EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

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

Research in Computing Science

Vol. 152 No. 6
June 2023

Research in Computing Science

Research in Computing Science, Año 22, Volumen 152, No. 6, junio de 2023, es una publicación mensual, editada por el Instituto Politécnico Nacional, a través del Centro de Investigación en Computación. Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738, Ciudad de México, Tel. 57 29 60 00, ext. 56571. https://www.rcs.cic.ipn.mx. Editor responsable: Dr. Grigori Sidorov. Reserva de Derechos al Uso Exclusivo del Título No. 04-2019-082310242100-203. ISSN: en trámite, ambos otorgados por el Instituto Politécnico Nacional de Derecho de Autor. Responsable de la última actualización de este número: el Centro de Investigación en Computación, Dr. Grigori Sidorov, Av. Juan de Dios Bátiz S/N, Esq. Av. Miguel Othon de Mendizábal, Col. Nueva Industrial Vallejo, C.P. 07738. Fecha de última modificación 01 de junio de 2023.

Research in Computing Science, year 22, Volume 152, No. 6, June 2023, is published monthly by the Center for Computing Research of IPN.

Volume 152(6)

Advances in Artificial Intelligence

Gilberto Ochoa Ruiz (ed.)

Instituto Politécnico Nacional, Centro de Investigación en Computación
México 2023

ISSN: in process

Formerly ISSNs: 1870-4069, 1665-9899

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

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

Indexed in LATINDEX, DBLP and Periodica

Electronic edition

ISSN 1870-4069

Table of Contents

Page

ISSN 1870-4069

Actividad de agentes robóticos regulada a través de información de temporalidad vía un cálculo lógico de eventos

Manuel Hernández Gutiérrez, Eduardo Sánchez Soto

Universidad Tecnológica de la Mixteca,
Instituto de Computación,
México

{manuelhg,esanchez}@mixteco.utm.mx

Resumen. Conforme las tecnologías del área de la Inteligencia Artificial se desarrollan, será en ocasiones de suma importancia para algunas aplicaciones el agregar a tales tecnologías mecanismos computacionales para tratar el paso del tiempo, según una noción humana de éste. En este artículo aplicamos un cálculo lógico de eventos a agentes robóticos para que tales agentes posean mecanismos computacionales básicos de tratamiento de tiempo. Estudiaremos el caso de una arquitectura multi-agente que depende de medios de comunicación remotos y de ciclos de agentes que se desarrollan en un ambiente cambiante. En esta arquitectura, cada agente pasa de observaciones a razonamientos y, de ahí, a acciones, con una noción adjunta del paso del tiempo.

Palabras clave: Temporalidad, lógica, agentes robóticos, robótica cognitiva.

Activity of Robotic Agents Ruled through Time Information via a Logic-Based Event Calculus

Abstract. With the advancement of Artificial Intelligence, there may be situations where it becomes necessary to incorporate time-processing mechanisms based on human perception into certain applications. Our article proposes a logical computation method for events in robotic agents, enabling them to possess these basic computational time processing mechanisms. We will focus on a multi-agent architecture that relies on remote communication media and agent cycles in a dynamic environment. Each agent in this architecture progresses from observations to reasoning and, ultimately, to actions, while taking into account the passage of time.

Keywords: Time, logic, robotic agents, cognitive robotics.

1. Introducción

La inteligencia humana está enmarcada dentro de una noción de tiempo. Sin embargo, para los humanos el tiempo es un concepto lineal, en tanto que para las inteligencias artificiales el tiempo tendría que ser considerado en su generalidad como

no-lineal [16, 14, 3]. Por ello, es indispensable considerar qué manejo de tiempo se asocia a una inteligencia artificial, sobre todo al permitir que inteligencias de este tipo tomen decisiones o lleven a cabo acciones. En este artículo se trata el tema de la deducción de estados temporales de agentes robóticos que realizan cierto conjunto de actividades dentro de un escenario físico.

Trataremos como caso de estudio a agentes robóticos móviles. Las deducciones así obtenidas son utilizadas para compartir información entre varios agentes y para tomar decisiones, ya sea grupales o individuales. Cada agente se interrelacionará con el mundo a través de un ciclo abductivo kowalskiano, en donde se realizan observaciones, se razona, y posteriormente se actúa, con posibilidades de actividades (mediante reglas de producción) emergentes o alternativas entre cada etapa [6].

También, aquí se muestra que las deducciones en relación a la espacio-temporalidad, un tema de conocimiento existencial fundamental, puede abordarse mediante un cálculo de eventos, así brindando algunos mecanismos computacionales extras a las reglas de producción para que, por un lado, los agentes 'conozcan' mejor su entorno y actúen en consecuencia, y por otro lado, su actuar sea enmarcado en una noción de tiempo apegada a la causalidad de la realidad y también sea asequible a la comprensión humana.

Panorama de este escrito. En la Sección 2 se tratará el tema de la programación lógica aplicada al cálculo de eventos, y de la descripción de este cálculo de eventos para conocer, con base en eventos y fluentes, los estados del mundo. En la Sección 3 se describen algunos aspectos tecnológicos que subyacen a los agentes robóticos a los que se puede aplicar nuestros resultados. En la Sección 4 se da un ejemplo amplio de cómo aplicar el cálculo de eventos descrito a agentes que, como supuesto, sigan un ciclo kowalskiano abductivo. Finalmente, en la Sección 5 se brindan algunas conclusiones y trabajo a futuro.

2. Lógica computacional

Aquí utilizamos una lógica computacional que considera la ocurrencia de eventos y la forma en que el mundo cambia, en su modalidad de programación lógica. Tal lógica es aplicada a agentes robóticos para obtener deducciones acerca del espacio y el tiempo. En la programación lógica se mantiene un equilibrio delicado entre la eficiencia en la ejecución de sus programas, mediante resolución SLDNF, y las lecturas declarativas de éstos, mediante modelos de Herbrand.

La programación lógica tiene como notable ventaja el proveer modelos sintácticos de relativa naturaleza simple, logrando así que los programadores mantengan certeza de cuál es el objetivo general a lograr cuando se implementa un sistema.

En este punto, enfatizamos que no solo es necesario considerar que los modelos propuestos sean asequibles, dados los supuestos axiomáticos formales, y aquellos supuestos obtenidos de la realidad"(domain theory), sino que también es fundamental mantener a la vista un enfoque computacional, que además sea de una eficiencia en tiempo real, un tema, éste último, de naturaleza fundamental en el caso de la posible aplicación de la lógica computacional a agentes robóticos.

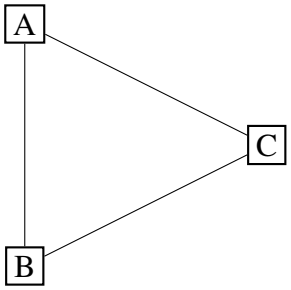


Fig. 1. Triángulo de Shanahan.

A continuación vemos un tema que tiene como origen el punto de vista de la programación lógica y se ha abierto paso en diversas aplicaciones que requieren el tratamiento del tiempo, a saber, el cálculo de eventos.

Cálculo de eventos. El cálculo de eventos basado en programación lógica fue ideado por Kowalski y Sergot hace ya algún tiempo [9], para incluir en el formalismo de la programación lógica la noción de predicados que cambian de valores de verdad conforme transcurre el tiempo.

Su relevancia en el tratamiento de temporalidad (dinámica) se muestra en publicaciones que aplican éste tipo de cálculo (o variantes y adaptaciones) a problemas que van desde temas de vigilancia por videos y sensores [12] hasta temas de salud pública [1]. Una excelente introducción al tema la da [15], en donde se aplica el cálculo de eventos a los temas de la inteligencia artificial, el no-determinismo, y a la abducción lógica, entre otros.

A continuación explicaremos los elementos esenciales requeridos del cálculo de eventos en la aplicación que tenemos destinada. De acuerdo con Shanahan [15] y basándose en la Fig 1, en donde de cada dos vértices conocidos se intenta estudiar un tercer vértice desconocido, se tiene:

A: Qué ha sucedido.

B: Qué efectos tienen algunos eventos o acciones sobre algunos fluentes.

C: Qué es verdadero en un momento dado (qué fluentes tienen el valor de verdad **Verdadero** en ese momento).

También de acuerdo con Shanahan, en la Fig. 1, dados dos vértices conocidos del triángulo, el tercero es posible de analizar con modalidades inductivas (B, si se conocen A y C), deductivas (C, si se conocen A y B) o abductivas (A, si se conocen B y C). Aquí nos enfocaremos en resultados deductivos (A y B conocidos, C desconocido): Hay una narración y se conoce qué hacen las acciones y cuáles son los fluentes, de donde se tiene que deducir qué es verdadero en cierto momento.

Algunas restricciones de integridad (integrity constraints) obtenidas de la abstracción que se obtiene de la realidad forman parte de la información que debe tomarse en consideración al momento de que un agente interactúe con el mundo. Estas restricciones complementan posible información necesaria para justificar la inducción, deducción o abducción realizada, según sea.

Las personas interesadas aún más en el tema son remitidas a [15] dónde se explican algunas suposiciones extra que hacen formalmente rigurosa la aplicación del cálculo de eventos (leyes inerciales, de unicidad de nombramiento, suposición del "mundo cerrado", teoría de la igualdad, entre otras suposiciones rigurosas y fundamentales).

2.1. Una ontología para cálculo de eventos

Para tratar el tema de un agente existiendo en un mundo, es necesario considerar varios conceptos e interrelaciones de un cálculo de eventos, es decir, requerimos de una ontología de cálculo de eventos[1]. Esta ontología consiste de identificar, en la medida de lo posible y en pro de tener un modelo adecuado a la ocurrencia de eventos en el tiempo y a la forma en que estas ocurrencias impactan en la veracidad del mundo en un instante dado, los siguientes conjuntos:

- Un conjunto finito de eventos, que se suponen instantáneos al ocurrir, completamente identificables, que afectan los valores de verdad de predicados llamados fluentes, descritos en el siguiente ítem.
- Un conjunto finito de fluentes, que son predicados con valor cambiante dependiendo de cómo están relacionados con la ocurrencia de eventos, y tales que está bien determinado su valor de verdad al momento de inspeccionarse, y en efecto, cada fluente debe estar relacionado con al menos un evento, sea para que el fluente comience a ser verdadero por la ocurrencia del evento o que el fluente comience a ser falso por tal ocurrencia.
- Un conjunto de valores de tiempo (instantes o momentos), los cuales deben ser comparables, discretizables (hasta un grado necesario), y de tal forma que todo evento tenga un momento de ocurrencia, y que todo fluente tenga un valor definido de veracidad dado un momento en el que el fluente se inspeccione. Hablamos de intervalos si tenemos dos momentos t_1 y t_2, con $t_1 < t_2$ tal que el conjunto, intervalo entre t_1 y t_2, $t \mid t > t_1$ y $t < t_2$ sea no vacío.
- Un conjunto de restricciones de integridad, como aquellas suposiciones, leyes, o condiciones de la realidad que son explícitamente establecidas en formulaciones lógicas, y que los eventos junto con los fluentes deben cumplir en cada ocasión (estipulando que, de otra manera, se incurriría en inconsistencias o violaciones de causalidad, por ejemplo).

La axiomática siguiente es identificada como "elemental" para el caso del tratamiento de un cálculo de eventos:

1. $\mathbf{Initiates}(\alpha, \beta, \tau)$: El fluente β comienza a ser verdadero cuando acontece el evento α en el instante τ.
2. $\mathbf{Terminates}(\alpha, \beta, \tau)$: El fluente β cesa de ser verdadero cuando acontece el evento α en el instante τ.

[1] Nota: La palabra ontología tiene en inteligencia artificial otra acepción a la aquí utilizada, de términos relacionándose con otros términos; no es la aquí aplicada.

$$\mathbf{HoldsAt}(f,t) \leftarrow \mathbf{InitiallyP}(f) \wedge \neg\mathbf{Clipped}(0,f,t) \tag{1}$$

$$\mathbf{HoldsAt}(f,t_3) \leftarrow \mathbf{Happens}(a,t_1,t_2) \wedge \mathbf{Initiates}(a,f,t_1) \wedge t_2 < t_3 \wedge \neg\mathbf{Clipped}(t_1,f,t_3) \tag{2}$$

$$\mathbf{Clipped}(t_1,f,t4) \leftrightarrow \exists a,t_2,t_3\,[\mathbf{Happens}(a,t_2,t_3) \wedge t_1 < t_3 \wedge t_2 < t4 \wedge [\mathbf{Terminates}(a,f,t_2) \vee \mathbf{Releases}(a,f,t_2)]] \tag{3}$$

$$\neg\mathbf{HoldsAt}(f,t) \leftarrow \mathbf{InitiallyN}(f) \wedge \neg\mathbf{Declipped}(0,f,t) \tag{4}$$

$$\neg\mathbf{HoldsAt}(f,t_3) \leftarrow \mathbf{Happens}(a,t_1,t_2) \wedge \mathbf{Terminates}(a,f,t_1) \wedge t_2 < t_3 \wedge \neg\mathbf{Declipped}(t_1,f,t_3) \tag{5}$$

$$\mathbf{Declipped}(t_1,f,t4) \leftrightarrow \exists a,t_2,t_3\,[\mathbf{Happens}(a,t_2,t_3) \wedge t_1 < t_3 \wedge t_2 < t4 \wedge [\mathbf{Initiates}(a,f,t_2) \vee \mathbf{Releases}(a,f,t_2)]] \tag{6}$$

$$\mathbf{Happens}(a,t_1,t_2) \rightarrow t_1 \leq t_2 \tag{7}$$

Fig. 2. Cálculo de eventos.

