-Hall (1994)
9. Ma, X., Bader, S., Oelmann, B.: Characterization of indoor light conditions by light source classification. *IEEE Sensors Journal*, vol. 17, no. 12, pp. 3884–3891 (2017)
10. Ramy G. S., El-Dardiry, S. F., Legendijk, A.: Classification of light sources and their interaction with active and passive environments (2011)
11. Villegas, A., Quiroz-Juarez, M., URen, A., Torres, J., León-Montiel, R.: Identification of particle mixtures using machine-learning-assisted laser diffraction analysis (2021)

Técnicas metaheurísticas aplicadas a un mecanismo de 4 barras implementado en dispositivos de rehabilitación

Francisco Javier Espinosa-García¹, Rocío Ambrocio-Delgado¹,
Esther Lugo-González², Alfredo Enrique Fera-De la Rosa²,
Alejandro Barragán-Bernardo², Lizeth Enriquez-Bravo²

¹Universidad Tecnológica de la Mixteca,
División de Estudios de Posgrado, Oaxaca,
México

²Universidad Tecnológica de la Mixteca,
Instituto de Electrónica y Mecatrónica, Oaxaca,
México

{fjeg_1234, cashe_11}@hotmail.com, elugog@mixteco.utm.mx

Resumen. Se presentan 4 técnicas metaheurísticas, Algoritmos Genéticos (AG), Evolución Diferencial (ED), Grey Wolf Optimizer (GWO) y Earthquake Algorithm (EA) para la optimización de los parámetros de diseño de un mecanismo de 4 barras en función de una trayectoria. El objetivo es evaluar el desempeño y determinar que algoritmo ofrece mejores resultados para este tipo de problemas. Posteriormente, se presentan implementaciones del mecanismo de cuatro barras para el seguimiento de trayectoria de un dedo robótico, una prótesis de rodilla y un exoesqueleto de miembro inferior. Se comparan los resultados reportados en estos tres artículos específicos contra los generados en este trabajo para validar que la optimización realizada con el algoritmo que dio un mejor resultado en los experimentos realizados puede mejorar el seguimiento de trayectoria. Los resultados muestran que el ED es la mejor solución para la resolución de problemas relacionados con la síntesis de trayectoria de un mecanismo de 4 barras.

Palabras clave: Optimización, mecanismo de cuatro barras, prótesis.

Metaheuristic Methods Applied to a 4-bar Mechanism Implemented in Rehabilitation devices

Abstract. Four metaheuristic techniques are presented, Genetic Algorithms (GA), Differential Evolution (DE), Gray Wolf Optimizer (GWO) and Earthquake

Algorithm (EA) for the optimization of the design parameters of a 4-bar mechanism as a function of a trajectory. The objective is to evaluate performance and determine which algorithm offers better results for this type of problem. Subsequently, implementations of the four-bar mechanism for tracking the trajectory of a knee prosthesis, a lower extremity exoskeleton, and a robotic finger are presented. The results reported in these three specific articles are compared with those generated in this work to validate that the optimization carried out with the algorithm that gave the best result can improve trajectory tracking. The results show that the DE is the best solution for solving problems related to the trajectory synthesis of a 4-bar mechanism.

Keywords. Optimization, 4-bar mechanism, prosthesis.

1. Introducción

En las últimas décadas, ha existido un gran interés entre los investigadores por generar alternativas enfocadas en la resolución del problema de síntesis de mecanismos de cuatro barras con más de 5 puntos de precisión, porque analíticamente no existe una solución exacta. Por ello, se han propuesto métodos como: gradiente exacto, programación lineal y métodos de curva de función de acoplamiento, capaces de generar soluciones aproximadas. El inconveniente de estos es que en ocasiones quedan “atrapados” en mínimos locales y se limitan a resolver funciones de optimización específicas [1].

En consecuencia, se han desarrollado técnicas de optimización heurísticas y metaheurísticas [2] capaces de resolver problemas con un gran número de restricciones. La optimización heurística trata de alcanzar una solución factible utilizando una evaluación de prueba y error en un periodo corto de tiempo. Por otro lado, los algoritmos metaheurísticos utilizan la probabilidad y estrategias para mejorar procedimientos heurísticos y hacer una búsqueda local para encontrar una solución global [3].

Una clasificación general de los algoritmos metaheurísticos es considerando 4 categorías: métodos basados en la evolución (ED), en fenómenos físicos (EA [4]), en enjambres (GWO [5]) y en seres vivos [1]. También existen las hiperheurísticas, que son técnicas metaheurísticas independientes del problema que automatizan el proceso de selección y generación de múltiples heurísticas de bajo nivel para resolver problemas estáticos de optimización combinatoria [6].

Respecto a los algoritmos metaheurísticos, los más utilizados para el problema de síntesis de trayectoria son los Algoritmo Genético (AG), las partes que conforman el algoritmo básico son la población, selección cruce y mutación [7]. Otro algoritmo que también utiliza la mutación y el cruce es el de Evolución Diferencial (ED), este es poblacional de búsqueda directa y simple, el cual es capaz de optimizar hasta alcanzar el óptimo global en funciones multimodales, no diferenciables y no lineales [8].

Respecto a las metodologías metaheurística basadas en animales, se tiene la Grey Wolf Optimizer (GWO) inspirada en los lobos grises [5].

El algoritmo GWO imita la jerarquía de liderazgo y el mecanismo de caza de los lobos grises en la naturaleza. Se emplean cuatro tipos de lobos grises: alfa, beta, delta y omega para simular la jerarquía de liderazgo. Además, se implementan los tres pasos principales de la caza: buscar la presa, rodearla y atacarla.

En [5] GWO mostro un alto rendimiento en problemas sin restricciones y con restricciones. Además, en la literatura se puede encontrar la aplicación de otros métodos como, colonias de abejas artificiales (ABC) [9], la evolución diferencial adaptativa con archivo externo opcional (JADE) [10] y la optimización basada en la enseñanza-aprendizaje (TLBO) [11] o la optimización de enjambre de partículas (PSO) [9]. Con relación a la implementación de mecanismos de 4 barras para la rehabilitación física, en [12] se presenta el diseño de un mecanismo para dedo robótico, el cual se obtuvo utilizando la síntesis de seguimiento de trayectoria en función del movimiento de flexión de un dedo. Se utilizaron los AG como método de solución, al final se presenta un mecanismo de tipo subactuado en función de la solución óptima. En [13, 14, 15] se analiza una prótesis de rodilla considerando como función objetivo minimizar el error entre el ICR (Centro Instantáneo de Rotación) de la rodilla humana de referencia y el ICR de la prótesis de rodilla.

La optimización del diseño de la rodilla protésica se ha realizado a través del MOMICA (algoritmo competitivo imperialista modificado multiobjetivo), GA; MOPSO entre otros para resolver el problema de dimensionamiento de eslabones y minimizar error en seguimiento de trayectorias. En [16] y [17] se presenta un sistema de exoesqueleto de miembro inferior que ayuda a la rehabilitación de la articulación de la rodilla. Se utiliza un mecanismo de cuatro barras cruzadas para simular el ligamento cruzado interno de la articulación de la rodilla humana para realizar el movimiento de centro de rotación instantáneo de la articulación de la rodilla.

En [9] se plantea un problema de optimización en dos etapas para minimizar el error entre las trayectorias de cadera generadas y las deseadas. Para esta problemática se utiliza la optimización por PSO y TLBO. Debido a la gran variedad de técnicas metaheurísticas existentes, se vuelve difícil determinar qué tipo de algoritmo dará los mejores resultados con menores recursos.

Por ello, en este artículo se presenta una comparativa entre los más utilizados (AG) y tres métodos de naturalezas diferentes (ED, GWO y EA) para hallar el más robusto y eficiente para la síntesis de mecanismos. Además, el algoritmo ganador es aplicado a tres casos enfocados en el desarrollo de dispositivos de rehabilitación, comparando los resultados obtenidos contra los reportados en dichos artículos y determinar cuál es el mejor para solucionar problemas referentes al diseño de mecanismos de 4 barras implementados en dispositivos de rehabilitación.

2. Función objetivo

Para obtener la función objetivo de un mecanismo de cuatro barras, se toma como referencia lo presentado en [7]. Para simplificar el desarrollo, aquí solo se presenta la función objetivo a utilizar para minimizar el error en el mecanismo.

Este se refiere a la diferencia entre los puntos deseados (P_{dx}, P_{xy}) y los generados (P_x, P_y) en el seguimiento de trayectorias para obtener el mínimo error.

$$f(x) = \sqrt{(P_{dx} - P_x)^2 + (P_{dy} - P_y)^2} \quad (1)$$

3. Casos de estudio

Los experimentos se basan en la optimización de mecanismos de cuatro barras para diferentes trayectorias. Se proponen cuatro casos de estudio: uno enfocado a la estructura del mecanismo de cuatro barras y tres orientados a una aplicación en el área de prótesis y rehabilitación. Para cada caso se muestra el diagrama cinemático del mecanismo, las ecuaciones cinemáticas y los datos correspondientes a la trayectoria deseada. La función objetivo para todos los casos es la que se muestra en la ecuación (1). Todos los experimentos descritos se realizaron en una computadora con un procesador Intel i7-9750 a 260 GHz, 6 núcleos y 16 Gb de memoria RAM, con un sistema operativo Windows 10. Los algoritmos se implementaron en MatLab.