3. $\mathbf{InitiallyP}(\beta)$: El fluente β es verdadero desde el instante inicial (que uno puede convenir que sea 0 o 1).
4. $\mathbf{Happens}(\alpha,\tau)$: El evento α ocurre en el instante τ.
5. $\mathbf{HoldsAt}(\beta,\tau)$: El fluente β es verdadero en el instante τ.
6. $\mathbf{Clipped}(\tau_1,\beta,\tau_2)$: El fluente β es terminado en el intervalo $(\tau_1,\tau_2]$.
7. $\mathbf{Releases}(\alpha,\beta,\tau_2)$: El fluente β no es ya más "inercial" después del evento α ocurrido en el instante τ: ya no es influido más en su valor de verdad por la ocurrencia del evento α.
8. $\mathbf{InitiallyN}(\beta)$: El fluente β no es verdadero desde el instante inicial (0 o 1, a convenir).
9. $\mathbf{Declipped}(\tau_1,\beta,\tau_2)$: El fluente β comienza a ser verdadero en algún instante del intervalo $(\tau_1,\tau_2]$.

Formalmente, colocamos esta axiomática en forma clausal en la Fig. 2. Se tratan a continuación cómo se manejarán algunos temas relacionados con el cálculo de eventos en este escrito, particularmente los de granularidad del tiempo, los intervalos y las narrativas.

2.2. Granularidad, intervalos y tipos de narrativas

El paso del tiempo es generalmente supuesto continuo; no obstante, para propósitos computacionales es mejor suponerlo discretizado, de donde surge el problema de qué tan fina o gruesa se necesita una subdivisión para modelar algunos fenómenos. Esto es conocido como el problema de la granularidad temporal [2]. La granularidad aquí propuesta hace que los eventos sean instantáneos, y que los fluentes sean verdaderos o falsos por intervalos (posiblemente semi-abiertos).

Aún más, también impondremos una granularidad espacial, para que la movilidad de los agentes robóticos sea únicamente dada por coordenadas bidimensionales enteras, pero en una región acotada. Pasamos ahora a considerar inervalos de tiempo. En el enfoque de los relojes lógicos ([10]) son los mismos eventos los "motores" del paso del tiempo; aquí, en su lugar, el tiempo es medido por un reloj global único.

Todos los intervalos serán supuestos sobre el tiempo discretizado, y serán en efecto abiertos por la izquierda y cerrados por la derecha. Además, y como un eco de las marcas de tiempo (timestamps) de Lamport en [10], así como de un procedimiento lógico para deshacerse de los cuantificadores existenciales mediante la técnica de skolemización, utilizaremos una etiquetación consistente con el paso del tiempo, cuando sea necesario, para relacionar a los eventos con el instante de tiempo de su ocurrencia.

Pasamos ahora a definir algunos tipos de narrativas. Una narración consiste en enfocarse cuándo acontecen algunos eventos según una cronología y dentro de un escenario físico. Una narrativa es la identificación temporal de un conjunto de ocurrencias de eventos junto con la descripción de qué se ha afectado al ocurrir tales eventos. Se identificará como lo afectado a un conjunto de fluentes. Para un predicado de este tipo, de aridad n, se agregará el parámetro de tiempo al final de todos los argumentos, así haciendo un fluente de aridad $n + 1$.

Ahora bien, los eventos pueden ser descritos globalmente, por un superagente omnisciente que "sabe"qué ocurre en todo momento y para todo agente dentro de un mundo. Esto conlleva una narrativa global; este tipo de narrativas es adecuada para análisis teóricos de sistemas de agentes, sobre todo en ambientes artificiales que tengan una cantidad pequeña de eventos y de los fluentes afectados por el paso del tiempo. Otras dos opciones son: o narrativas grupales o narrativas individuales.

En una narrativa grupal el enfoque se da sobre los eventos que sean testimoniados (o realizados) por un subconjunto de agentes durante momentos o intervalos de tiempo bien definidos. Notar que una narrativa grupal es una proyección de una narrativa global.

La ventaja de las narrativas grupales es que son una más precisa descripción de los momentos de ocurrencia de los eventos o bien un estrechamiento de los intervalos narrados globalmente. Aquí se destaca el tema de la consistencia (ausencia de contradicciones) de las narraciones: ningún fluente de la narrativa global debe ser contradicho por una narrativa grupal.

En una narrativa individual se toman a agentes específicos para describir qué eventos y fluentes ocurren y se modifican, respectivamente, pero enfocándose en tales agentes. Cuando las narrativas son dadas por separado, se busca, en efecto, hallar consistencia entre las narrativas globales, grupales e individuales, ya que ello fortalece la consistencia de la descripción total del mundo.

2.3. Ciclos abductivos de agentes: Ciclo de agente Kowalskiano

En un ciclo abductivo kowalskiano [6] se tienen que cumplir ciclos del tipo: observar-razonar-actuar, con algunas partes esporádicas basadas en reglas de producción. Planeamos tratar tal enfoque que incluya reglas de producción en un trabajo a futuro. Por el momento, la parte de observar se traduce en un agente robótico como la capacidad de recabar información sensorial.

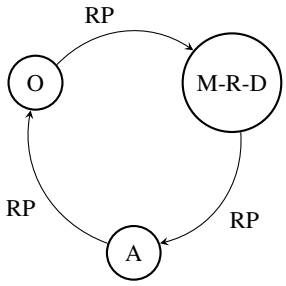


Fig. 3. Ciclo de agente kowalskiano: O: observaciones, M: metas, R: Razonamiento (diverso), D: Decisiones, A: Acciones, RP: Reglas de producción.

La parte de razonar es establecida, para el caso kowalskiano, con programas de la programación lógica. (El observar y el razonar se supondrán actividades realizables en tiempos acotados). No obstante, partes del actuar se ha considerado que es mejor manejarlo por reglas de producción (sobre todo acatando la exigencia de tiempo real para el funcionamiento de agentes), aunque las reglas de producción tienen el supuesto de que cada acción conlleva que el mundo cambie (actualización destructiva), en notoria contraposición a la referencia transparencial que la programación lógica ha propuesto a través de los años.

Lo que queda es tratar de realizar una entremezcla de estrategias computacionales: deducción en complemento a reglas de producción, o bien, la aplicación directa de reglas de producción preestablecidas cuando las deducciones auxiliares no sean posibles. Detallando, el ciclo abductivo kowalskiano, existen dos tipos de mecanismos computacionales posibles: de encadenamiento hacia adelante y hacia atrás.

El primer tipo de encadenamiento es típico de reglas de producción **if**-**then**, de la forma **if** ANTECEDENTE(S) **then** CONSECUENTE. Existen propuestas para manejar en lógica éste tipo de reglas per se [8]. El segundo tipo permite manejar cláusulas de Horn, de la forma CONSECUENTE **if** ANTECEDENTE(S) [5]. La necesidad de considerar reactividad y racionalidad de forma intermezclada [7], sin que una domine sobre la otra, nace de dotar a agentes (individual o grupalmente) de racionalidad en ambientes realistas [4].

En este escrito nos enfocamos en la parte de razonamiento temporal y clausal, ya que nuestro objetivo primordial es la realización de deducciones auxiliares, y trataremos de mostrar que éstas deducciones pueden contribuir a una mejora en la toma de decisiones para activar reglas de producción y la planeación a largo plazo, entre otros aspectos.

3. Agentes robóticos y diseño de comunicación

En agentes robóticos, el tema de reactividad es manejado independientemente de algunos temas relacionados al conocimiento existencial dentro de un marco de referencia temporal. Pero frecuentemente la creación de planes, por ejemplo, es un producto resultante de “reflexionar” históricamente, y no nada más del conocimiento inmediato del entorno.

Una de las propuestas dadas en este artículo es que la incorporación de información temporal coadyuva en la mejora conductual de los agentes robóticos en términos espaciales. Se tendría que aceptar que el almacenar información espacial a lo largo del tiempo afecta las futuras decisiones y acciones del agente.

Luego, es de esperar que un agente espacial se beneficie de una habilidad de integrar observaciones locales en instantes específicos para derivar interrelaciones espaciales en una mayor escala.

Con propósitos de experimentación, se ha ideado una arquitectura modular, con módulos basados en una computadora ejecutando Linux (Raspberry Pi)[2] y otros módulos basados en microcontroladores Esp32[3].

Dentro de este conjunto de módulos basados en microcontroladores, se tienen de 3 tipos: de recabado de datos sensoriales, de puesta en marcha de actuadores, y de gestores de comunicación Wi-Fi (por ejemplo, como puntos de acceso). Detallando:

1. Utilizamos una computadora Raspberry Pi junto con un robot PiCrawler[4] modificado (agregándole llantas, básicamente) para tener un nodo robótico (programado en Python, con interacción de sockets y TCP a algunos programas de Prolog), y hemos llamado a este robot Tribot.
2. Utilizamos un microcontrolador Esp32 (en modalidades Esp32 Lolin mini y Esp32 Wroom) para establecer un punto de acceso ligado al nodo robótico (programado en Arduino).
3. Utilizamos un Esp32 como receptor de datos (servidor) de parte del nodo robótico (programado en Python).
4. Utilizamos otro controlador Esp32 como emisor de datos de un ambiente a el nodo robótico (programado en Arduino).

Se han utilizado programas en varios lenguajes de programación para completar el funcionamiento del sistema (Prolog, Python, MicroPython y Arduino). Algunos sensores que hemos utilizado son cámaras, sensores de detección de luz y de distancia.

Los actuadores utilizados son algunos motores (llantas), algunos servos (que conforman brazos robóticos), e indicadores lumínicos (leds) (para indicar algunas actividades o reportar situaciones especiales del entorno). Los microcontroladores Esp32 tienen capacidades de comunicación Wi-Fi.

Utilizamos este tipo de comunicación entre un robot y varios microcontroladores, dispersos, Esp32 S2 Mini, mismos que tienen diversas posibilidades relacionadas con la recepción sensorial o, a su vez, capacidades reactivas.

En el ambiente Linux tipo Ubuntu de Raspberry, se han creado diversos servidores de acciones, tanto atómicas como compuestas que nuestro robot Tribot acepta. La forma de controlar estos agentes robóticos es por codificación directa, por sockets, y por comunicación Wi-Fi.

[2] www.raspberrypi.org

[3] www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf

[4] De la empresa Sunfounder.

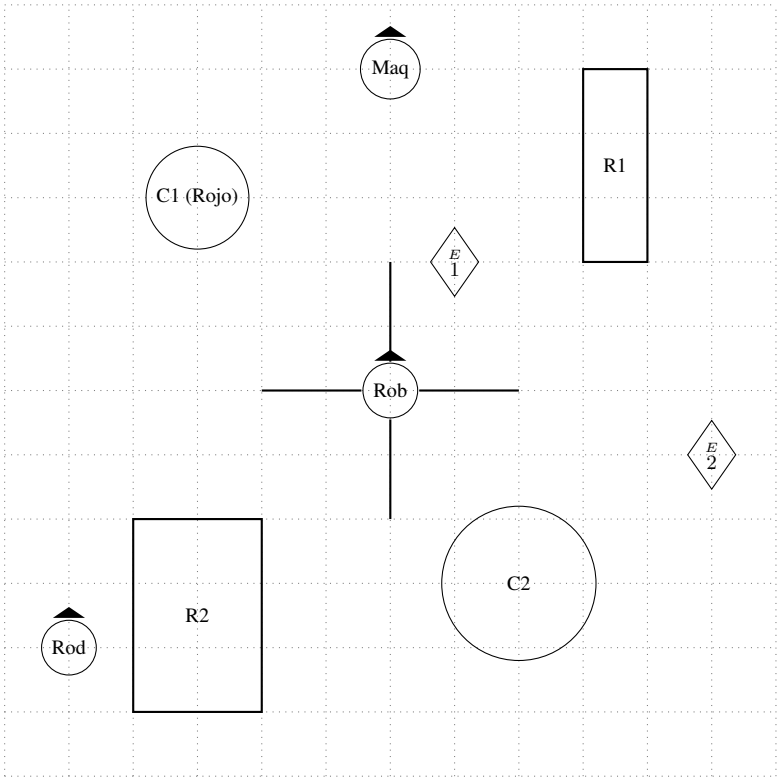


Fig. 4. Escenario inicial. Alcance de percepción.

4. Escenario robótico de estudio

Utilizaremos un conjunto de tres agentes robóticos (robots, para acortar). Nombraremos a los robots Rob, Maq y Rod. Los robots deambulan en un ambiente relativamente controlado pero semi-desconocido. En [17] se establece un problema parecido con métodos y objetivos diferentes, aunque con un punto de intersección esencial basado en la transferencia de información ocasional entre agentes.

Los robots tienen diversos sensores, algunos de los cuales son particulares o privados y otros más son irradiantes o públicos. Consideraremos que cada robot tiene un vector de n entradas, con algunas entradas de información dedicadas a percepción sensorial, otros a la emisión, y otros más de información diversa. Todos los robots comparten un punto de acceso.

No se supondrá una red de Internet ya sea por razones de ausencia de señal, o bien de privacidad o bien por la oportunidad de los microcontroladores Esp32 de configurarse como puntos de acceso por sí mismos, así siendo esto una decisión de diseño. Dada la falla de un punto de acceso, otro más se activará inmediatamente, dando lugar a un sistema tolerante a desconexiones y autorreparable en relación a las fallas de intercomunicación.

Como supuestos de movilidad (ver Fig. 4) tenemos lo siguiente: Cada robot puede moverse a lo largo de la cuadrícula señalada en esta figura, en la dirección hacia adelante (f), hacia atrás (b), a la derecha (r) o a la izquierda (l), en unidades enteras (siendo la actual cuadrícula una de esquina inferior izquierda ubicada en $(-6, -6)$ y en esquina superior derecha ubicada en $(6, 6)$). La marca de un triángulito oscuro sobre cada robot señala su respectiva dirección hacia adelante, en el momento captado en la figura.

En la Fig. 4 se muestra la posición de los robots Rod, Rob y Maq. Las figuras sombreadas son obstáculos sólidos, detectables por cierto tipo de sensores (ultrasónicos, por ejemplo). Además, los objetos marcados por una E son microcontroladores Esp32 con capacidades sensoriales y de emisión de información (pero no actuadores), configurables como puntos de acceso.

Con estos puntos de acceso es posible compartir información remotamente. Por un diseño en meta-nivel, estipulamos que el círculo pequeño C1 es rojo, el círculo grande C2 es azul, el rectángulo pequeño R1 es verde y el rectángulo grande R2 es amarillo. En la Fig. 4 también se describe la "percepción" de objetos. Aquí nos bastará una abstracción de un umbral de cercanía para estudiar los efectos de cambios internos (fluentes) de la información de los robots (su "conocimiento") a través del tiempo.

Mediante Wi-Fi es posible que cada robot tenga información de su propia ubicación. Bajo la instancia de interacción con un objecto (acción), el fluente asociado al "conocimiento" del robot cambia de "coordenada (x, y)" vacía a ocupada.

Bajo la instancia de encuentro casual entre dos robots, cada robot cambia algunos fluentes de "información no compartida" a "información compartida", de tal forma que los robots se comunican entre sí y se envían la información de qué lugares están ocupados por objetos. Describimos ahora algunas narrativas basadas en intervalos. Suponemos que los agentes robóticos están informados de lo siguiente [5]:

1. Hay tres robots en este mundo.
2. Hay 4 objetos.
3. Dos objetos son circulos, C1, C2, de diferente tamaño, C1 siendo el pequeño y C2 siendo el grande.
4. Dos objetos son rectángulos, R1, R2, también de diferente tamaño, esta vez siendo R1 el pequeño y R2 el grande.

Establecmos ahora una narrativa global, basada en intervalos. Durante el primer intervalo (de 0 a 10 unidades de tiempo, digamos), tenemos:

1. Rod parte de la parte inferior izquierda y camina en línea recta hacia arriba.
2. Rod encuentra un rectángulo, pero no sabe si es el grande o el pequeño, aunque detecta que es de color amarillo.
3. Rod sigue su camino y se encuentra con un círculo, que puede detectar que es el pequeño, y que además detecta que es rojo.
4. Rod permanece quieto durante el resto de este periodo.

[5] Notemos que es indispensable contar con bases de datos que manejen actualizaciones posiblemente destructivas. La recuperación de registros históricos longevos no está descartada si se requiere de, por ejemplo, del aprendizaje automático en alguna modalidad.

5. Rob permanece siempre quieto durante todo este intervalo.
6. Maq parte de su posicion hacia la parte superior derecha del escenario. Encuentra un rectángulo, que sabe que es el pequeño y que su color es verde.

Definamos el predicado **conoce/**4, **conoce**(AR,NObj,Color,tamaño(Tam)), el cual describe que el agente robótico, AR, conoce un objeto con nombre NObj, color Color, y tamaño Tam, siendo Tam grande, pequeño o desconocido. Al final de este intervalo se tiene el siguiente conjunto de instancias del fluente **conoce/**4:

{**conoce**(Rod,R2,amarillo,tamaño(`desconocido`)),
conoce(Rod,C1,rojo,tamaño(`pequeño`)),
conoce(Maq,R1,verde,tamaño(`pequeño`))}

Rob no ha hecho uso de su predicado **conoce**. Describamos lo que pasa durante el segundo intervalo (de 11 a 20 unidades de tiempo):

1. Esta vez Rod permanece quieto.
2. Rob, por su parte, camina del centro del escenario hacia la izquierda (desde nuestro punto de vista) y se halla tanto con el rectángulo R2, que sabe que es el grande y es amarillo, como con el círculo C1, que no sabe de su tamaño, pero sabe que es rojo.
3. Maq se desplaza hacia el centro de la parte izquierda del escenario, en donde es capaz de percibir la presencia del círculo C1, que detecta que es rojo pero ignora su tamaño.

Al final de este intervalo se tiene:

{**conoce**(Rob,R2,amarillo,tamaño(`grande`)),
conoce(Rob,C1,rojo,tamaño(`desconocido`)),
conoce(Maq,C1,rojo,tamaño(`desconocido`))}

Describamos ahora qué acontece durante el tercer intervalo (de 21 a 30 unidades de tiempo). Los robots se reunen en el punto medio de la izquierda del diagrama. (Otra variante es suponer que se utilizan sus capacidades Wi-Fi para comunicación remota). Se tiene, dado que las leyes inerciales se apliquen (los objetos permanezcan como objetos existentes, con sus colores y tamaños descubiertos en su momento):

{**conoce**(Rob,R2,amarillo,tamaño(grande)),
conoce(Rob,C1,rojo,tamaño(desconocido)),
conoce(Maq,C1,rojo,tamaño(desconocido)),
conoce(Maq,R1,verde,tamaño(pequeño)),
conoce(Rod,R2,amarillo,tamaño(desconocido)),
conoce(Rod,C1,rojo,tamaño(pequeño))}

Supongamos que cada robot comparte su conocimiento del mundo con el resto de robots:

1. Primera conclusión: Hay 4 objetos (geométricos) en este mundo: C1, C2, R1 y R2.
2. Segunda conclusión, de los colores:

a) C1 es rojo (descubierto por Rod).
b) C2 es azul (descubierto por Maq).
c) R1 es verde (descubierto por Maq).
d) R2 es amarillo (descubierto por Rod, confirmado por Rob).

3. Tercera conclusión, de los tamaños.
   a) C1 es el círculo pequeño (sabido por Rod).
   b) C2 es de tamaño desconocido, según Maq, pero conociendo la información de Rod, ahora se sabe que es de tamaño grande.
   c) R1 es de tamaño pequeño (según Maq).
   d) R2 es de tamaño grande (según Rob, pero apoyándose en lo dicho por Maq).

Notemos que en el caso de una experiencia propia es posible anotar que un objeto se ha llegado a conocer de forma directa. En este caso describimos un evento que sería el de "cercanía". Este evento dependería de algunos elementos que se involucren, tal como el agente robótico y el objeto:

Evento: **cercanía**(AR,Ob,conocer(Ob,Props*),t),

donde Props* indica una lista de atributos de un objeto, puestos como argumentos en el fluente. El fluente resultante con un argumento adicional de temporalidad sería el de **conoce/**5, con un argumento de cuándo el AR conoce tal objeto. Otro evento que destaca en la anterior descripción es el siguiente:

Evento: **traspasoInfo**(AR1,AR2,[ListaObjConocidos]).

En el caso de incorporar cómo se ha conocido el objeto, si por experiencia propia o ajena, definiríamos **conoceMedio/**6, que es como el fluente **conoce**, pero con un argumento extra para indicar si se llegó a conocer un objeto por experiencia propia o ajena:

Fluente: **conoceMedio**(AR,Obj,Props*,experienciaPropia,t).

Dado que un evento de traspasar información ocurra, para un agente receptor le es posible conocer la existencia de un objeto de una forma indirecta, a lo que nombramos, como ya mencionamos, experiencia ajena:

Fluente: **conoceMedio**(AR,Ob,Props*,experienciaAjena,t).

Notemos que hemos tomado el intervalo de 21 a 30 como un intervalo de actividad relacionado con compartir información. En un instante o intervalo posterior al momento 30 se puede utilizar la información sea de forma individual, grupal o global para decidir qué posibles actividades más podría realizar cada agente robótico.

En un formalismo que de énfasis al hecho de que los agentes se pueden organizar en agrupaciones, tal como [13], el tener criterios para realizar agrupaciones mediante la caracterización de los agentes (mediante algo conocido como atributos) permite que sea la base de trabajo realizable en equipo, por ejemplo. Dado el conocimiento geométrico grupalmente obtenido, es posible obtener planificaciones que optimicen el actuar espacial (planificando rutas óptimas) de los agentes robóticos [11].

5. Conclusiones y trabajo a futuro

En muchos sistemas multi-agente, las capacidades de los agentes se realzan si se toma en consideración el factor tiempo. En algunas descripciones de agentes, tales como [4]), las creencias, deseos e intenciones implícitamente tienen una noción de tiempo, pero para su manejo se requiere de algunas lógicas modales que parecen complicar los fundamentos teóricos.

En [6] se afirma que, desde la perspectiva del ciclo abductivo kowalskiano, las creencias son condiciones, los deseos son metas y las intenciones son planes, todo en un subsistema de la lógica de primer orden (aunque con preprocesamientos tipo cosificación (reification), en el que los fluentes pasan a ser argumentos de predicados).

En un trabajo a futuro se planteará cómo las reglas de producción pueden ser temporalmente complementadas. En este escrito se ha diseñado una arquitectura distribuida basada en comunicación Wi-Fi y controladores Esp32, con agentes robóticos capaces de deambular en un ambiente artificialmente diseñado, y distinguir colores y tamaños de algunos objetos colocados al azar.

También, se ha diseñado un recabamiento parcial de datos que se desarrolla en el tiempo para obtener consenso y conocimiento de una realidad global. Para ello se han utilizado diversas instancias de narrativas, cada una aportando un grado de generalidad apropiado al número de agentes involucrados, así como el intercambio oportuno de información.

Queda como trabajo a futuro el considerar la utilización de bases de datos locales y globales, con actualización destructiva y dando entrada a la formulación de razonamiento no-monótono (vía la adquisición de nueva información). También, es trabajo a futuro integrar estos resultados con un formalismo de sistemas multi-agentes específico; tenemos ya seleccionado para tal fin a SCEL [13].

En un escenario que posteriormente estudiaremos, se planea utilizar un conjunto de agentes robóticos para realizar trabajo cooperativo que requiera deliberación y planeación; esto se hará posible con base en historiales individuales y mediante la gestión de información remota entre los agentes.

Agradecimientos. Agradecemos al Instituto de Computación de la Universidad Tecnológica de la Mixteca las facilidades brindadas para llevar a cabo nuestra investigación acerca de agentes robóticos. También, agradecemos las valiosas observaciones para la mejora de este trabajo por parte de los árbitros de COMIA 2023.