Caso 1: Comparación de los métodos

El primer ejercicio se basa en la optimización de un mecanismo de cuatro barras utilizando la trayectoria de la Tabla 1 propuesta por [7]. En este trabajo se empleó el método de diferencia central para la solución del problema y el error reportado es 6.276×10^{-3} . Como alternativa se programan cuatro algoritmos (AG, ED, GWO y EA), los parámetros empleados se muestran en la Tabla 2. Se considera el diagrama mostrado en la Fig. 1 y las ecuaciones descritas en [7].

donde G corresponde al número de generaciones, NP es el número de población, F es un factor de escalamiento y Sr, representa el rango de onda, parámetro exclusivo del algoritmo EA. Se realizaron 31 corridas para cada algoritmo con la finalidad de examinar la variabilidad de los resultados. En la Fig.2 se muestra la gráfica deseada y las correspondientes a los mejores resultados obtenidos por cada algoritmo. Además, en la Fig. 3 se muestra la respuesta de cada uno de ellos.

Por otro lado, en la Tabla 3 se muestran los mejores y peores errores en el seguimiento de la trayectoria deseada.

Con base en los resultados mostrados en la Fig. 2 y en la Tabla 3, se observa que el error reportado en [7] mejoró con la aplicación de los algoritmos GA y ED. En cambio, comparado con el GWO el resultado no mejoró ya que el obtenido presenta una diferencia de -5.4×10^{-4} [mm], que no es tan grande como error presentado por EA, con una diferencia de -0.5506[mm].

De acuerdo con la Tabla 3, de los 4 algoritmos se observa que el peor resultado fue arrojado por EA, seguido de AG, GWO y ED. Referente a la Fig. 3, se observa que el GA presenta una mayor velocidad de respuesta y antes de las 100 iteraciones logra una aproximación a 0. En segundo lugar, se encuentra el ED, se observa que después de 150 iteraciones logra aproximarse a cero.

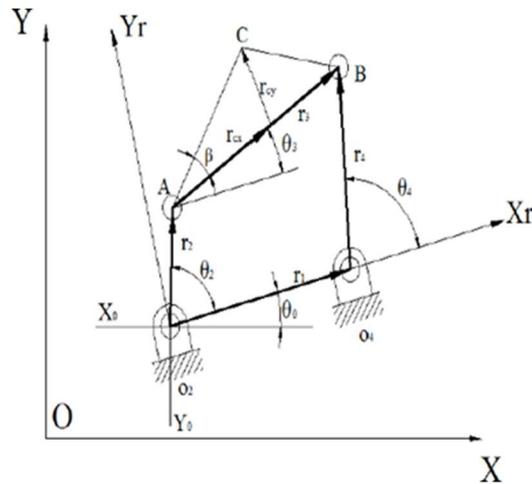


Fig. 1. Mecanismo de cuatro barras.

Tabla 1. Trayectoria deseada.

Coordenadas					
X	3.000	2.759	2.372	1.890	1.355
Y	3.000	3.363	3.663	3.862	3.943

Tabla 2. Parámetros.

DE	GA	EA	GWO
$G = 500$	$G = 1000$	$G = 2000$	$G = 2500$
Cruzamiento 0.5	Cruzamiento 0.8	$Sr = 2$	$NP = 600$
Mutación 0.5	Mutación 0.1	$NP = 150$	
$NP = 100$	$NP = 100$		
$F = 0.6$			

Respecto a GWO y EA el número de iteraciones es mayor ya que tienen respuestas más lentas comparadas con AG y DE.

En cuanto a la calidad de los resultados se observó que las soluciones obtenidas con el EA en un 0.36% están en el orden de 1×10^{-1} y el 99.96% fueron resultados no válidos, es decir no se logró una aproximación cercana a cero.

De las soluciones obtenidas con el GA el 2.117% están en el orden de 1×10^{-1} , el 0.161% en el orden de 1×10^{-2} , 0.02% en el orden de 1×10^{-3} y el 97.7% fueron resultados no válidos. En cuanto al GWO de los resultados obtenidos el 98.14% están en el orden de 1×10^{-1} , el 1.81% en el orden de 1×10^{-2} y el .049% en el orden de 1×10^{-3} . De los resultados obtenidos con el DE el 84.5% están en el orden de 1×10^{-2} y el 15.5% en el orden de 1×10^{-3} .

Tabla 3. Comparación de resultados.

Algoritmo	Mejor	Peor	Promedio	Desviación estándar
AG	0.0050	49.4405	4.3556	10.2457
ED	0.0043	0.0261	0.0054	0.0038
GWO	0.0068	0.9816	0.4401	0.2768
EA	0.5569	826.2087	128.5539	175.4942

Después de contrastar la velocidad de respuesta, el mejor resultado obtenido y la calidad de las soluciones se determinó que el mejor algoritmo es el ED. No es el más rápido pero la calidad de las soluciones presentadas es superior a la de los otros algoritmos, además no presentó soluciones no validadas, lo cual le da la característica que con menos recursos se pueden hallar soluciones por debajo de 1×10^{-1} . Por ello, en los siguientes casos se implementará el ED.

Caso 2: Optimización de un mecanismo para dedo

El segundo experimento se orientó a la optimización de un mecanismo para dedo presentado en [12].

Básicamente, el problema plantea obtener las dimensiones de un mecanismo en función de la trayectoria generada por el dedo índice, tomando como punto de seguimiento P_d (Fig. 4) para generar la trayectoria deseada cuyos valores se muestran en la Tabla 4.

Las ecuaciones que describen el mecanismo son expresadas como:

$$B_{x0} = a_1 \cos \theta_1, \quad (2)$$

$$B_{y0} = a_1 \sin \theta_1, \quad (3)$$

$$C_{x0} = B_x + a_2 \cos \theta_2, \quad (4)$$

$$C_{y0} = B_y + a_2 \sin \theta_2, \quad (5)$$

$$P_{x0} = C_x + a_3 \cos \theta_{a_3+\beta}, \quad (6)$$

$$P_{y0} = C_y + a_3 \sin \theta_{a_3+\beta}. \quad (7)$$

donde P_{x0} y P_{y0} corresponden a las coordenadas del punto P . A continuación, se describe el problema de optimización considerando un algoritmo genético y los parámetros implementados en [12], además se proponen los parámetros para utilizar el DE.

El problema de optimización se define por:

- Variables de diseño: $[\theta_1^1, \dots, \theta_1^5, a_1, a_2, x_0, y_0, \beta]$.
- Puntos deseados: $[(P_{xd}^1, P_{xd}^5), \dots, (P_{yd}^1, P_{yd}^5)]$ mostrados en la Tabla 4.

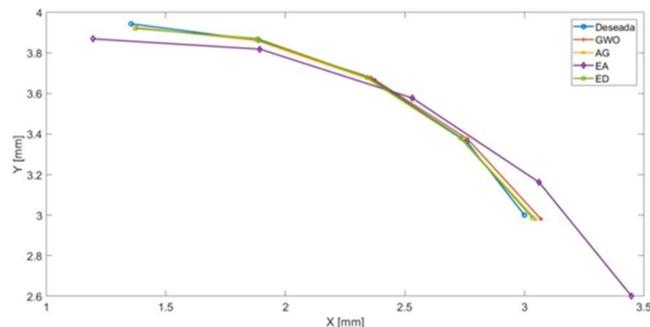


Fig. 2. Trayectoria deseada vs generadas por los algoritmos AG, ED, GWO, EA.

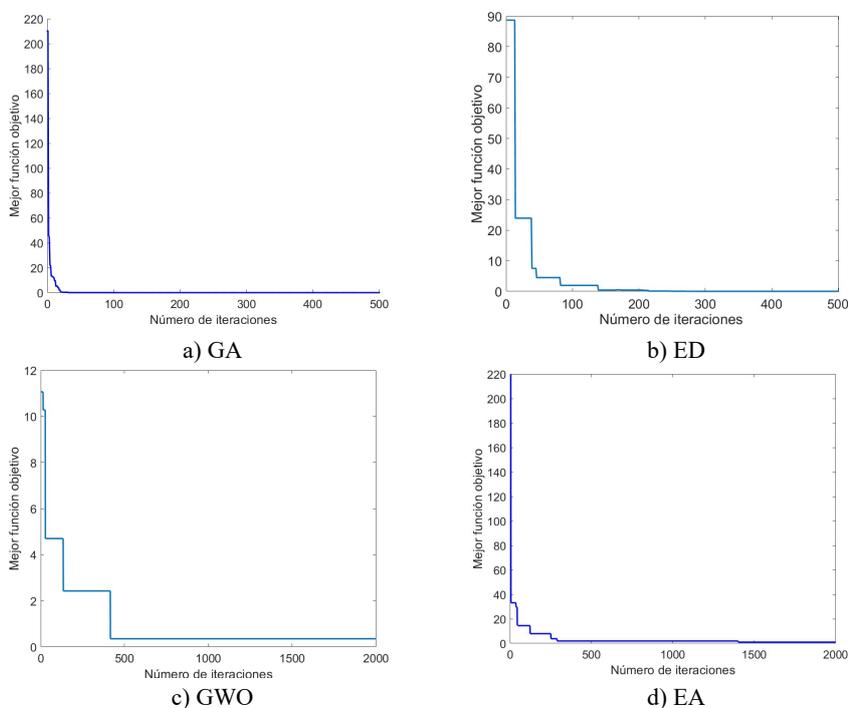


Fig. 3. Respuesta de los algoritmos en la mejor solución.

- Límites de las variables: $a_1 \in [20,50]$, $a_2 \in [6,16]$, $x_0 \in [-5,15]$, $y_0 \in [-20,20]$, $\beta^1 \in [-65,180]$ y $\theta^5 \in [0,2\pi]$.
- Parámetros utilizados en GA: Generaciones = 1000, cruce = 0.6, mutación = 0.1, población = 200.
- Parámetros utilizados en el DE: Generaciones = 500, cruce = 0.2, factor de escalamiento = 0.5, mutación = 0.8, población = 100.

Tabla 4. Trayectoria deseada para el dedo.

Coordenadas					
X	0	-17.95	-35.92	-38.03	34.77
Y	47.76	34.4	21.67	7.36	-12.43

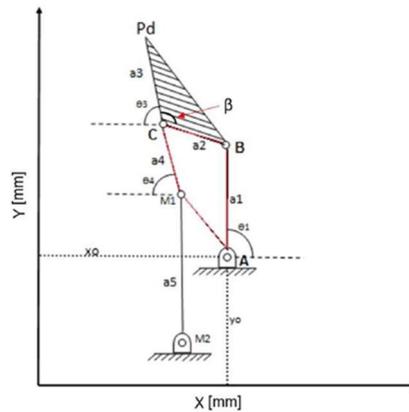


Fig. 4. Diagrama cinemático del mecanismo [12].

Se muestran los vectores solución, el primero corresponde al obtenido por GA y el segundo al obtenido utilizando el DE:

$$S_{ol\ GA} = [43.43\ 10.00\ 1.80\ -19.99\ 1.9087\ 1.235].$$

$$S_{ol\ DE} = [46.25\ 13.20\ 2.21\ -17.95\ 2.0171\ 1.4024].$$

En la Fig. 5 se muestran tres gráficas que representan la trayectoria deseada, la obtenida por el autor en [12] y la generada utilizando el algoritmo DE. Se puede observar que los resultados obtenidos mejoraron respecto a los presentados en [12].

El error reportado es de 13.30 mm y el obtenido se redujo hasta 4.70 [mm].

Caso 3: Optimización de un mecanismo para una pierna biónica

El problema consiste en encontrar las dimensiones del mecanismo de la Fig. 6, para la simulación del movimiento de una pierna centrándose en el movimiento de la rodilla. Las ecuaciones que describen el mecanismo son:

$$x_{k1} = l_1 \cos \theta_1, \tag{8}$$

$$y_{k1} = l_1 \sin \theta_1, \tag{9}$$

$$x_{k2} = l_2 \cos \theta_2, \tag{10}$$

$$y_{k3} = l_2 \sin \theta_2, \tag{11}$$

$$x_{k4} = l_1 \cos \theta_1 + l_4 \cos \theta_4, \tag{12}$$

$$y_{k4} = l_1 \sin \theta_1 + l_4 \sin \theta_4. \tag{13}$$

Tabla 5. Trayectoria deseada.

Coordenadas X	Coordenadas Y	Coordenadas X	Coordenadas Y
2	133	14	129
5	133	16	127
7	132	18	124
9	132	21	118
11	131	26	107
12	130	31	87

Las coordenadas del ICR se puede escribir como:

$$x_{P_{ICK}} = \frac{y_{K_2} - y_{K_3} + ax_3 - bx_{K_2}}{a - b}, \quad (14)$$

$$y_{P_{ICK}} = \frac{x_{K_2} - x_{K_3} + a^{-1}x_{K_3} - b^{-1}x_{K_2}}{a^{-1} - b^{-1}}, \quad (15)$$

donde:

$$a = \frac{y_{K_4} - y_{K_3}}{x_{K_4} - x_{K_3}}, \quad (16)$$

$$b = \frac{y_{K_2} - y_{K_1}}{x_{K_2} - x_{K_1}}. \quad (17)$$

En la Tabla 5 se tienen las coordenadas en X y Y para el seguimiento de trayectoria presentados en [15].

El problema de optimización se define por [15]:

- Variables de diseño: $[\theta_1^1, \dots, \theta_1^{12}, l_1, l_2, l_3, l_4]$,
- Puntos deseados: $[(P_{xd}^1, P_{xd}^{12}), \dots, (P_{yd}^1, P_{yd}^{12})]$ mostrados en la Tabla 5.
- Límites de las variables: $l_1 \in [40,60]$, $l_2 \in [60,55]$, $l_3 \in [35,55]$, $l_4 \in [20,40]$ y $\theta_5 \in [0,0.9]$
- Parámetros utilizados en GA: Generaciones = 1000, cruce = 0.8, mutación = 0.1, población = 20
- Parámetros utilizados en el DE: Generaciones = 150, cruce = 0.2, factor de escalamiento = 0.5, mutación = 0.8, población = 50

Se muestran los vectores solución los cuales representan l_1, l_2, l_3, l_4 y θ_1 . El primero corresponde al obtenido por GA y el segundo al obtenido utilizando el DE:

$$S_{ol\ GA} = [57.0100\ 43.9000\ 41.8200\ 23.9300\ 0.8203],$$

$$S_{ol\ DE} = [41.3036\ 49.6885\ 53.8985\ 32.4434\ 1.5532].$$

En la Fig. 7 se muestran las trayectorias, que sigue el miembro inferior a través del CIR, la generada por el GA y la obtenida utilizando el DE, de las cuales se realizaron 31 corridas para tener una mejor evaluación de resultados.

El error reportado por el autor es de 2.3408 mm y el error obtenido en este trabajo fue de 0.0135 mm, es decir que se redujo un 99.42%.

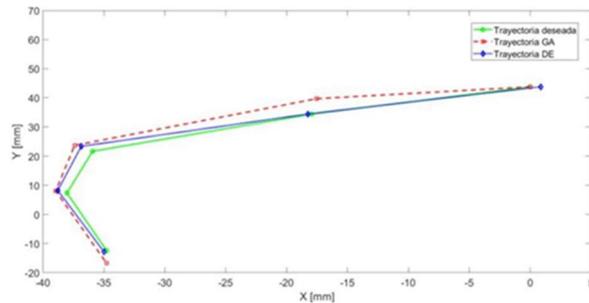


Fig. 5. Trayectorias obtenidas vs deseada.

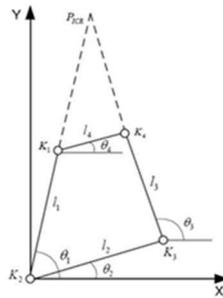


Fig. 6. Diagrama esquemático del mecanismo para rodilla [15].

Caso 4 Optimización de un exoesqueleto

Basado en [9], se hace la optimización de un mecanismo de cuatro barras utilizando un Algoritmo Evolutivo Diferencial para un exoesqueleto de miembro inferior con trayectorias naturales de la marcha humana.

En los procedimientos de síntesis convencionales, la articulación de la cadera debe estar fija y los datos se transforman en relación con esta, sin embargo, en [9] el eslabón se mueve desde la izquierda hacia el origen como se muestra en la Fig. 8.

El exoesqueleto con rodilla de cuatro barras tiene los eslabones 1, 2, 3 y 4 como se muestra en la Fig. 8 A y están definidos por CD, CE, EF y DF respectivamente. Los puntos P y A representan las uniones de la cadera y el tobillo. Las longitudes de los eslabones están denotadas por r_1 , r_2 , r_3 y r_4 , y su orientación por $(270^\circ + \theta_1)$, θ_2 , θ_3 y θ_4 , como se muestra en la Fig. 8 B.

Las longitudes de los eslabones extendidos AB y PG están denotados por l_1 y l_2 , los cuales están a 90° respecto a CD y EF. El eslabonamiento CDEF forman una cadena cerrada, donde θ_1 y θ_2 son los ángulos desconocidos en el ciclo de marcha. Los ángulos θ_3 y θ_4 de los enlaces EF y FD, respectivamente, pueden determinarse con la ecuación de enlace de la cadena cerrada.

Las coordenadas del tobillo A (A_{xd} , A_{yd}) se toman como entrada y las coordenadas C, E, G y P se derivan del marco global XOY, que se puede expresar como:

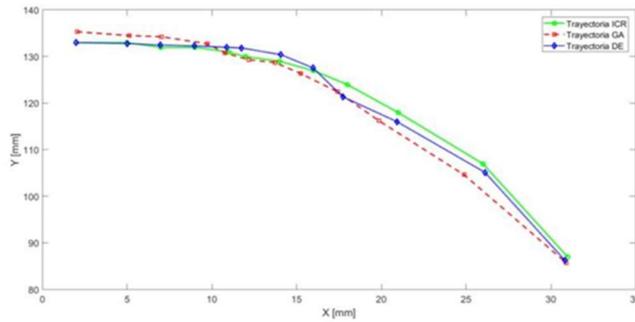


Fig. 7. Trayectorias obtenidas vs deseada.

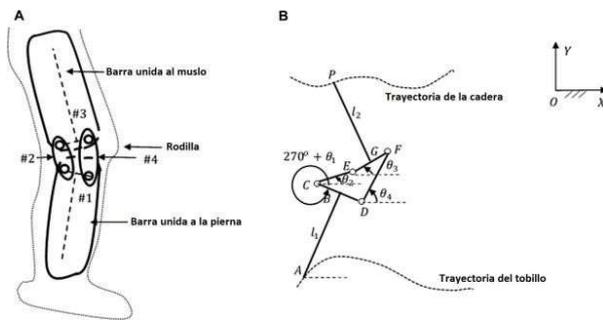


Fig. 8. (A) Exoesqueleto de cuatro barras para miembro inferior y (B) Sistema vectorial para exoesqueleto de cuatro barras en rodilla [9].

$$B_x = A_{xd} + l_1 \cos \theta_1, \quad (18)$$

$$B_y = A_{yd} + l_1 \sin \theta_1, \quad (19)$$

$$C_x = B_x + \frac{r_1}{2} \cos \left(\frac{\pi}{2} + \theta_1 \right), \quad (20)$$

$$C_y = B_y + \frac{r_1}{2} \sin \left(\frac{\pi}{2} + \theta_1 \right), \quad (21)$$

$$E_x = C_x + r_2 \cos \theta_2, \quad (22)$$

$$E_y = C_y + r_2 \sin \theta_2, \quad (23)$$

$$G_x = E_x + \frac{r_3}{2} \cos \theta_3, \quad (24)$$

Tabla 6. Puntos de precisión seleccionados en la curva de ajuste.