Referencias

1. Chaudet, H.: Extending the event calculus for tracking epidemic spread. Artificial Intelligence in Medicine, vol. 38, no. 2, pp. 137–156 (2006) doi: 10.1016/j.artmed.2005.06.001
2. Fisher, M., Gabbay, D. M., Vila, L.: Handbook of temporal reasoning in artificial intelligence (2005)
3. Furia, C. A., Mandrioli, D., Morzenti, A., Rossi, M.: Modeling time in computing. ACM Computing Surveys, vol. 42, no. 2, pp. 1–59 (2010) doi: 10.1145/1667062.1667063

4. Georgeff, M., Pell, B., Pollack, M., Tambe, M., Wooldridge, M.: The belief-desire-intention model of agency. Intelligent Agents V: Agents Theories, Architectures, and Languages, pp. 1–10 (1999) doi: 10.1007/3-540-49057-4_1
5. Kowalski, R.: Logic for problem solving. North–Holland (1979)
6. Kowalski, R., Sadri, F.: Integrating logic programming and production systems in abductive logic programming agents. Web Reasoning and Rule Systems, pp. 1–23 (2009) doi: 10.1007/978-3-642-05082-4_1
7. Kowalski, R., Sadri, F.: Reactive computing as model generation. New Generation Computing, vol. 33, no. 1, pp. 33–67 (2015) doi: 10.1007/s00354-015-0103-z
8. Kowalski, R., Sadri, F.: Programming in logic without logic programming. Theory and Practice of Logic Programming, vol. 16, no. 3, pp. 269–295 (2016) doi: 10.1017/s1471068416000041
9. Kowalski, R., Sergot, M.: A logic-based calculus of events. New Generation Computing, vol. 4, no. 1, pp. 67–95 (1986) doi: 10.1007/bf03037383
10. Lamport, L.: Time, clocks, and the ordering of events in a distributed system. Communications of the ACM, vol. 21, no. 7, pp. 558–565 (1978) doi: 10.1145/359545.359563
11. Lien, J. M., Lu, Y.: Planning motion in similar environments with obstacles. In: Proceedings of Robotics: Science and Systems V (2009)
12. McAreavey, K., Bauters, K., Liu, W., Hong, J.: The event calculus in probabilistic logic programming with annotated disjunctions. In: Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems, pp. 105–113 (2017)
13. Nicola, R. D., Loreti, M., Pugliese, R., Tiezzi, F.: A formal approach to autonomic systems programming: The SCEL language. ACM Transactions on Autonomous and Adaptive Systems, vol. 9, no. 2, pp. 1–29 (2014) doi: 10.1145/2619998
14. Pfeifer, R., Bongard, J. C.: How the body shapes the way we think: A new view of intelligence. MIT Press (2007)
15. Shanahan, M.: The event calculus explained. Artificial Intelligence Today: Recent Trends and Developments, pp. 409–430 (1999) doi: 10.1007/3-540-48317-9_17
16. Varela, F. J., Thompson, E. T., Rosch, E.: The embodied mind: Cognitive science and human experience. MIT Press (1992)
17. Wijmans, E., Savva, M., Essa, I., Lee, S., Morcos, A. S., Batra, D.: Emergence of maps in the memories of blind navigation agents (2023) doi: 10.48550/ARXIV.2301.13261

ISSN 1870-4069

Reconocimiento de movimientos finos de la mano basado en señales de electromiografía usando algoritmos de aprendizaje automático supervisados

José Miguel Figarola, Yelile Iga Valdés,
Victor González, Javier Mauricio Antelis Ortíz,
Omar Mendoza Montoya

Tecnológico de Monterrey,
Facultad de Ciencias Escuela de Ingeniería y Ciencias,
México

{A01632557,A01634978,A01382698,
mauricio.antelis,omendoza83}@tec.mx

Resumen. El artículo describe la implementación de un sistema de clasificación de movimientos finos de la mano utilizando señales electromiográficas superficiales (sEMG). El objetivo es controlar una órtesis robótica para la rehabilitación de pacientes con enfermedades cardiovasculares e infartos cerebrales. Para lograr esto, se utilizó un enfoque de Aprendizaje Automático e Incremental y se llevó a cabo un experimento de laboratorio con 30 sujetos sanos. Se compararon diferentes clasificadores y se encontró que el Análisis Discriminante Lineal (LDA) y la Máquina de Soporte Vectorial Lineal (SVML) tuvieron un rendimiento sobresaliente. Además, se utilizó la técnica de Inicio Rápido o Warm Start y Ajuste Parcial o Partial Fit para mejorar la exactitud del modelo, lo que permitió adaptarlo a datos de pacientes sin entrenamiento previo y disminuir el tiempo de entrenamiento y procesamiento. Los resultados del estudio mostraron que LDA tuvo un rendimiento promedio del 94.8 %$\pm$3.21 para modelos individuales y SMVL con 93.4 %$\pm$3.69 para un modelo generalizado, lo que indica un rendimiento sobresaliente entre los clasificadores. Además, al utilizar técnicas de Inicio Rápido y Ajuste Parcial, la exactitud en la fase de pruebas aumentó hasta un 8 % en el modelo MLP3, pasando de un 58.09 % a un 61.42 %. En general, el estudio demostró la viabilidad de implementar Aprendizaje Automático y Aprendizaje Incremental en la detección y clasificación de movimientos finos de la mano utilizando señales sEMG para una Interfaz Cerebro-Computador (ICC) para la rehabilitación de pacientes con enfermedades cardiovasculares e infartos cerebrales.

Palabras clave: Aprendizaje incremental, inicio rápido, ajuste parcial, EMG, movimientos finos de la mano.

Recognition of Fine Hand Movements based on Electromyography Signals Using Supervised Machine Learning Algorithms

Abstract. The article describes the implementation of a fine hand movement classification system using surface electromyographic (sEMG) signals. The goal is to control a robotic orthosis for the rehabilitation of patients with cardiovascular diseases and cerebral infarctions. To achieve this, a Machine Learning and Incremental Learning approach was used, and a laboratory experiment was carried out with 30 healthy subjects. Different classifiers were compared, and it was found that Linear Discriminant Analysis (LDA) and Linear Support Vector Machine (SVML) had outstanding performance. In addition, the Quick Start or Warm Start and Partial Fit techniques were used to improve the accuracy of the model, which allowed it to be adapted to patient data without previous training and reduced training and processing time. The study results showed that LDA had an average performance of 94.8%± 3.21 for individual models and SMVL with 93.4%± 3.69 for a generalized model, indicating outstanding performance among classifiers. Furthermore, by using Quick Start and Partial Fit techniques, the accuracy in the testing phase increased by up to 8% in the MLP3 model, from 58.09% to 61.42%. Overall, the study demonstrated the feasibility of implementing Machine Learning and Incremental Learning in the detection and classification of fine hand movements using sEMG signals for a Brain-Computer Interface (BCI) for the rehabilitation of patients with cardiovascular diseases and cerebral infarctions.

Keywords: Incremental learning, warm start, partial fit, EMG, fine hand movements.

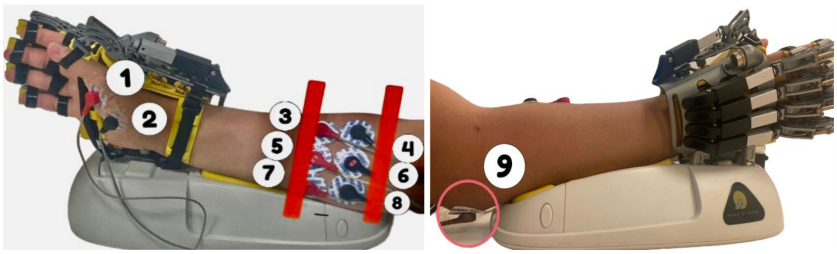
1. Introducción

Los problemas cardiovasculares e infartos cerebrales son muy comunes, crean afectaciones neurológicas y otras complicaciones [6]. En Estados Unidos cada 40 segundos ocurre un caso, un total de 805,000 al año y 1 de cada 5 son silenciosos por lo que pasan desapercibidos [5]. Yu et al. definen un ataque cerebrovascular como una alteración en el suministro de sangre al cerebro y esto afecta el rendimiento motor [17].

Esto puede generar limitaciones en rutinas diarias; National Washington Post y Journal of Stroke and Cerebrovascular Diseases estimaron un costo promedio de $20,396 a $43,652 USD por día en el hospital por un accidente cerebrovascular y un total del $17.5 billones de USD en Estados Unidos [16]. El uso de biomécanica y robótica ha generado grandes resultados como en la rehabilitación de extremidades inferiores.

Esto puede prevenir la fatiga y el esfuerzo en los pacientes y el personal de atención [14]. Por lo que es necesario encontrar nuevas técnicas y tecnologías que ayuden a reducir los costos y mejorar los resultados en la rehabilitación de pacientes que han sufrido problemas cardiovasculares e infartos cerebrales.

(a)



(b)

Fig. 1. (a) Descripción visual de cada época. (b) Orientación de los electrodos.

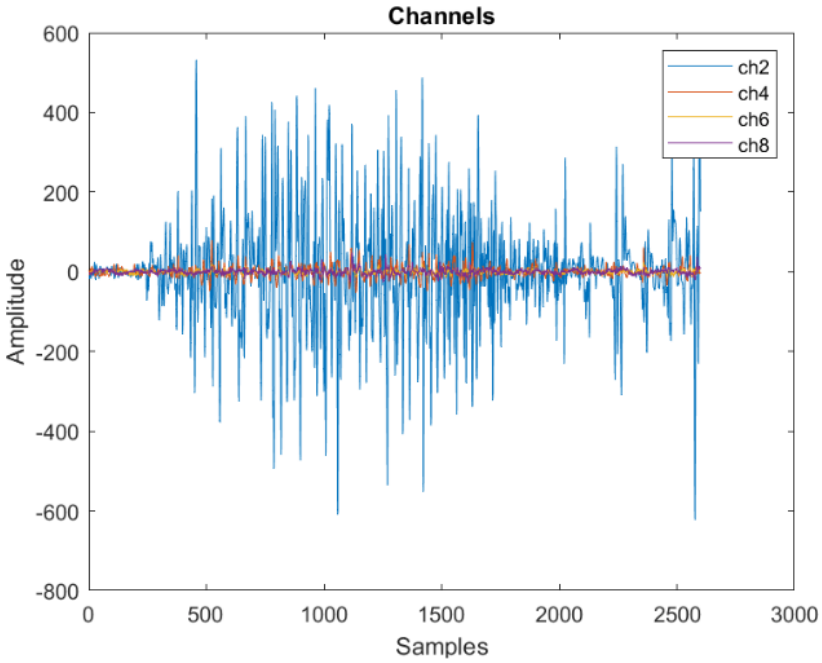
Además, es importante encontrar formas de mejorar la calidad de vida de estos pacientes y reducir el impacto de estas afecciones en su vida diaria. La biomécanica y la robótica pueden ser una solución efectiva para estos problemas y deben seguir siendo investigadas y desarrolladas para su uso en la práctica clínica.

1.1. Trabajos relacionados

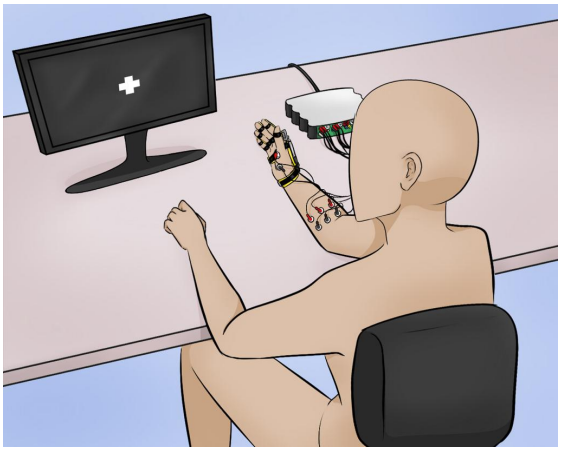
La investigación de Hazarika et al. combinó Análisis de Correlación Canónica (CCA) y K-Vecinos Más Próximos (KNN) para crear un método de clasificación de movimientos finos de la mano utilizando señales EMG [10], que podría ser útil en la rehabilitación de pacientes con enfermedades cardiovasculares y accidentes cerebrovasculares. El Aprendizaje Incremental se define como la capacidad de ajustar el modelo de acuerdo con nuevos ejemplos, sin la necesidad de un conjunto de entrenamiento adecuado antes del proceso de aprendizaje [7].

Redes Neuronales, el Perceptrón Multicapa y el Bosque Aleatorio son modelos de aprendizaje que pueden tener un gran impacto en la precisión del entrenamiento y la prueba cuando se aplica el Aprendizaje Incremental [9]. También se ha propuesto el uso de órtesis impulsadas por señales EMG con un gran rendimiento para medir movimientos compuestos de la mano [2].

Durante los últimos años, se ha estudiado la detección y clasificación de movimientos de la mano o movimientos finos de la mano utilizando señales EMG. En un estudio realizado por Lee et al. se entrenó una Red Neuronal con 4 capas y 1,000 neuronas en cada capa, utilizando 18 características en el dominio del tiempo, obteniendo una precisión media del 95 %.