Posición	P_{xd}	P_{yd}	A_{xd}	A_{yd}
1	-3685.9	-1159	-3685.9	-2082.6
2	-3519.6	-1177.5	-3523.6	-1890.6
3	-3353.2	-1198.1	-3361.4	-1859
4	-3186.9	-1207.1	-3199.1	-1878.3
5	-3020.6	-1191.9	-3036.8	-1902.4
6	-2854.2	-1160	-2874.5	-1976.7
7	-2687.9	-1157.7	-2712.3	-2027.3
8	-2521.5	-1187.8	-2550	-2028.8
9	-2355.2	-1205.2	-2387.7	-2050.9
10	-2188.9	-1190	-2225.5	-2091.8

$$G_y = E_y + \frac{r_3}{2} \sin \theta_3. \quad (25)$$

La trayectoria del punto P es obtenida a partir de las ecuaciones (18) a la (25) y para obtener los puntos generados se plantean las ecuaciones (26) y (27):

$$P_x = G_x + l_2 \cos \left(\frac{\pi}{2} + \theta_3 \right), \quad (26)$$

$$P_y = G_y + l_2 \sin \left(\frac{\pi}{2} + \theta_3 \right). \quad (27)$$

Para realizar la síntesis del mecanismo de 4 barras se considera una optimización para $2N$ puntos de precisión con $4N + 6$ variables de diseño, $r_1, r_2, r_3, r_4, l_1, l_2, \theta_1^k$ y θ_2^k , donde $k = 1, \dots, N$ posiciones. Para reducir la complejidad del problema, en [9] lo dividen en dos etapas, considerando $2N$ puntos de precisión con $2N + 6$ variables en la primera etapa, $r_1, r_2, r_3, r_4, l_1, l_2, \theta_1^k$ y θ_2^k , donde $k = 1, \dots, N$ posiciones y $N = 10$.

Aquí se calculan las dimensiones de los eslabones $r_1, r_2, r_3, r_4, l_1, l_2$, y los ángulos de entrada θ_1^k y θ_2^k , donde $K = 1, \dots, N$. En la segunda etapa se calculan los siguientes puntos de precisión solo con θ_1^k y θ_2^k , donde $K = N + 1, \dots, N2$. La trayectoria para seguir es proporcionada por la cadera (P_{xd}, P_{yd}) y el pie (A_{xd}, A_{yd}), las cuales se muestran en la Tabla 6. El problema de optimización está definido por [9] como sigue.

Primera etapa:

- Variables de diseño: $[\theta_1^1, \dots, \theta_1^N, \theta_2^1, \theta_2^N, r_1, r_2, r_3, r_4, l_1, l_2]$.
- Puntos deseados: $[(P_{xd}^1, P_{yd}^1), \dots, (P_{xd}^5, P_{yd}^5)]$ mostrados en la Tabla 6.
- Límites de las variables: $r_1, r_2, r_3, r_4 \in [40, 170], l_1 \in [280, 500], l_2 \in [280, 400], \theta_1^k \in [0, \pi/2]$ y $\theta_2^k \in [0, 2\pi]$.

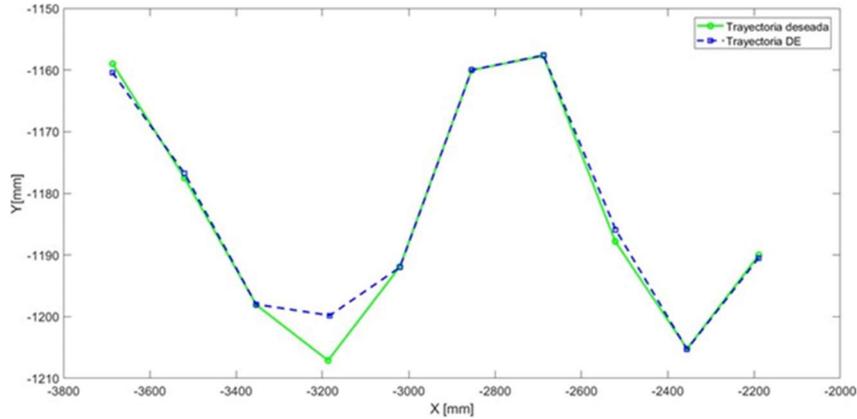


Fig. 9. Trayectoria deseada y trayectoria generada con DE del punto P en la cadera.

- Parámetros utilizados en el DE: Generaciones = 1000, cruce = 0.2, factor de escalamiento = 0.5, mutación = 0.8, población = 1000.

Segunda etapa:

- Variables de diseño: $[\theta_1^{N+1}, \dots, \theta_1^{2N}, \theta_2^{N+1}, \theta_2^{2N}]$.
- Puntos deseados: $[(P_{xd}^6, P_{xy}^6), \dots, (P_{xd}^{10}, P_{xy}^{10})]$ mostrados en la Tabla 6.
- Límites de las variables: $\theta_1^k \in [0, \pi/2]$ y $\theta_2^k \in [0, 2\pi]$.

El mejor mecanismo encontrado con Optimización por Enjambre de Partículas (PSO por sus siglas en inglés) [9] es:

$$r_1 = 72.4620, r_2 = 104.4731, r_3 = 123.6035, r_4 = 119.0642, l_1 = 471.0019, l_2 = 374.0438$$

$$[\theta_1^1, \dots, \theta_1^{2N}] = [1.2374, 0.7530, 0.6750, 0.6825, 0.7323, 0.9179, 1.0401, 0.9637, 0.9705, 1.1220].$$

$$[\theta_2^1, \dots, \theta_2^{2N}] = [1.6622, 2.4527, 2.6601, 2.6131, 2.4457, 1.9755, 1.7669, 1.8648, 1.8414, 1.6623].$$

El mejor mecanismo encontrado con el Algoritmo Evolución Diferencial propuesto es:

$$r_1 = 87.22, r_2 = 162.30144, r_3 = 110.0024, r_4 = 44.3247, l_1 = 394.6355, l_2 = 387.5617$$

$$[\theta_1^1, \dots, \theta_1^{2N}] = [1.2847, 0.7664, 0.7355, 1.1819, 1.2314, 0.7954, 0.9667, 1.1264, 1.0022, 1.1406].$$

$$[\theta_2^1, \dots, \theta_2^{2N}] = [2.1013, 1.2685, 3.24053, 5.7739E-4, 0.07061, 2.1068028229425786, 2.1488, 2.6364, 2.4905, 2.1806].$$

Para obtener la mejor solución en el seguimiento de la trayectoria de la cadera respecto al pie, en [9] hicieron 25 corridas empleando los algoritmos de optimización PSO y TLBO (Teaching-Learning-Based Optimization) con los cuales la mejor respuesta la obtuvieron con PSO en dos etapas.

En la primera etapa obtuvieron un error de 5.2×10^{-2} y en la segunda de 9.92×10^{-8} , teniendo un error total por los 10 puntos de 0.0520 [mm]. Optimizando el mismo

problema con DE, se hicieron 25 corridas sin dividir la trayectoria, es decir, se optimizaron las 26 variables ($r_1, r_2, r_3, r_4, l_1, l_2, \theta^1$ y θ^{10}) en una sola etapa.

Se establecieron 1000 generaciones con una población de 1000 obteniendo un error de 0.0694 mm en la mejor solución donde se observa que la trayectoria generada se aproxima a la trayectoria deseada (Fig. 9). Aunque no se generó una mejora en la solución que se reporta en [9] con PSO, DE presenta un mejor rendimiento en el consumo de recursos computacionales, ya que se necesita un menor número de iteraciones para encontrar la mejor solución.

Como se observó en general, comparando todos los resultados, utilizando la técnica de ED se obtienen resultados más cercanos al error cero, teniendo precisiones de milésimas.

4. Conclusiones y trabajo futuro

Como se observa los mecanismos de cuatro barras son ampliamente utilizados en diversas aplicaciones como en la rehabilitación. Para obtener una buena síntesis que permita el seguimiento de trayectorias específicas, se han utilizado técnicas como los métodos metaheurísticos, de los cuales se han seleccionado en este trabajo los más representativos para demostrar la versatilidad de estas técnicas, ya que fue posible resolver seguimientos de trayectorias con 5, 6 y 10 puntos de precisión. Como se observó, los algoritmos de Evolución Diferencial muestran ventajas respecto a la obtención de un error en el orden de 1×10^{-3} de 98.86 %, 96.68 % y 100 % comparado con los AG, WGO, EA respectivamente.

Esto no indica que las otras técnicas sean malas, solo se observa que para la síntesis de mecanismos los DE ofrecen mejores resultados. También se demostró que con la optimización en las dimensiones de los mecanismos se pueden generar trayectorias como el movimiento de cadera, de rodilla o de un dedo con una precisión de 0.14325 comparada con el del humano. Como trabajo futuro se analizará el comportamiento de los ED con diferentes técnicas de mutación, aplicadas al diseño de una prótesis de rodilla policéntrica para determinar cuál tiene el error mínimo en el seguimiento de la poloide.