(a)



(b)

Fig. 2. (a) Movimiento del pulgar por canal. (b) Gráfico del experimento.

Además, su Máquina de Soporte Vectorial Radial tuvo una precisión del 87,4 %, y utilizaron 10 participantes para crear su conjunto de datos [12]. Otro estudio, realizado por Ahsan et al. clasificó 4 gestos de la mano con una Red Neuronal que utiliza retropropagación durante el entrenamiento para lograr un mejor rendimiento. Los datos también se generalizaron para evitar el sobreajuste y se creó un conjunto de datos de 10 sujetos [1].

Tabla 1. Resultados de un sujeto independiente.

CLF	Media	Std	Min	Max
LDA	**90.78 %**	**±3.22**	**86.32 %**	**94.86 %**
SVML	88.14 %	±3.70	82.92 %	**93.47 %**
SVMR	88.42 %	±3.84	82.43 %	92.99 %
MLP1	88.04 %	±3.94	82.43 %	93.19 %
MLP2	87.78 %	±3.84	82.43 %	92.78 %
MLP3	88.26 %	±3.56	83.06 %	92.57 %
NNET1	88.32 %	±3.71	83.13 %	93.26 %
NNET2	87.99 %	±3.53	82.64 %	92.5 %
NNET3	**88.99 %**	**±3.26**	**84.38 %**	93.2 %

A diferencia de la mayoría de los estudios de EMG que se centran en discapacidades, Junior et al. propusieron diferentes modelos de Aprendizaje Automático y selección de características de Análisis de Componentes Principales (PCA) para clasificar 4 gestos de la mano de un sujeto diagnosticado con parálisis cerebral y un sujeto sano, obteniendo una precisión del 89,55 % con el sujeto 1 y del 93,13 % con el sujeto 2 [15].

Por último, Muhammad et al. intentaron obtener una comparación entre diferentes clasificadores, cuyo rendimiento no fue tan alto como en los estudios mencionados anteriormente, pero que sigue una de las ideas principales de este artículo [13].

Este artículo no se limita únicamente en crear modelos de clasificación, como Redes Neuronales o Máquinas de Soporte Vectorial, sino que evalúa y compara su rendimiento. El objetivo principal del estudio es mejorar los modelos de clasificación utilizando técnicas de Aprendizaje Incremental como Ajuste Parcial e Inicio Rápido.

Esto implica utilizar un clasificador previamente entrenado y ajustado para adaptarse a nuevas entradas, lo que puede reducir significativamente el tiempo necesario para entrenar y ajustar los parámetros del modelo.

2. Métodos y materiales

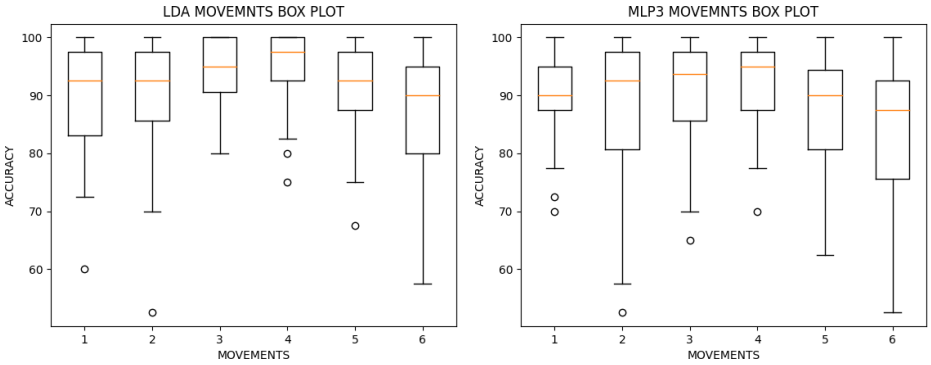
2.1. Descripción del experimento

El experimento consistió en que el participante utilizara una órtesis de mano robótica y se colocaran electrodos de EMG en su antebrazo. Se realizaron 40 épocas por clase y cada una contenía 3 imágenes diferentes para que el participante respondiera a la instrucción dada en un tiempo limitado de 2 segundos por imagen. El experimento duró 24 minutos y se realizaron 240 pruebas en total.

– Relax: Se le pidió al participante que relajara el brazo para evitar ruido en la señal.

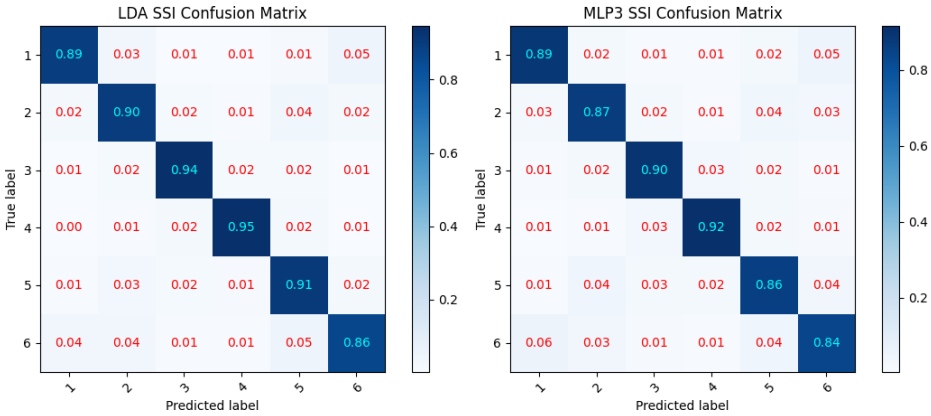
– +: La cruz de fijación exhorta al participante a prestar atención y concentrarse en la siguiente instrucción.

– Movement: Despliega que dedo flexionar o cerrar la mano, solo una vez.



(a) (b)

Fig. 3. (a) Diagrama de Caja de los Movimientos de LDA en USI. (b) Diagrama de Caja de los Movimientos de MLP3 en USI.



(a) (b)

Fig. 4. (a) Matriz de Confusión de LDA en USI. (b) Matriz de Confusión de MLP3 en USI.

En el estudio mencionado, se utilizó un protocolo aprobado por el comité de ética institucional de la Universidad Tecnológico de Monterrey. Los datos de EMG se registraron durante 3 minutos mientras los sujetos realizaban contracciones isométricas en el músculo flexor digitorum superficialis y profundus, así como en el músculo opponens pollicis, en diferentes niveles de esfuerzo.

Los datos se analizaron con el software MATLAB y en Python, y se realizaron pruebas estadísticas para evaluar las diferencias en la actividad muscular en función de la intensidad del esfuerzo. La figura 1 muestra la configuración de los electrodos utilizados en el experimento.

La adquisición de señales se realizó con un dispositivo de adquisición g.USBamp (g.tec, Austria) y los electrodos desechables Kendall (CardinalHealth, Canadá). Además, se aplicó un filtro Butterworth pasa-bandas de 8° orden con unas frecuencias de 1200Hz, $>$ 5Hz, $<$ 200Hz y un filtro Notch de Butterowrth de 4° orden a una

Tabla 2. Resultados de dejar un participante fuera.

CLF	Media	Std	Min	Max	Test
LDA	60.25 %	±**0.8**	59.26 %	61.52 %	55.25 %
SVML	65.32 %	±0.94	64.04 %	66.61 %	**55.54 %**
SVMR	73.74 %	±**0.8**	72.78 %	75.02 %	**56.93 %**
MLP1	80.8 %	±0.87	79.59 %	81.95 %	51.5 %
MLP2	**82.33 %**	±0.91	81.02 %	83.57 %	53.083 %
MLP3	**83.06 %**	±0.93	**81.69 %**	**84.3 %**	53.42 %
NNET1	81.77 %	±0.99	80.51 %	83.22 %	52.94 %
NNET2	82.35 %	±0.91	81.11 %	83.63 %	52.82 %
NNET3	82.89 %	±0.99	81.49 %	84.24 %	52.5 %

frecuencia de 1200Hz, $>$ 58Hz, $<$ 62 Hz para eliminar el ruido y una referencia bipolar se agregó para cada par de electrodos. Este estudio se basa en trabajos previos como el de Kyung-Jin You et al. debido a la configuración de electrodos utilizada en la adquisición de EMG [11].

2.2. Preprocesamiento

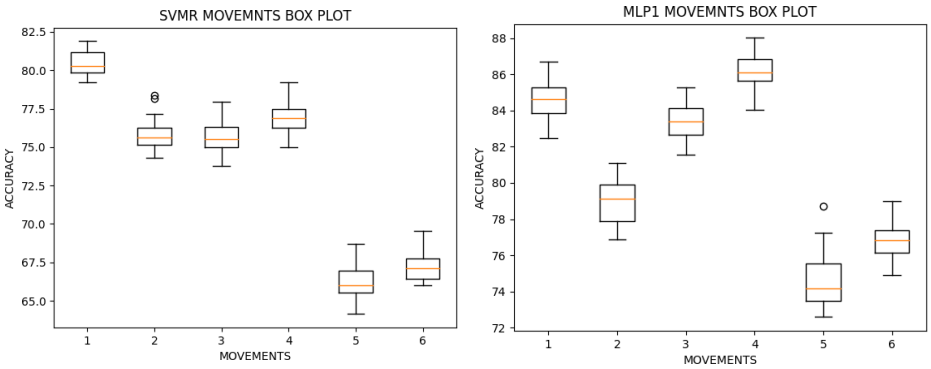
En esta etapa se obtuvieron épocas de 6 segundos a partir de la primera instrucción visual y cada una tenia una marca, esta época se redujo a una época de 2,2 segundos o 2600 muestras como en la figura 2 y se aplicó una corrección de línea base dado un preestímulo y estímulo para cada canal ID $\in \{1 : \text{Fixation}; 101 : \text{Thumb}; 102 : \text{Index}; 103 : \text{Middle}; 104 : \text{Ring}; 105 : \text{Little}; 101 : \text{Hand}; 200 : \text{Relaxation}\}$.

Por lo tanto, al final de la adquisición de datos y el preprocesamiento de datos, obtuvimos para cada participante una matriz de datos tridimensional S con 4 canales, 2600 muestras y 240 épocas, y un vector con una etiqueta para el movimiento de cada época que es representada por $y \in \{1 : \text{Thumb}; 2 : \text{Index}; 3 : \text{Middle}; 4 : \text{Ring}; 5 : \text{Little}; 6 : \text{Hand}\}$.

2.3. Extracción y selección de características

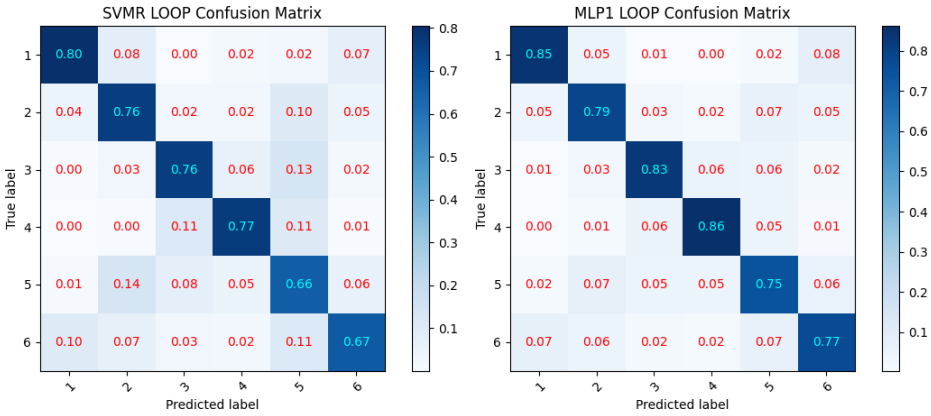
La extracción de características implica identificar características de una partición de datos [4]. Para entrenar un modelo, los datos crudos se analizaron con varias métricas numéricas, y se seleccionaron 10 características por canal en el dominio del tiempo, lo que resultó en un total de 40 características por prueba.

Mientras que Geethanjali et al. intentaron extraer al menos 7 características para una ICC [8], Arteaga et al. intentaron extraer 6 características de dominio de tiempo [3]. En este caso, seleccionamos 10 características, incluyendo Valor Cuadrático Medio, Varianza, Desviación Estándar, Valor Absoluto Máximo, EMG Integrada, Detector de Logaritmo, Cruce por Cero, Longitud de Onda, Integral Simple al Cuadrado, y Entropía de Wavelet.



(a) (b)

Fig. 5. (a) Diagrama de Caja de los Movimientos de SVMR en DUPF. (b) Diagrama de Caja de los Movimientos de MLP1 en DUPF.



(a) (b)

Fig. 6. (a) Matriz de Confusión de SVMR en DUPF. (b) Matriz de Confusión de MLP3 en DUPF.

Para el dominio del tiempo, se extrajeron 10 características por canal. El vector se representa como $\mathbf{x} \in \mathbb{R}^P$, con $P = 40$, indicando las 40 características totales que se plantean en la matriz X. Esta última tiene un tamaño de 240×40. Asimismo, la matriz de etiquetas, denominada y, tiene un tamaño de 240×1. A través del Análisis de Componentes Principales (PCA), se seleccionaron las características más significativas para el entrenamiento de los clasificadores.

2.4. Modelos de clasificación

Para entrenar modelos con múltiples datos de diferentes sujetos, se estandarizan las muestras utilizando la fórmula $z = (x - u)/s$, donde x representa las muestras de entrenamiento, s es la desviación estándar. Se seleccionan las 36 características más relevantes de la matriz de datos X, que se transforma mediante PCA.

Tabla 3. Resultados de Inicio Rápido.

CLF	0 %	10 %	20 %	30 %	40 %	50 %	60 %	70 %	80 %	90 %
MLP1	51.5	54.39	54.91	54.78	54.77	54.36	53.96	54.17	54.93	56.81
MLP2	53.08	55.91	56.42	57.28	57.55	57.19	57.08	56.71	57.29	61.39
MLP3	**53.42**	**56.40**	**57.48**	**58.09**	**57.87**	**58.5**	**57.99**	**58.38**	**58.82**	**61.53**

Se utilizan 4 modelos de clasificación diferentes para entrenar los datos con 5 K-Folds y Validación Cruzada, y se utilizan 36 características estandarizadas y seleccionadas por PCA. Los modelos son: Análisis Discriminante Lineal (LDA), Máquina de Vectores de Soporte (SVM), Perceptrón Multicapa (MLP) y Red Neuronal (NNET).

2.5. Descripción de los modelos de clasificación

Los modelos de clasificación nos permiten generar estructuras capaces de predecir etiquetas según los datos. Existen diversos tipos como bayesianos, de regresión, neuronales, entre otros. Debajo, se describen los modelos a utilizar y sus hiperparámetros.

– **LDA:** El Análisis Discriminante Lineal es un modelo de clasificación con un límite de decisión lineal ajustando las densidades condicionales a los datos usando la regla de Bayes.

– **SVM:** SVM encuentra un hiperplano en el espacio N-dimensional que clasifica los puntos de datos. Se implementaron dos SVM, ambas utilizan un coeficiente de regularización $C = 1$ y los kernels fueron lineal (SVML) y de función de base radial (SVMR).

– **MLP:**El Perceptrón Multicapa es una Red Neuronal Artificial (ANN) utilizada para resolver problemas de regresión o clasificación. Se uso una red de 4 capas; con 'tanh' como función de activación; 1,000 iteraciones máximas; 'adam' como optimizador.

1. MLP1 tiene 100,70,30,6 neuronas por capa respectivamente.
2. MLP2 tiene 100,100,100,6 neuronas por capa respectivamente.
3. MLP3 tiene 150,130,130.6 neuronas por capa respectivamente.

– **NNET:** Se creó una Red Neuronal secuencial de 5 capas con Keras y TensorFlow; se utilizó 'ReLU' como función de activación y para la última capa se seleccionó la función 'Softmax'; se entrenan durante 120 épocas y tienen un tamaño de lote de 30.

1. NNET1 tiene 36,100,70,30,6 neuronas por capa respectivamente.
2. NNET2 tiene 36,100,100,100,6 neuronas por capa respectivamente.
3. NNET3 tiene 36,150,130,130,6 neuronas por capa respectivamente.

Tabla 4. Resultados de Ajuste Parcial.

CLF	0 %	10 %	20 %	30 %	40 %	50 %	60 %	70 %	80 %	90 %
MLP1	51.5	56.27	56.99	56.69	56.62	56.14	55.56	56.02	56.39	61.11
MLP2	53.08	57.29	58.23	59.35	59.86	59.39	59.65	59.907	59.65	**65.97**
MLP3	**53.42**	**57.69**	**60.10**	**61.43**	**61.04**	**60.95**	**60.45**	**60.93**	**61.25**	64.72

3. Estudios de clasificación

Se realizaron 3 estudios considerando a los 30 participantes y los clasificadores se entrenaron respecto a la descripción de cada estudio para posteriormente evaluar su rendimiento.

3.1. Un sujeto independiente (USI)

Para este estudio, el entrenamiento de los modelos depende solo de un sujeto específico, es decir, se toma a uno de los 30 sujetos. Por lo tanto, las 240 pruebas del sujeto pasaron por la etapa de preprocesamiento y se aplicó la extracción y selección de características a los datos. El estudio entrenó los 9 clasificadores de forma independiente para obtener los resultados, el rendimiento y analizar el comportamiento de los clasificadores después del entrenamiento y se refleja en la tabla 1 en la sección 4.

3.2. Dejar un participante fuera (DUPF)

Para el estudio DUPF se toman en cuenta los 30 sujetos y se entrenan los clasificadores con 29 de los 30 sujetos, teniendo un total de 6,960 pruebas. Para este proceso se dejó fuera a cada sujeto y se entrenó con los otros 29, el participante restante se usó como datos de prueba con un total de 240 pruebas. Los resultados se muestran en la sección 4 en la tabla 2 para analizar el comportamiento de los clasificadores.

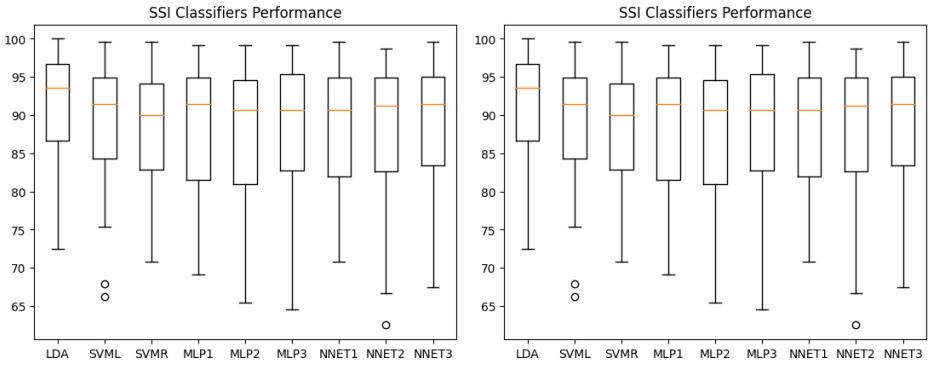
3.3. Aprendizaje incremental basado en inicio rápido y ajuste parcial

En esta sección, es crucial comprender la diferencia entre Inicio Rápido y Ajuste Parcial. En Inicio Rápido, los parámetros de los atributos ya aprendidos de los clasificadores no cambian, aunque algunos hiperparámetros pueden cambiar aqui se agregaron 30 y 50 neuronas a la segunda y tercer capa respectivamente.

Ajuste Parcial por otro lado, puede cambiar estos parámetros al aprender de los nuevos datos. El estudio siguió el mismo principio que el estudio DUPF pero los datos de prueba se dividieron en dos subconjuntos. Los resultados se registraron en la sección 4 como en los estudios 1 y 2.

3.4. Métricas de desempeño y pruebas estadísticas

Para analizar los resultados de los estudios descritos anteriormente, se utilizarán pruebas estadísticas como mecanismo para tomar decisiones cuantitativas basadas en el desempeño de los clasificadores, con el fin de detectar diferencias significativas con un nivel de significancia dado $\alpha < 0{,}05$.



(a) (b)

Fig. 7. (a) Rendimiento de los Clasificadores en USI. (b) Rendimiento de los Clasificadores en DUPF.

4. Resultados

4.1. Clasificación de un sujeto independiente (USI)

Para el estudio USI, cada clasificador se registró con los valores medios de su desempeño utilizando 5 K-Folds y Validación Cruzada, lo cual se muestra en la tabla 1. Como se muestra en la tabla, LDA tiene el mejor desempeño con una precisión media de validación del 90,78 %±3,22 y NNET3 con el 88,99 %±3,26. A pesar de su alto rendimiento, los demás clasificadores no se quedan atrás con su alta precisión, teniendo en cuenta que hay 6 clases.

En la tabla 1 se muestra el desempeño general de los clasificadores, así mismo se analizó por movimiento independiente y se encontró que LDA es capaz de diferenciar los movimientos 1, 5 y 6 sin dudarlo y MLP1 superó en los movimientos 2, 3 y 4. LDA tuvo una mejor precisión (98,33 %) al clasificar los movimientos que MLP3 (72,5 %). LDA es mucho mejor clasificando los movimientos 2, 3, 4 y 5.

MLP3 tiene dificultades con el movimiento 1 y ambos tienen dificultades con el movimiento 6, estos resultados se pueden ver en la figura 3. A pesar de que los clasificadores tienen dificultades para clasificar algunos movimientos, su precisión de validación es alta considerando que hay una probabilidad del 16 % de acertar al azar.

Ambos diagramas de caja se pueden representar en una matriz de confusión, la cual se muestra en la figura 4, que muestra un color azul más claro en el movimiento 6 (Movimiento en el cual se complica más la clasificación) y un color azul más oscuro en los movimientos 3 y 4, donde ambos clasificadores tuvieron un gran desempeño.

4.2. Clasificación de dejar un participante fuera (DUPF)

El estudio DUPF utilizó datos de 29 sujetos para crear un conjunto de datos de entrenamiento con 6,960 pruebas y un conjunto de datos de prueba con 240 pruebas. Cada sujeto se utilizó como datos de prueba una vez.

Tabla 5. Prueba de Tukey's HSD en USI.

	Group 1	Group 2	Diff	q-value	p-value
Comparación 0	LDA	SVML	2.64	1.71	**0.9**
Comparación 8	SVML	SVMR	0.278	0.18	**0.9**
Comparación 15	SVMR	MLP1	0.38	0.24	**0.9**
Comparación 21	MLP1	MLP2	0.264	0.17	**0.9**
Comparación 26	MLP2	MLP3	0.49	0.31	**0.9**
Comparación 30	MLP3	NNET1	0.056	0.036	**0.9**
Comparación 33	NNET1	NNET2	0.34	0.22	**0.9**
Comparación 35	NNET2	NNET3	1.0	0.64	**0.9**

Se utilizaron clasificadores propuestos y se encontró que tanto MLP2 como MLP3 tuvieron un gran rendimiento en la precisión media de validación, con 82,33 %$\pm$0.91 y 83,06 %$\pm$0.93 respectivamente.

Sin embargo, SVML y SVMR tuvieron mejor rendimiento en pruebas que ambos MLP, con un 55.54 % y 56.93 % respectivamente, esto se registró en la tabla 2. En el estudio DUPF, se observó que diferentes clasificadores funcionaron mejor para diferentes movimientos. NNET1 y NNET2 obtuvieron la mejor precisión media de todos los movimientos, con 83,91 % y 83,23 % respectivamente.

SVMR tuvo un rendimiento destacado en el movimiento 3, mientras que MLP1 y SVML funcionaron mejor en los movimientos 4 y 1, respectivamente. Sin embargo, ambos SVMR y MLP3 tuvieron dificultades en los movimientos 5 y 6, al igual que en el estudio USI. Se puede ver el rendimiento por clasificador y movimiento en el diagrama de caja de la figura 5.

La matriz de confusión del estudio DUPF muestra la comparación de los clasificadores y resalta la dificultad en los movimientos 5 y 6 (Figura 6). La dominancia de SVMR con los movimientos 1 a 4 se muestra en el diagrama de caja de la Figura 5.

Aunque la precisión de prueba fue inferior a la precisión de validación, todos los clasificadores superaron la probabilidad de acierto al azar del 16 %, siendo SVMR y SVML los mejores con una precisión de prueba del 56,93 % y 55,54 %, respectivamente, mientras que MLP1 tuvo la precisión más baja con el 51,5 %.

4.3. Aprendizaje incremental basado en inicio rápido y ajuste parcial

En las tablas 3 y 4, se observa que la precisión del clasificador MLP3 mejora a medida que se aumenta el porcentaje de datos utilizados para el entrenamiento. MLP3 tuvo un mejor rendimiento en ambas pruebas, con una mejora del 53,42 % al 61,53 % con Inicio Rápido y del 53,42 % al 64,72 % con Ajuste Parcial. Todos los MLP tuvieron un incremento en el número de neuronas en dos de sus cuatro capas.

Gracias a las tablas de resultados podemos ver una tendencia de convergencia después de utilizar el 40 % de los datos como entrenamiento. Por lo tanto, utilizando un porcentaje de nuevos datos, la precisión de prueba puede aumentar considerablemente.