Referencias

1. Ponce-Cruz, P., Molina-Gutiérrez, A., Ramírez-Mendoza, R., Méndez-Flores, E., Ortiz-Espinoza, A. A., Balderas-Silva, D. C.: A practical approach to metaheuristics using LabView and MatLab. *Computer Science, Engineering & Technology*, (2020) doi: 10.1201/9780429324413
2. Csiszar, S.: Optimization algorithms (survey and analysis). In: 2007 International symposium on logistics and industrial informatics. IEEE, pp. 185–188 (2007)
3. Desale, S., Rasool, A., Andhale, S., Rane, P.: Heuristic and Meta-Heuristic Algorithms and their relevance to the real world: A Survey. *International journal of computer engineering in research trends*, vol. 2, no. 5, pp. 296–304 (2015)

4. Mendez, E.: Electric machines control optimization by a novel geo-inspired earthquake metaheuristic algorithm. *Nanotechnology Instrumentation and Measurement*, pp. 1–6 (2018) doi: 10.1109/NANO FIM.2018.8688616
5. Mirjalili, S., Mohammad, S., Lewis, A.: Grey wolf optimizer. *Advances in engineering software*, vol. 69, pp. 46–61 (2014)
6. Topcuoglu, H. R., Ucar, A., Altin, L.: A hyper-heuristic based framework for dynamic optimization problems. *Applied Soft Computing*, vol. 19, pp. 236–251 (2014) doi: 10.1016/j.asoc.2014.01.037
7. Cabrera, J. A., Simon, A., Prado, M.: Optimal synthesis of mechanisms with genetic algorithms. *Mech. Mach. Theory*, vol. 37, no. 10, pp. 1165–1177 (2002) doi: 10.1016/S0094-114X(02)00051-4
8. Cuevas-Jiménez, E. V., Osuna-Enciso, J. V., Oliva-Navarro, D. A., Díaz Cortés, M. A.: *Optimización - Algoritmos Programados con MATLAB*. Alfaomega (2016)
9. Singh, R., Chaudhary, H., Singh, A. K.: A novel gait-based synthesis procedure for the design of 4-bar exoskeleton with natural trajectories. *Journal of Orthopaedic Translation*, vol. 12, pp. 6–15 (2018) doi: /10. 1016/j.jot.2017.09.001
10. Slesongsom, S., Panagant, N., Bureerat, S.: Synthesis of four-bar linkage motion generation using optimization algorithms. *Advances in Computational Design*, vol. 4, no. 3, pp. 197–210 (2019) doi: 10.12989/acd.2019.4.3.000
11. Slesongsom, S., Bureerat, S.: Optimal synthesis of four-bar linkage path generation through evolutionary computation with a novel constraint handling technique. *Computational Intelligence and Neuroscience*, vol. 2018 (2018) doi: 10.1155/2018/5462563
12. Espinosa-Garcia, F. J., Tapia-Herrera, R., Lugo-González, E., Arias-Montiel, M.: Development of a robotic hand based on a palm with a metamorphic mechanism for extending the thumb's functionality. *J. Brazilian journal of the Brazilian Society of Mechanical Sciences and Engineering*, vol. 43 (2021) doi: 10.1007/s40430-021-03094-2
13. Bapat, G. M., Sujatha, S.: A method for optimal synthesis of a biomimetic four-bar linkage knee joint for a knee-ankle-foot orthosis. *Journal of Biomimetics, Biomaterials and Biomedical Engineering*, vol. 32, pp. 20–28 (2017) doi: 10.4028/www.scientific.net/JBBBE.32.20
14. Nejlaoui, M., Alateyah, A. I., El-Garaihy, W. H.: Defect free optimization of a polycentric prosthetic knee design using imperialist competition-inspired optimization method. *Journal of Engineering Research*, pp. 1–15 (2021)
15. Xie, H., Wang, S., Li, F.: Knee joint optimization design of intelligent bionic leg based on genetic algorithm. *Int. J. Bioautomation*, vol. 18, no. 3, pp. 195–206 (2014) doi: 10.36909/jer.13063
16. Singh, R.: A Novel gait-inspired four-bar lower limb exoskeleton to guide the walking movement. *Journal of mechanics in Medicine and Biology*, vol. 19, no. 4, pp. 1–24 (2019) doi: 10.1142/S0219519419500192
17. Gao, M., Wang, Z., Li, S., Li, J., Pang, Z., Liu, S., Duan, Z.: Design and optimization of exoskeleton structure of lower limb knee joint based on cross four-bar linkage. *AIP Advances*, vol. 11, no. 6 (2021) doi: 10.1063 /5.0053899

Detección del alcance de las negaciones en español usando Conditional Random Fields

Antonio Tamayo, Jason Angel, Alexander Gelbukh

Instituto Politécnico Nacional (IPN),
Centro de Investigación en Computación (CIC),
Ciudad de México, México

{ajtamayoh, ajason08}@gmail.com, www.gelbukh.com

Resumen. Detectar automáticamente el alcance de las negaciones en los textos es un tema de mucha importancia para varias tareas del procesamiento de lenguaje natural como el análisis de sentimientos, la recuperación de información y la extracción de información. El presente trabajo propone un enfoque híbrido basado en el modelo Conditional Random Fields con características sintácticas y el uso de reglas del mismo tipo para resolver el problema del alcance de las negaciones en un corpus de reseñas de películas, libros y productos escritas en español. El sistema propuesto alcanza 0,77 en la medida Macro-F1 para esta tarea, superando a un fuerte baseline que obtuvo 0,73.

Palabras clave: Alcance de la negación, Conditional Random Fields, reglas sintácticas.

Negation Scope Detection in Spanish Using Conditional Random Fields

Abstract. Automatic detection of negation scopes in texts is a relevant topic for various natural language processing tasks such as sentiment analysis, information retrieval and information extraction. This work presents a hybrid approach based on the model called Conditional Random Fields, using both syntactic features and rules to solve the negation scope detection issue in a corpus of films, books and products reviews written in Spanish. The system proposed achieved 0.77 Macro-F1, overcoming a strong baseline which obtains 0.73 for the same task.

Keywords: Negation scope, Conditional Random Fields, syntactic rules.

1. Introducción

La negación es un fenómeno lingüístico mediante el cual se invierte el valor de verdad de la unidad lingüística (proposición, sintagma o palabra) a la que se aplica [1]; sin embargo, hay palabras que en determinados contextos funcionan como indicadores de negación, pero en otros, no niengan o funcionan como indicadores de contraste o de comparación. Para analizar este fenómeno es necesario definir los siguientes dos aspectos fundamentales: a) el indicador de negación y b) el alcance (también llamado ámbito o dominio en algunos casos) de dicho indicador. También se suele hacer énfasis en el foco, que se refiere a la palabra o sintagma dentro del alcance que se niega explícitamente, y al evento, que se refiere al proceso, acción o estado afectado por la negación. En la figura 1, se muestra un ejemplo de una oración con presencia de negación, con su indicador, alcance y evento debidamente marcados.

```
<sentence>
  A veces enfría cuando hay que calentar, a veces falla el motor de la ventilación forzada y
  <neg_structure>
    <scope>
      <negexp>
        no
      </negexp>
      <event>
        puedo desempañar
      </event>
      el cristal delantero
    </scope>
  </neg_structure>
  ;
</sentence>
```

Fig. 1. Ejemplo de negación con alcance y evento marcados.

La importancia de determinar automáticamente el alcance de las negaciones ha sido evidenciada en múltiples tareas como: extracción de información [2], traducción automática [3], análisis de sentimientos [?,5,6], entre otros. Sin embargo, esta no es una tarea sencilla, ya que presenta retos considerables al interactuar con otros fenómenos lingüísticos, requiriendo un análisis textual profundo para su tratamiento [7]. Entre estos retos se encuentra la dependencia de la lengua del texto o del dominio del mismo; esto no es un aspecto menor, puesto que “el carácter idiosincrático de la expresión de la negación en cada lengua requiere un análisis lingüístico específico” [1]. También son retos los casos de negaciones anidadas y la dependencia de una correcta marcación previa de los indicadores de negación; esto último con sus propios escollos, ya que un indicador puede tener diferentes categorías gramaticales, puede encontrarse a nivel sintáctico, léxico o morfológico y puede ser de una palabra o de varias (incluso discontinuas). Adicionalmente, en la negación se puede presentar gradación (con incrementadores y decrementadores), negaciones comparativas (i.e., ... no es todo lo ...), frases hechas que expresan negación (i.e., ni lo sueñes) o estructuras negativas que no expresan negación (i.e., no me iré hasta no verte). Además, en el español se encuentran negaciones reforzadas mediante otras partículas de negación [1]. Por

ejemplo:

Mi coche, marca Chevy, es genial, **no** me ha fallado **jamás**.

En este trabajo se presentan dos modelos para marcar alcance de las negaciones, en un conjunto de datos de reseñas de películas, libros y productos escritos en español. El primero es un clasificador que utiliza una regla de puntuación simple, y se toma como *baseline*; el segundo es nuestra propuesta, un modelo basado en *Conditional Random Fields* (CRF) más un procesamiento posterior sencillo usando patrones morfosintácticos.

Este artículo está organizado de la siguiente forma: la sección 2 muestra los trabajos similares al nuestro tanto en español como en inglés. En la sección 3 se encuentra la descripción del conjunto de datos usado para en este trabajo. Los experimentos realizados son presentados en la sección 4. La sección 5 muestra los resultados encontrados y el análisis de los mismos. Finalmente, en la sección 6 se presentan las conclusiones y el trabajo futuro en la línea de investigación concerniente a la detección del alcance de las negaciones.

2. Trabajos relacionados

La detección del alcance de la negación ha sido bien explorada en el idioma inglés, sin embargo, existen muy pocos trabajos para el español, pese a ser la segunda lengua con mayor número de hablantes en el mundo¹. Mencionaremos aquí los principales trabajos llevados a cabo para solucionar esta tarea, desde cuatro enfoques: aplicación de reglas, usando aprendizaje automático supervisado con modelos clásicos, híbridos entre los dos anteriores y usando aprendizaje profundo.

2.1. Enfoque basado en reglas

El trabajo de Martí et al. (2016) [1], es un referente en cuanto a la definición y el análisis de los diferentes patrones de negación y su tipología en español. De igual manera lo es, el trabajo realizado alrededor de la negación y sus alcances, por Cotik et al. (2016) [8], en el que se implementan diferentes algoritmos para detectar si un término de interés se encuentra dentro del alcance de una negación en el dominio de la radiología, para lo cual usaron características sintácticas como categorías gramaticales, árboles sintácticos y de dependencia y una adaptación del sistema NegEx basado en reglas, propuesto por Chapman et al. (2001) [9]. Por su parte, el trabajo de Jimenez-Zafra et al. (2018) [10], representa una contribución muy importante para los avances en los estudios de la negación en español, por su trabajo de anotación de un corpus en 8 diferentes dominios con sus negaciones, sus respectivos alcances y otras características importantes de este fenómeno, y que además fue adaptado para analizar el rol de la negación en

¹ Fuente: <https://www.ethnologue.com/>

el análisis de sentimientos en español. Para el inglés, cabe destacar los trabajos de Gindl et al. (2008) [11] con su método llamado *NegHunter* para detectar conceptos negados en documentos clínicos; Harkema et al. (2009) [12] con el algoritmo llamado *ConText* para determinar condiciones clínicas negadas dentro de reportes clínicos; Apostolova et al. (2011) [13], quienes propusieron un sistema llamado *ScopeFinder*, basado en reglas y patrones léxico-sintácticos extraídos automáticamente, para detectar el alcance de las negaciones y especulaciones en el corpus Bioscope [14]; Ballesteros et al. (2012), quienes introdujeron el análisis de dependencia y el sintáctico, mostrando sus capacidades para la detección de la negación; y finalmente, Mehrabi et al. (2015), quienes desarrollaron el algoritmo DEEPEN, el cual incorpora relaciones de dependencia al sistema NegEx [9], para detectar negaciones en textos clínicos.

2.2. Enfoque basado en aprendizaje automático clásico

La tarea de detectar las negaciones y sus alcances en un texto también ha sido abordada usando aprendizaje automático supervisado con modelos clásicos. En esta línea se destacan los trabajos de Rokach et al. (2008) [15], quienes usaron una arquitectura de clasificadores en cascada con árboles de decisión, Morante et al. (2008) [16], Morante y Daelemans (2009) [17], Cruz et al. (2012) [18] y Wu et al. (2014) [19], los cuales combinaron diferentes modelos de aprendizaje automático, Councill et al. (2010) [20], con su sistema basado en CRF, indicadores de negación y dependencia sintáctica, Agarwal y Yu (2010) [21], también usando CRF, Zhu et al. (2010) [22], con su trabajo de análisis semántico, en el cual los indicadores de negación son tratados como predicados y sus alcances como los argumentos de dichos indicadores, Zou et al. (2013) [23] y Shivade et al. (2015) [24], con sus sistemas basados en kernel y características sintácticas, Cruz et al. (2016) [4] y Attardi et al. (2015) [25], quienes usaron árboles de dependencia sintáctica.

2.3. Enfoque híbrido entre aprendizaje automático clásico y reglas

Goryachev et al. (2006) [26] fueron los primeros en probar sistemas combinando reglas con modelos como Naïve Bayes y máquinas de vectores de soporte (SVM, por sus siglas en inglés), modificando el sistema NegEx [9], sin conseguir buenos resultados. También se destacan los trabajos de Huang y Lowe (2007) [27], combinando expresiones regulares y análisis gramatical, Fujikawa et al. (2013) [28], con el sistema llamado *NegFinder*, que combina métodos estadísticos y heurísticos, White (2012) [29], quienes usaron el modelo *Conditional Random Field* (CRF) y expresiones regulares, así como Gyawali y Solorio (2012) [30] que usaron Máquinas de Vectores de Soporte y expresiones regulares, Reitan et al. (2015) [5], usaron CRF y una lista de indicadores de negación, y Pröllochs et al. (2017) [31], quienes propusieron una novedosa estrategia basada en aprendizaje por refuerzo.

2.4. Enfoque basado en aprendizaje profundo

Los avances de los últimos años en el área del aprendizaje profundo para procesamiento de lenguaje natural, se han aplicado para la detección de las negaciones y sus alcances. Entre los trabajos más relevantes se encuentran los de Fancellu et al. (2016) [32], quienes probaron las arquitecturas de perceptrón multicapa y LSTM bidireccional, Qian et al. (2016) [33], con su enfoque basado en redes neuronales convolucionales, Lazib et al. (2016) [34], con su método basado en redes neuronales recurrentes, Fei et al. (2020) [35], con su sofisticada arquitectura que combina redes neuronales recursivas y CRF en su capa de predicción, y finalmente, Fabregat et al. (2019) [36], quienes trabajaron con el mismo conjunto de datos del presente trabajo, con una arquitectura que combina redes convolucionales y LSTM, alcanzando 0,72 en la medida F1 (no especifican si micro-F1 o macro-F1) para detección del alcance de las negaciones a nivel de token.

3. Conjunto de datos de entrenamiento

El presente trabajo fue realizado usando el corpus SFU Review SP-NEG [10], proporcionado para la tarea B de las competencias NEGES 2018 [37] y NEGES 2019 [38]. Dicha tarea no consistió en detectar el alcance de las negaciones, sino determinar su influencia en el problema de análisis de sentimientos. No obstante, el conjunto de datos cuenta con 3,078 oraciones, cada una de las cuales está marcada con sus respectivos indicadores de negación y sus alcances, facilitando así su uso para el trabajo propuesto en este artículo. Siguiendo con la mecánica de la competencia NEGES, para este trabajo se conservó la partición del conjunto de datos en: a) entrenamiento, b) desarrollo y c) validación. En la tabla 1, se observa la distribución de las oraciones con negación para las tres particiones por dominio.

Tabla 1. Cantidad de oraciones con negación en el conjunto de datos por dominio.

Dominio	Entrenamiento	Desarrollo	Validación
Coches	175	37	48
Hoteles	187	51	46
Lavadoras	183	31	61
Libros	312	111	219
Móviles	224	75	40
Música	192	25	63
Computadoras	118	39	73
Películas	523	88	157
Total	1914	457	707

El conjunto de datos se encuentra en formato XML; para nuestro trabajo se realizó un preprocesamiento, ya que para detectar el alcance de las negaciones

se usó el esquema BIO [39], *begin* (B), *inside* (I), *outside* (O), para marcar la unidad léxica que determina el inicio del alcance de un indicador de negación (B), la(s) unidad(es) léxica(s) dentro de dicho alcance (I) y las que están fuera del alcance de algún indicador de negación (O). Hay que precisar que las negaciones presentan retos como: a) múltiples instancias de negación (indicadores), b) alcances de negación anidados y c) alcances discontinuos. En el presente trabajo, se tiene en cuenta el primero, pero no el segundo ni el tercero. En el siguiente ejemplo se ilustra el esquema de representación usado:

pero “ el orfanato ” no te engancha , ni consigue intrigar y mucho menos dar miedo .

[O, O, O, O, O, B, I, I, O, B, I, I, O, B, I, I, O]

4. Experimentos

4.1. Baseline

El siguiente *baseline* se definió como punto de partida para marcar el alcance de las negaciones:

El alcance inicia a partir de la aparición del indicador de negación (o si es multipalabra, a partir de la aparición de su primera palabra) y va hasta la unidad léxica que está inmediatamente antes del primer signo de puntuación que se encuentre (de izquierda a derecha en el texto). Veamos un ejemplo:

“vamos, por 11900 euros yo no me lo compraba.”

[O, O, O, O, O, O, B, I, I, I, O]

En este ejemplo, el indicador de negación es la palabra “no”, por lo tanto, el *baseline* la marca con 'B' y a las palabras “me”, “lo” y “compraba”, las marca con 'I'.

4.2. Modelo

El modelo propuesto es *Conditional Random Fields* [40] (CRF), el cual fue entrenado usando características lingüísticas y la siguiente configuración:

Características: En la tabla 2 se observa el nombre, el tipo y la descripción de las características usadas para entrenar el modelo CRF.

Todas las características descritas en la tabla 2, fueron extraídas usando la librería SpaCy³. Es importante aclarar que estas características se encuentran definidas a nivel de palabra (o unidad léxica), por lo tanto, cada oración del

³ <https://spacy.io/api/token>

Tabla 2. Características usadas para el entrenamiento del modelo CRF.

Característica	Tipo	Descripción
Forma	cadena de texto	La palabra tal como aparece en el texto
Lemma	cadena de texto	La palabra como aparece en los diccionarios
Part-of-Speech	cadena de texto	Categoría gramatical de la palabra
is_cue	buleana	Marca si es un indicador de negación o no ²
ancestors	cadena de texto	Secuencia de descendientes sintácticos
rights	cadena de texto	Los hijos inmediatos a la derecha, en el análisis de dependencia sintáctica
head	cadena de texto	El padre sintáctico de la palabra
r_edge	cadena de texto	El descendiente sintáctico más a la derecha
l_edge	cadena de texto	El descendiente sintáctico más a la izquierda
dep_	cadena de texto	Relación de dependencia sintáctica
indice	númerica	Posición de la palabra en la oración
is_punct	buleana	Marca si es un signo de puntuación o no
is_sent_start	buleana	Marca si es el inicio de la oración o no
subtree	cadena de texto	Secuencia que incluye a la palabra y a todos sus descendientes sintácticos

²La palabra se marca como negación si está dentro de la siguiente lista: [apenas, nada, sin, a_falta.de, ninguno, no_solo, no, ni, ninguna, ningún].

conjunto de datos fue representada como una lista de diccionarios en Python, donde cada diccionario representa una unidad léxica y contiene las 14 características previamente descritas, y cada lista representa una oración y contiene tantos diccionarios como unidades léxicas tiene la oración.

Se realizaron múltiples combinaciones de estas características para finalmente determinar este conjunto como el que mejor representaba los datos para solucionar la tarea. Además, se probaron otras características como la representación de las palabras usando los *word embeddings* de SpaCy y la norma de dichos vectores, pero estas no aportaron información relevante, o empeoraban los resultados.