Tabla 6. Prueba de Tukey's HSD en DUPF.

	Group 1	**Group 2**	**Diff**	**q-value**	**p-value**
Comparación 0	LDA	SVML	2.29	0.08	**0.9**
Comparación 8	SVML	SVMR	1.39	0.36	**0.9**
Comparación 15	SVMR	MLP1	5.43	1.41	**0.9**
Comparación 21	MLP1	MLP2	0.264	0.17	**0.9**
Comparación 26	MLP2	MLP3	1.58	0.41	**0.9**
Comparación 30	MLP3	NNET1	0.47	0.12	**0.9**
Comparación 33	NNET1	NNET2	0.12	0.032	**0.9**
Comparación 35	NNET2	NNET3	0.32	0.082	**0.9**

4.4. Métricas de rendimiento y análisis estadísticos

El test ANOVA se usa para comparar las medias de más de dos grupos y analizar sus varianzas para verificar si son iguales. En el estudio presentado en la figura 7, se aplicó el test ANOVA para evaluar el rendimiento de los clasificadores USI y DUPF.

Aunque no se encontró una diferencia significativa entre las medias de los grupos, se utilizó la prueba HSD de Tukey para comparar los clasificadores individualmente y se realizaron 35 comparaciones en total. Como se muestra en la figura 7 de arriba, los diagramas de caja sugieren que no hay diferencias entre los clasificadores porque la media de su rendimiento es casi igual. A pesar de que parecen ser iguales, la prueba ANOVA determinará si hay una diferencia significativa.

Prueba ANOVA y prueba Tukey (HSD) para USI: La prueba ANOVA de una vía no encontró diferencias significativas entre los clasificadores en el estudio de sujetos individuales ($p > 0{,}946 > \alpha$), lo que respalda la hipótesis nula $H_0 = \mu_0 = ... = \mu_p$. Tukey (HSD) permite comparaciones por grupo y también encontró que no hay diferencias significativas entre los clasificadores ($p = 0{,}9$), apoyando nuevamente la hipótesis nula y rechazando la hipótesis alternativa H_A.

Prueba ANOVA y prueba HSD de Tukey para DUPF: En los estudios USI y DUPF, se aplicó la prueba ANOVA y la prueba Tukey (HSD) para evaluar los clasificadores. En ambos estudios, no hubo una diferencia significativa entre los clasificadores, lo que se reflejó en los valores p que fueron mayores que α. Por lo tanto, se aceptó la hipótesis nula H_0 y se rechazó la hipótesis alternativa H_A. La prueba HSD de Tukey también mostró que no había una diferencia significativa entre los clasificadores comparados de forma independiente. En la tabla 6 se ven los resultados.

Se concluye que el rendimiento de los clasificadores no tiene diferencias significativas según las pruebas ANOVA y Tukey (HSD). Pero es relevante mencionar el estudio de Aprendizaje Incremental Basado en Ajuste de Inicio Rápido y Ajuste Parcial, donde el Perceptrón Multicapa superó a los otros clasificadores con una mejora cercana al 8 %.

5. Conclusiones

La clasificación de movimientos finos de la mano para la rehabilitación de pacientes con accidentes cerebrovasculares o neurodegenerativas es un trabajo de hace años y esto sigue evoluciuonando con los nuevos paradigamas de Inteligencia Artificial, los estudios muestran la efectividad de los modelos de Aprendizaje Automático en la clasificación de movimientos finos de la mano, con alta precisión y tiempo de entrenamiento reducido.

La metodología propuesta demuestra la adaptabilidad de los modelos y mantiene la generalización mientras aumenta la precisión. Los resultados obtenidos destacan el potencial de la robótica de rehabilitación para ayudar a pacientes con diversas condiciones médicas.

En el estudio de USI, LDA fue el mejor clasificador, logrando una precisión de validación del 90,77 %±3,2, mientras que en el estudio de DUPF, Perceptrón Multicapa y Redes Neuronales mostraron un rendimiento de validación más alto. Aunque las SVM obtuvieron la precisión de prueba más alta, necesitaron menos tiempo de entrenamiento que los MLP y NNET.

Por otro lado, el estudio de Aprendizaje Incremental utilizando Arranque de Inicio Rápido y Ajuste Parcial muestra la capacidad de los clasificadores para adaptarse a nuevos datos y rendir mejor con las entradas nunca vistas, aumentando hasta un 8 %.

Además, las pruebas estadísticas respaldan fuertemente los estudios y destacan el uso del Arranque de Inicio Rápido y el Ajuste Parcial como una herramienta para mejorar el rendimiento de los clasificadores y crear nuevos modelos en trabajos posteriores.

En resumen, estos estudios demuestran que los modelos de Aprendizaje Automático pueden ser una alternativa efectiva y de bajo costo para la rehabilitación robótica, especialmente en pacientes con problemas o riesgos cerebrales o cardiovasculares, y pueden ser adaptados para nuevos sujetos sin necesidad de una fase de entrenamiento extensa.

Referencias

1. Ahsan, M. R., Ibrahimy, M. I., Khalifa, O. O.: Electromygraphy (EMG) signal based hand gesture recognition using artificial neural network. In: 4th International Conference on Mechatronics, pp. 1–6 (2011) doi: 10.1109/icom.2011.5937135
2. Alshalali, T., Josyula, D.: Fine-tuning of pre-trained deep learning models with extreme learning machine. In: International Conference on Computational Science and Computational Intelligence, pp. 469–473 (2018) doi: 10.1109/csci46756.2018.00096
3. Arteaga, M. V., Castiblanco, J. C., Mondragon, I. F., Colorado, J. D., Alvarado-Rojas, C.: EMG-driven hand model based on the classification of individual finger movements. Biomedical Signal Processing and Control, vol. 58, pp. 101834 (2020) doi: 10.1016/j.bspc.2019.101834
4. Bhattachargee, C. K., Sikder, N., Hasan, M. T., Nahid, A. A.: Finger movement classification based on statistical and frequency features extracted from surface EMG signals. In: International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering, pp. 1–4 (2019) doi: 10.1109/ic4me247184.2019.9036671

5. Centers for disease control and prevention: Heart disease facts (2023) www.cdc.gov/heartdisease/facts.htm
6. Chen, Z., Venkat, P., Seyfried, D., Chopp, M., Yan, T., Chen, J.: Brain–heart interaction. Circulation Research, vol. 121, no. 4, pp. 451–468 (2017) doi: 10.1161/circresaha.117.311170
7. Espinoza, D. L., Eli Sanchez Velasco, L.: Comparison of EMG signal classification algorithms for the control of an upper limb prosthesis prototype. In: 17th International Conference on Electrical Engineering, Computing Science and Automatic Control, pp. 1–4 (2020) doi: 10.1109/cce50788.2020.9299208
8. Geethanjali, P., Mohan, Y. K., Sen, J.: Time domain feature extraction and classification of EEG data for brain computer interface. In: 9th International Conference on Fuzzy Systems and Knowledge Discovery, pp. 1136–1139 (2012) doi: 10.1109/fskd.2012.6234336
9. Geng, X., Smith-Miles, K.: Incremental learning. Encyclopedia of Biometrics, Springer US, pp. 912–917 (2015) doi: 10.1007/978-1-4899-7488-4_304
10. Hazarika, A., Dutta, L., Barthakur, M., Bhuyan, M.: Fusion of projected feature for classification of EMG patterns. In: International Conference on Accessibility to Digital World, pp. 69–74 (2016) doi: 10.1109/icadw.2016.7942515
11. KyungYou, K. J., Rhee, K. W., Shin, H. C.: Finger motion decoding using EMG signals corresponding various arm postures. Experimental Neurobiology, The Korean Society for Brain and Neural Science, vol. 19, no. 1, pp. 54–61 (2010) doi: 10.5607/en.2010.19.1.54
12. Lee, K. H., Min, J. Y., Byun, S.: Electromyogram-based classification of hand and finger gestures using artificial neural networks. Sensors, vol. 22, no. 1, pp. 225 (2021) doi: 10.3390/s22010225
13. Muhammad, F., Rashid, N., Akhtar, H., Muhammad, Z., Gilani, S. O., Ansari, U.: Evaluation of LDA, QDA and decision trees for multifunctional controlled below elbow prosthetic limb using EMG signals. In: International Conference on Robotics and Emerging Allied Technologies in Engineering, pp. 115–117 (2014) doi: 10.1109/icreate.2014.6828350
14. Physiopedia: Robotic rehabilitation for the lower extremity (2023) https://www.physio-pedia.com/Robotic_Rehabilitation_for_the_Lower_Extremity
15. Prado-Júnior, F. J., dos-Santos, F. V., Fernandes, C. A.: Classification of hand movements from EMG signals for people with motor disabilities. In: IEEE Latin America Transactions, vol. 18, no. 11, pp. 2019–2026 (2020) doi: 10.1109/tla.2020.9398644
16. Washington National: Why insurance: Covering the cost of stroke (2023) https://washingtonnational.com/explore/why-insurance/how-to-cover-stroke-cost/\#:~:text=According\%20to\%20the\%20\\Journal\%20of,ranges\%20from\%20\\\%2420\%2C396\%20to\%20\%2443\%2C652
17. Yu, Z., Prado, R., Quinlan, E. B., Cramer, S. C., Ombao, H.: Understanding the impact of stroke on brain motor function: A hierarchical bayesian approach. Journal of the American Statistical Association, vol. 111, no. 514, pp. 549–563 (2016) doi: 10.1080/01621459.2015.1133425

ISSN 1870-4069

Estimación de lluvias mensuales promedio con regresión lineal múltiple y redes neuronales artificiales en una cuenca semiárida

José Armando Rodríguez Carrillo, Julián González Trinidad, Gamaliel Moreno Chávez, Carlos Francisco Bautista Capetillo, Hugo Enrique Júnez Ferreira, Luis Fernando Castillo Martínez, Sandra Dávila Hernández

Universidad Autónoma de Zacatecas,
Unidad Académica de Ingeniería Eléctrica,
México

{jarmando.rc, jgonza, gamalielmch, baucap, hejunez, fercast, sandra.davila}@uaz.edu.mx

Resumen. En las últimas décadas, la estimación de lluvias es crucial debido a la importancia del recurso hídrico para el ser humano. Conocer su comportamiento y distribución es esencial para investigadores de la hidrología, permitiendo prevenir desastres como sequías e inundaciones, y aprovechar la lluvia en campos como agricultura, ganadería, industria, hidráulica, entre otros. Técnicas de Machine Learning han sido empleadas para la estimación de este fenómeno, por su capacidad de trabajar con grandes volúmenes de datos, presentando resultados más precisos que los métodos convencionales. El área de estudio es una cuenca semiárida en Valparaíso, Zacatecas, donde hay 4 estaciones pluviométricas con registros de 31 años (1990-2020). Los algoritmos empleados fueron Regresión Lineal Múltiple y Redes Neuronales Artificiales. Las métricas de evaluación fueron Error Absoluto Medio, Raíz del Error Cuadrático Medio y Coeficiente de Determinación. En la mayoría de los casos, los modelos de Redes Neuronales presentan mejores resultados que los de Regresión Lineal Múltiple, con valores de hasta 0.834, 0.255 y 0.378 en Coeficiente de Determinación, Error Absoluto Medio y Raíz del Error Cuadrático Medio respectivamente. Los modelos de Redes Neuronales son considerados buenos estimadores de lluvia en la cuenca.

Palabras clave: Estimación de lluvia, aprendizaje profundo, cuenca semiárida.

Estimation of Average Monthly Rainfall with Multiple Linear Regression and Artificial Neural Networks in a Semi-Arid Basin

Abstract. In recent decades, rainfall estimation has become a worldwide trend due to the importance of water resources for human beings. Knowing its behavior and distribution is one of the most important tasks for hydrology researchers, since this allows preventing natural disasters such as droughts and floods, as well as taking advantage of rainfall for different fields such as agriculture, livestock,

industry, hydraulics, among others. Machine Learning techniques have been used for the estimation of this hydrological phenomenon, because they allow working with a large amount of data, as well as presenting more accurate results than conventional hydrological methods. The study area is a semi-arid basin in the municipality of Valparaíso, Zacatecas, where there are 4 rainfall stations with 31-year records (1990-2020). The algorithms used for this work were Multiple Linear Regression and Artificial Neural Networks. The evaluation metrics were the Mean Absolute Error, the Root Mean Square Error and the Coefficient of Determination. In most cases, the Artificial Neural Networks models perform better than the Multiple Linear Regression models obtaining values up to 0.834, 0.255 and 0.378 in Coefficient of Determination, Mean Absolute Error and Root Mean Squared Error respectively. Artificial Neural Network models are considered good estimators of rainfall in the basin.