Configuración: El modelo CRF fue implementado usando la librería *sklearn-crfsuite*⁴ con los siguientes parámetros:

$c_1 = 0,9$; $c_2 = 0,03$; *iteraciones* = 60; *todas_las_posibles_transiciones* = *True*

4.3. Procesamiento posterior

Este procesamiento consistió en aplicar reglas sintácticas a la salida entregada por el modelo CRF, las cuales son descritas mediante el siguiente algoritmo:

⁴ <https://sklearn-crfsuite.readthedocs.io/en/latest/>

```

para cada oración hacer
  |
  |  $i = 0;$ 
  | para cada unidad léxica hacer
  | |
  | | si la unidad léxica en  $i$  está marcada con  $B$  entonces
  | | |
  | | | si se encuentra ADP-DET-(NOUN o PRON) desde  $i - 3$ 
  | | | entonces
  | | | | corregir la salida con: [B, I, I, I] desde la posición  $i - 3;$ 
  | | | fin
  | | | si si se encuentra DET-NOUN-ADJ desde  $i - 3$  entonces
  | | | | corregir la salida con: [B, I, I, I] desde la posición  $i - 3;$ 
  | | | fin
  | | | si si se encuentra DET-(NOUN o PRON) desde  $i - 2$ 
  | | | entonces
  | | | | corregir la salida con: [B, I, I] desde la posición  $i - 2;$ 
  | | | fin
  | | | fin
  | |  $i = i + 1$ 
  | fin
fin

```

Algoritmo 1: Procesamiento posterior.

Con, ADP: preposición; DET: determinante; NOUN: nombre; PRON: pronombre y ADJ: adjetivo.

Aplicando estas reglas a la salida del modelo CRF, se consigue expandir el alcance de las negaciones hacia la izquierda, corrigiendo el inicio del mismo e incluyendo tokens del alcance no detectados por el modelo. Esto es, cuando en la salida del modelo se detecta una unidad léxica marcada con B, se valida si las tres unidades léxicas anteriores presentan alguno de los siguiente patrones morfosintácticos (de izquierda a derecha): ADP-DET-(NOUN o PRON), DET-NOUN-ADJ o DET-(NOUN o PRON); la validación se hace en ese orden, y en caso de cumplirse, se sobrescriben con [B, I, I, I], las posiciones desde $i - 3$ hasta i de la dicha salida, para los dos primeros patrones, y con [B, I, I], las posiciones desde $i - 2$ hasta i , para el último patrón.

5. Resultados y análisis

Para medir los resultados en el alcance de las negaciones se usaron las medidas *Macro-precision* (Pr.), *Macro-recall* (Rec.) y Macro-F1, determinando la clasificación correcta a nivel de palabra (token). Es decir, marcando el número de palabras clasificadas correctamente como parte del alcance de la negación o fuera de él.

En la tabla 3 y en la tabla 4, se observan los resultados obtenidos en la detección de los alcances de la negación usando el conjunto de datos de desarrollo

y validación respectivamente, para el *baseline*, el modelo CRF y CRF extendido con el procesamiento posterior usando reglas sintácticas.

Tabla 3. Resultados en el conjunto de datos de desarrollo.

Dominio	Baseline			CRF			CRF + Reglas		
	Pr.	Rec.	F1	Pr.	Rec.	F1	Pr.	Rec.	F1
Coches	0.68	0.78	0.72	0.75	0.73	0.74	0.73	0.72	0.72
Hoteles	0.66	0.76	0.70	0.83	0.73	0.77	0.81	0.73	0.76
Lavadoras	0.73	0.87	0.78	0.78	0.70	0.74	0.77	0.70	0.73
Libros	0.69	0.79	0.73	0.76	0.72	0.74	0.78	0.74	0.76
Móviles	0.78	0.87	0.81	0.82	0.76	0.78	0.78	0.74	0.75
Música	0.64	0.71	0.67	0.76	0.73	0.74	0.76	0.74	0.75
Ordenadores	0.66	0.73	0.69	0.74	0.69	0.71	0.75	0.71	0.73
Películas	0.71	0.81	0.75	0.81	0.74	0.77	0.83	0.76	0.79
Promedio	0.69	0.79	0.73	0.78	0.73	0.75	0.78	0.73	0.75

Tabla 4. Resultados en el conjunto de datos de validación.

Dominio	Baseline			CRF			CRF + Reglas		
	Pr.	Rec.	F1	Pr.	Rec.	F1	Pr.	Rec.	F1
Coches	0.70	0.75	0.72	0.75	0.69	0.72	0.74	0.70	0.72
Hoteles	0.63	0.72	0.66	0.78	0.70	0.74	0.79	0.72	0.75
Lavadoras	0.71	0.81	0.75	0.83	0.77	0.79	0.85	0.80	0.82
Libros	0.66	0.74	0.69	0.77	0.73	0.75	0.80	0.76	0.78
Moviles	0.75	0.88	0.80	0.80	0.80	0.80	0.79	0.879	0.79
Musica	0.68	0.78	0.72	0.72	0.71	0.71	0.73	0.72	0.73
Ordenadores	0.68	0.78	0.72	0.79	0.72	0.75	0.82	0.75	0.78
Peliculas	0.70	0.79	0.74	0.81	0.73	0.76	0.82	0.76	0.79
Promedio	0.69	0.78	0.73	0.78	0.73	0.75	0.79	0.76	0.77

Como se mencionó en la sección de experimentos (sección 4), el uso de los *word embeddings* de SpaCy, no aportó información relevante a nuestro modelo. Por el contrario, las características sintácticas sí aportaron dicha información y ayudan al modelo a resolver el problema de mejor manera. Encontramos además, que nuestro modelo CRF algunas veces no detecta correctamente el inicio de los alcances de las negaciones y en otros casos también corta el alcance mucho antes de lo debido. La primera limitación fue mitigada usando las tres reglas sintácticas definidas en el algoritmo 1, en la sección 4.3. Estas reglas fueron diseñadas analizando los patrones morfosintácticos encontrados en las muestras del conjunto de datos para las cuales el modelo cometía errores. A continuación se muestra un ejemplo para ilustrar esta situación:

Antonio Tamayo, Jason Angel, Alexander Gelbukh

hay que decir también que el ordenador no viene con el cd de el sistema operativo , sino éste está preinstalado .

Salida correcta:

[O, O, O, O, O, O, B, I, I, I, I, I, I, I, I, I, O, O, O, O, O, O]

Salida de CRF:

[O, O, O, O, O, O, O, O, B, I, I, I, I, I, I, I, I, O, O, O, O, O, O]

Salida de CRF + Reglas:

[O, O, O, O, O, O, B, I, I, I, I, I, I, I, I, I, O, O, O, O, O, O]

En el ejemplo anterior, el procesamiento posterior corrige el alcance de la negación al identificar el patrón DET-NOUN.

El segundo problema realmente no es fácil de corregir a partir de reglas, puesto que no se identificó ninguna regularidad sintáctica que permitiera expandir los alcances identificados por el modelo y de esa forma mejorar los resultados. Se probó haciendo la expansión de los alcances identificados por el modelo hasta dónde se encontrara el primer signo de puntuación, emulando al *baseline*, pero esto empeoró los resultados.

Al observar los resultados bajo la medida *recall*, se aprecia que la mayor dificultad de nuestro modelo consiste en cortar los alcances de la negación antes de encontrar la unidad léxica que cierra dichos alcances, redundando en la aparición de falsos negativos e impidiéndole superar al *baseline* en esta medida y así poder presentar un mejor resultado en la medida macro-F1. Esto se aprecia claramente en el siguiente ejemplo:

de el cassette también se rompio ellector a los 6 meses y no lo cubre la garantia .

Salida correcta:

[O, O, B, I, I, I, I, O]

Salida de CRF + Reglas:

[O, O, B, I, I, O, O, O]

Salida del *baseline*:

[O, O, B, I, I, I, I, O]

No obstante, aún bajo las limitaciones mencionadas, nuestros modelos empleando CRF superaron al *baseline* en todos los dominios, excepto en coches y móviles. Esto ocurre porque en estos dominios en particular, la puntuación es un rasgo que permite marcar el alcance de las negaciones con mayor precisión que en los demás dominios, haciendo que la regla en la cual está basado el *baseline*, entregue mejores resultados que el modelo propuesto. Particularmente el modelo extendido con reglas supera al *baseline*, en al menos un 5%, en la mayoría de los dominios, bajo la medida Macro-F1. Notamos además que el empleo de reglas que extiende las capacidades del CRF mejora, o mantiene sus resultados para todos los dominios, a excepción de móviles.

Finalmente, se resalta que nuestro modelo supera en un 5% en la clasificación a nivel de token⁵, al modelo basado en aprendizaje profundo, propuesto por Fabregat et al. (2019) [36], el único trabajo realizado para alcance de las negaciones en español y que usa el mismo conjunto de datos que usamos en el presente trabajo.

6. Conclusiones y trabajo futuro

En este trabajo presentamos un enfoque para detectar el alcance de las negaciones en español. Para ello empleamos un conjunto de datos de reseñas de películas, libros y productos, un campo que ha sido muy poco explorado en la comunidad científica. Se estableció un simple pero fuerte *baseline*, basado en una regla de puntuación, y que mejora los resultados del trabajo que se puede considerar el estado del arte en esta tarea [36]. Se propuso un modelo que combina Conditional Random Fields con características sintácticas, logrando mejorar en un 4% los resultados obtenidos por el *baseline*, en el conjunto de datos de validación. Nuestro trabajo permitió establecer la importancia de los patrones morfosintácticos, al combinarlos con el clasificador CRF. Finalmente, cabe mencionar la contribución de este trabajo para el español, la segunda lengua con mayor número de hablantes en el mundo, y para la que se ha explorado muy poco en la detección del alcance de las negaciones, cuyas aplicaciones son muy relevantes para diferentes tareas del procesamiento de lenguaje natural.

Como trabajo futuro se planea realizar una explotación enfatizada en características (patrones) morfosintácticas, las cuales parecen ser altamente discriminantes en detectar el alcance de la negación.

Se proyecta además, fusionar nuestro modelo CRF con modelos de aprendizaje profundo capaces de reconocer otros patrones inherentes al comportamiento de los alcances de la negación en español.

De igual manera, se abre la puerta para aplicar el modelo presentado en este trabajo en tareas como la extracción de información en dominios tan importantes

⁵ Asumiendo que Fabregat et al. (2019) usaron la medida macro-F1, puesto que si usaron micro-F1, nuestro modelo lo superaría por mucho más.

como el clínico, en el que ya se encuentran disponibles algunos corpus marcados con indicadores de negación y sus alcances [8].

Agradecimientos. Este trabajo fue posible gracias al CONACyT, entidad que otorga las becas doctorales a dos de los autores de este trabajo.

Referencias

1. Martí, M. A., Taulé, M., Nofre, M., Marsó, L., Martín-Valdivia, M. T., Jiménez-Zafra, S. M.: La negación en español: análisis y tipología de patrones de negación. *Procesamiento del Lenguaje Natural* (57), 41–48 (2016)
2. Savova, G. K., Masanz, J. J., Ogren, P. V., Zheng, J., Sohn, S., Kipper-Schuler, K. C., Chute, C. G.: Mayo clinical Text Analysis and Knowledge Extraction System (cTAKES): architecture, component evaluation and applications. *Journal of the American Medical Informatics Association*, 17(5), 507–513 (2010)
3. Baker, K., Bloodgood, M., Dorr, B. J., Callison-Burch, C., Filardo, N. W., Piatko, C., ... Miller, S.: Modality and negation in simt use of modality and negation in semantically-informed syntactic mt. *Computational Linguistics*, 38(2), 411–438 (2012)
4. Cruz, N. P., Taboada, M., Mitkov, R.: A machine-learning approach to negation and speculation detection for sentiment analysis. *Journal of the Association for Information Science and Technology*, 67(9), 2118–2136 (2016)
5. Reitan, J., Faret, J., Gambäck, B., Bungum, L.: Negation scope detection for twitter sentiment analysis. In: *Proceedings of the 6th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pp. 99–108 (2015)
6. Tamayo, A., Londoño, J. A., Burgos, D., Quiroz, G.: Sentiment Analysis of News Articles in Spanish using Predicate Features. *Lenguaje*, 47(2), 235–267 (2019)
7. Díaz, N. P. C., López, M. J. M.: Negation and speculation detection, Vol. 13, John Benjamins Publishing Company (2019)
8. Cotik, V., Stricker, V., Vivaldi, J., Rodríguez Hontoria, H.: Syntactic methods for negation detection in radiology reports in Spanish. In: *Proceedings of the 15th Workshop on Biomedical Natural Language Processing, BioNLP 2016: Berlin, Germany, August 12, 2016*, pp. 156–165, Association for Computational Linguistics (2016)
9. Wendy W. Chapman, Will Bridewell, Paul Hanbury, Gregory F. Cooper, and Bruce G. Buchanan. 2001a. A Simple Algorithm for Identifying Negated Find- ings and Diseases in Discharge Summaries. *Journal of Biomedical Informatics*, 34(5): 301–310 (2001)
10. Jiménez-Zafra, S. M., Taulé, M., Martín-Valdivia, M. T., Ureña-López, L. A., Martí, M. A.: SFU Review SP-NEG: a Spanish corpus annotated with negation for sentiment analysis. A typology of negation patterns. *Language Resources and Evaluation*, 52(2), 533–569 (2018)
11. Gindl, S., Kaiser, K., Miksch, S.: Syntactical negation detection in clinical practice guidelines. *Studies in health technology and informatics*, 136, 187 (2008)
12. Harkema, H., Dowling, J. N., Thornblade, T., Chapman, W. W.: ConText: an algorithm for determining negation, experienter, and temporal status from clinical reports. *Journal of biomedical informatics*, 42(5), 839–851 (2009)

13. Apostolova, E., Tomuro, N., Demner-Fushman, D.: Automatic extraction of lexico-syntactic patterns for detection of negation and speculation scopes. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2, pp. 283–287, Association for Computational Linguistics (2011)
14. Vincze, V., Szarvas, G., Farkas, R., Móra, G., Csirik, J.: The BioScope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC bioinformatics*, 9(11), 1–9 (2008)
15. Rokach, L., Romano, R., Maimon, O.: Negation recognition in medical narrative reports. *Information Retrieval*, 11(6), 499–538 (2008)
16. Morante, R., Liekens, A., Daelemans, W.: Learning the scope of negation in biomedical texts. In: Proceedings of the 2008 conference on empirical methods in natural language processing, pp. 715–724 (2008)
17. Morante, R., Daelemans, W.: A metalearning approach to processing the scope of negation. In: Proceedings of the thirteenth conference on computational natural language learning (CoNLL-2009), pp. 21–29 (2009)
18. Cruz Díaz, N. P., Mana López, M. J., Vázquez, J. M., Álvarez, V. P.: A machine-learning approach to negation and speculation detection in clinical texts. *Journal of the American society for information science and technology*, 63(7), 1398–1410 (2012)
19. Wu, S., Miller, T., Masanz, J., Coarr, M., Halgrim, S., Carrell, D., Clark, C.: Negation's not solved: Generalizability versus optimizability in clinical natural language processing. *PloS one*, 9(11) (2014)
20. Councill, I. G., McDonald, R., Velikovich, L.: What's great and what's not: Learning to classify the scope of negation for improved sentiment analysis. In: Proceedings of the workshop on negation and speculation in natural language processing, pp. 51–59, Association for Computational Linguistics (2010)
21. Agarwal, S., Yu, H.: Biomedical negation scope detection with conditional random field. *Journal of the American medical informatics association*, 17(6), 696–701 (2010)
22. Zhu, Q., Li, J., Wang, H., Zhou, G.: A unified framework for scope learning via simplified shallow semantic parsing. In: Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing, pp. 714–724, Association for Computational Linguistics (2010)
23. Zou, B., Zhou, G., Zhu, Q.: Tree kernel-based negation and speculation scope detection with structured syntactic parse features. In: Proceedings of the 2013 conference on empirical methods in natural language processing, pp. 968–976 (2013)
24. Shivade, C., de Marneffe, M. C., Fosler-Lussier, E., Lai, A. M.: Extending NegEx with kernel methods for negation detection in clinical text. In: Proceedings of the Second Workshop on Extra-Propositional Aspects of Meaning in Computational Semantics (ExProM 2015), pp. 41–46 (2015)
25. Attardi, G., Cozza, V., Sartiano, D.: Detecting the scope of negations in clinical notes. *CLiC it*, 14 (2015)
26. Goryachev, S., Sordo, M., Zeng, Q. T., Ngo, L.: Implementation and evaluation of four different methods of negation detection, pp. 2826–2831, Technical report, DSG (2006)
27. Huang, Y., Lowe, H. J.: A novel hybrid approach to automated negation detection in clinical radiology reports. *Journal of the American medical informatics association*, 14(3), 304–311 (2007)

28. Fujikawa, K., Seki, K., Uehara, K.: NegFinder: A web service for identifying negation signals and their scopes. *Information and Media Technologies*, 8(3), 884–889 (2013)
29. White, J. P.: UWashington: Negation resolution using machine learning methods. In: *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pp. 335–339, Association for Computational Linguistics (2012)
30. Gyawali, B., Solorio, T.: UABCoRAL: a preliminary study for resolving the scope of negation. In: *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, pp. 275–281, Association for Computational Linguistics (2012)
31. Pröllochs, N., Feuerriegel, S., Neumann, D.: Understanding Negations in Information Processing: Learning from Replicating Human Behavior. Available at SSRN 2954460 (2017)
32. Fancellu, F., Lopez, A., Webber, B.: Neural networks for negation scope detection. In: *Proceedings of the 54th annual meeting of the Association for Computational Linguistics (volume 1: long papers)*, pp. 495–504 (2016)
33. Qian, Z., Li, P., Zhu, Q., Zhou, G., Luo, Z., Luo, W.: Speculation and negation scope detection via convolutional neural networks. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 815–825 (2016)
34. Lazib, L., Zhao, Y., Qin, B., Liu, T.: Negation scope detection with recurrent neural networks models in review texts. *International Journal of High Performance Computing and Networking*, 13(2), 211–221 (2016)
35. Fei, H., Ren, Y., Ji, D.: Negation and speculation scope detection using recursive neural conditional random fields. *Neurocomputing*, 374, 22–29 (2020)
36. Fabregat, H., Araujo Serna, L., Martínez Romo, J.: Deep learning approach for negation trigger and scope recognition. *Procesamiento de Lenguaje Natural*, 62, 37–44 (2019)
37. Jiménez-Zafra, S. M., Díaz, N. P. C., Morante, R., Martín-Valdivia, M. T.: Neges 2018: Workshop on negation in spanish. *Procesamiento del Lenguaje Natural*, 62, 21–28 (2019)
38. Jiménez-Zafra, S. M., Cruz Díaz, N. P., Morante, R., Martín-Valdivia, M. T.: NEGES 2019 Task: Negation in Spanish. In: *Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2019)*. CEUR Workshop Proceedings, CEUR-WS, Bilbao, Spain (2019)
39. Ramshaw, L. A., Marcus, M. P.: Text chunking using transformation-based learning. In: *Natural language processing using very large corpora*, pp. 157–176, Springer, Dordrecht (1999)
40. Lafferty, J., McCallum, A., Pereira, F. C.: Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In: *Proceedings of the 18th International Conference on Machine Learning 2001 (ICML 2001)*, pp. 282–289 (2001)

Electronic edition
Available online: <http://www.rcs.cic.ipn.mx>



<http://rsc.cic.ipn.mx>



Centro de Investigación
en Computación