Keywords: Rainfall estimation, deep learning, semi-arid basin.

1. Introducción

La lluvia es la principal fuente de agua que tiene la sociedad para su desarrollo social y económico, teniendo una influencia directa en las actividades como la agricultura, sector de salud pública, industrial y la producción de energía eléctrica, se requiere seguir las investigaciones científicas y el manejo del agua para optimizar su aprovechamiento [8, 13]. Conocer el comportamiento espacio-temporal de la lluvia ayuda a crear planes y estrategias para la prevención de desastres naturales, como inundaciones y sequías [17].

En los últimos años, el cambio climático ha ocasionado que la cantidad de lluvia que llega a la superficie sea menor [8]. La medición de datos de lluvia se logra a través de una estación meteorológica, la cual registra la cantidad e intensidad de esta. Conocer la información registrada da ventajas para poder aplicar métodos que permitan determinar las relaciones y tendencia de la lluvia a lo largo del tiempo.

Generalmente, la predicción de lluvia se realiza a partir de modelos numéricos climatológicos en conjunto con datos de radares meteorológicos, basándose estos métodos en regresiones lineales, métodos numéricos, promedios y fórmulas empíricas, además de correlaciones entre diferentes variables meteorológicas y geográficas.

Sin embargo, tener modelos con exactitud y rapidez que predigan la cantidad e intensidad de la lluvia en una determinada área ayuda a entender mejor el fenómeno [13]. La Inteligencia Artificial se ha aplicado a problemas hidrológicos debido a que se pueden detectar patrones y tendencias de la lluvia con el análisis de los datos, además de que permite trabajar con una gran cantidad de información, a diferencia de los métodos convencionales [11].

Los algoritmos de Machine Learning (ML, por sus siglas en inglés), específicamente las Redes Neuronales Artificiales (ANN, por sus siglas en inglés) se han popularizado debido a que estas han demostrado tener una gran capacidad de modelar la no linealidad de los patrones [2]. En el 2014, [9] implementaron un algoritmo de ANN para modelar el comportamiento lluvia-escorrentía en el sur de Australia, además, implementaron correlaciones entre el flujo de la cuenca y la lluvia.

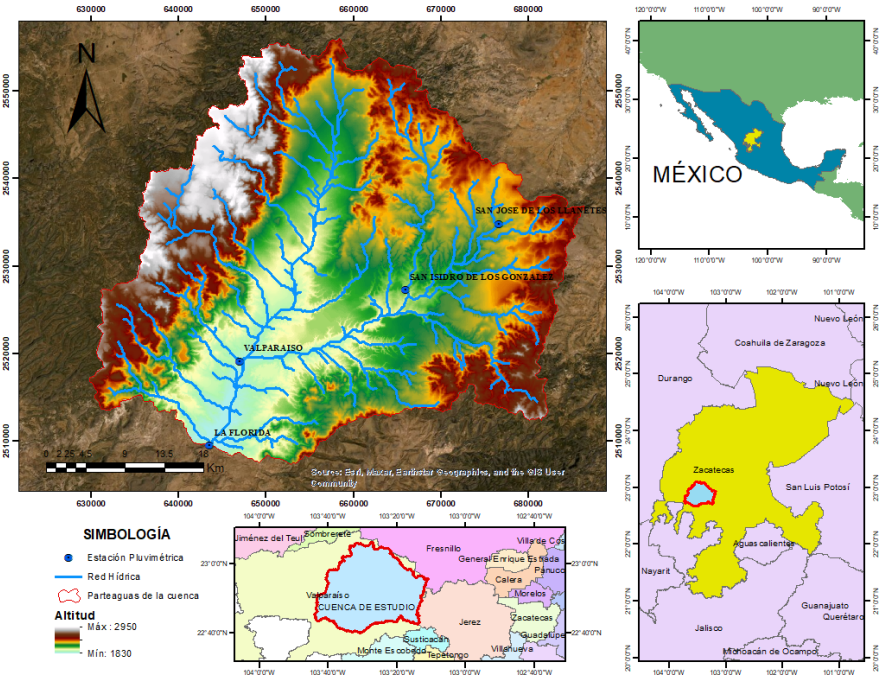


Fig. 1. Cuenca de Valparaíso, Zacatecas, México.

Por otro parte, [10] en 2018, propusieron un algoritmo de ANN para la predicción de lluvia, introduciendo como entradas del modelo índices climáticos, creando una red preliminar para identificar las relaciones de la lluvia con cada una de las variables para posteriormente, implementar un modelo con las características más significativas.

En 2019 [17], realizaron un modelo híbrido para la predicción de lluvia combinando dos modelos de pre-procesamiento de datos: un modelo ANN y Redes Neuronales Artificiales Estacionales evaluando el modelo en una ANN simple para posteriormente comparar los resultados con el Algoritmo Genético y el algoritmo de Reconocimiento Simulado con el modelo de Media Móvil Integrada Autorregresiva.

Igualmente, [13] en 2020 realizó una comparación de diferentes modelos de inteligencia artificial, los cuales fueron ANN, Máquina de Soporte Vectorial y Sistema de Inferencia Difusa basado en Red Adaptativa con Optimización de Enjambre de Partículas para la predicción diaria de lluvia en la provincia Hoa Binh, Vietnam, utilizando una base de datos de parámetros meteorológicos comprendidos entre la humedad relativa, velocidad del viento, temperatura máxima y temperatura mínima como datos de entrada y la lluvia diaria como salida.

De igual forma, [4] en 2020 implementaron algoritmos de aprendizaje automático para la predicción de lluvia diaria con término largo usando modelos lineales generalizados y algoritmos como Máquina de Soporte Vectorial, K- Vecino Más Próximo, Bosque Aleatorio, agrupamientos K-promedio y ANN, implementando patrones meteorológicos de bases de datos como entrenamiento del modelo.

Tabla 1. Coordenadas en proyección WGS84 UTM Zona 13N de las estaciones pluviométricas.

Estación Pluviométrica	Coordenada N	Coordenada Y	Coordenada Z
La Florida	2509447.13	643557.02	1830
Valparaíso	2519079.76	646947.2	1890
San Isidro de los González	2527237.95	665968.4	2034
San José de Llanetes	2534778.33	676596.17	2187

El objetivo principal de este estudio es estimar la lluvia mensual promedio aplicando los algoritmos de Regresión Lineal Múltiple (MLR, por sus siglas en inglés) y ANN a partir de datos de lluvia exclusivamente de 4 estaciones pluviométricas en la cuenca y un modelo general, que es una combinación de las 4 estaciones.

En la investigación se define el capítulo 2 con el área de estudio, mostrando algunas características geomorfológicas y sociales de la misma; en el capítulo 3 se muestran los materiales y métodos utilizados para el desarrollo de la investigación, desde la obtención de la base de datos, así como los modelos de ML implementados y sus métricas de evaluación; en el capítulo 4 se presentan los resultados obtenidos con los modelos de ML seleccionados y finalmente en el capítulo 5 las conclusiones.

2. Zona de estudio

La cuenca se localiza en el municipio de Valparaíso, Zacatecas, México (Figura 1), con clima semiárido con temperaturas que varía entre 12-24 °C, rango de lluvias de 500 a 1000 mm anuales principalmente en verano, con una población de aproximadamente 24,000 habitantes; dentro de la cuenca existen varios reservorios de agua, como la presa Manuel Pelgueres y presa El Salitre, entre otros.

La cuenca tiene un área de aproximadamente 1,771.19 km^2, con centroide en el eje X de 657526.08 y en el eje Y de 2529970.50 en la proyección WGS84 UTM Zona 13N (Tabla 1). Su geología comprende mayormente riolita-toba ácida (59.8 %), arenisca-conglomerado (31 %), basalto (3.4 %), aluvial (3 %), andesita (2 %), toba ácida (0.6 %), pórfido andesítico (0.3 %) y caliza-lutita (0.1 %).

El uso de suelo de la cuenca está compuesto principalmente por vegetación secundaria arbustiva pastizal natural (20.9 %), agricultura de temporal anual (19 %), bosque de pino-encino (14.8 %), vegetación secundaria arbustiva de bosque de encino-pino (8.8 %), vegetación secundaria arbustiva de bosque de encino (6.6 %), bosque de encino-pino (5.4 %), vegetación secundaria arbustiva de bosque de pino-encino y bosque de encino (5.3 %), pastizal inducido (4.1 %), agricultura de riego anual (2.6 %), matorral crasicaule (2.3 %), pastizal natural (1.9 %), vegetación secundaria arbustiva de selva baja caducifolia (1 %), vegetación secundaria arbórea de bosque de pino-encino (0.6 %), asentamientos humanos (0.5 %), agricultura de temporal permanente (0.4 %), vegetación secundaria arbustiva de bosque de pino (0.2 %), agricultura de temporal anual y permanente con los cuerpos de agua (0.1 %), y vegetación secundaria arbórea de bosque de encino-pino y bosque de pino (0.044 %).

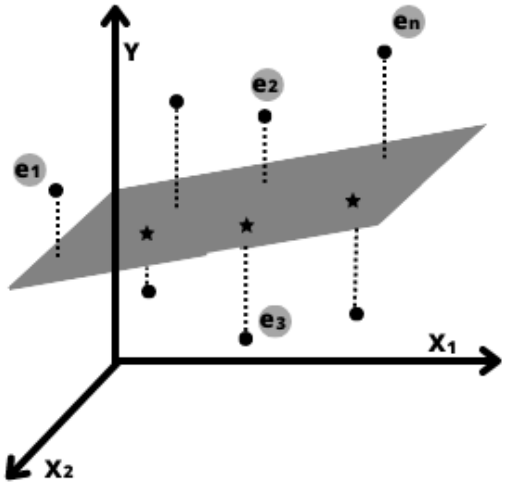


Fig. 2. Representación de MLR con dos variables independientes (x_1) y (x_2), donde se mapean las variables de entrada y se visualizan los residuos con respecto al modelo predicho [1].

La edafología está conformada por leptosol (26.02 %), lluvisol (20.80 %), phaeozem (18.96 %), cambisol (14.87 %), chernozem (9.52 %), kastañozem (5.12 %), Fluvisol (3.50 %), durizol (0.74 %), regosol (0.22 %), suelo ocupado por localidades (0.15 %) y cuerpos de agua (0.09 %).

3. Materiales y métodos

La metodología empleada en la investigación consistió en la obtención de los datos de lluvia registrada en los pluviómetros, reportados por la Comisión Nacional del Agua (CONAGUA), así como la aplicación de los métodos de ML para la estimación y sus métricas de evaluación. Se utilizó una validación cruzada de K-fold = 5, con un conjunto de entrenamiento de 70 % de los datos y el resto para prueba de acuerdo al criterio de muestreo aleatorio simple [19].

3.1. Obtención de los datos

Para delimitar el área de estudio se partió del Sistema Digital de Elevaciones del Instituto Nacional de Estadística y Geografía [7], posteriormente se descargó el índice de estaciones meteorológicas reportado por CONAGUA [3] para el software Google Earth Pro, esto con el fin de identificar las estaciones que se encontraban dentro de la cuenca (La Florida (LF), Valparaíso (VP), San Isidro de los González (SI) y San José de Llanetes (LL)), se extrajeron los datos de lluvia mensual promedio para el período 1990-2020, como se observó que faltaban datos, estos se calcularon con el método racional deductivo. Completados los datos se generó la base de datos con una combinación de entradas (3 estaciones) y una como salida.

Tabla 2. Resultados de los modelos de MLR.

Estación Pluviométrica	R^2	MAE	RMSE
Modelo General	0.757	0.295	0.499
La Florida	0.772	0.274	**0.443**
Valparaíso	0.759	0.293	0.485
San Isidro de los González	**0.790**	**0.268**	0.494
San José de Llanetes	0.767	0.274	0.455

3.2. Regresión lineal múltiple

El modelo MLR (ecuación 1) fue el primer algoritmo de ML y el más utilizado debido a su facilidad de aplicación, considera la relación lineal entre la variable dependiente e independiente [18]. La Figura 2 representa el mapeo de las variables y sus residuales con respecto al modelo MLR:

$$y = \beta_0 + \beta_1 \cdot x_1 + ... + \beta_n \cdot x_n + \epsilon, \tag{1}$$

donde y es la variable dependiente, x_i se refiere a la variable independiente, β_i es el parámetro y ϵ el error del dependiente de (x).

3.3. Redes neuronales artificiales

Una ANN es un modelo de Aprendizaje Profundo que está inspirado en la estructura del cerebro humano para la clasificación y reconocimiento de patrones, con el fin de realizar predicciones. Generalmente se utiliza una arquitectura de Perceptrón Multi-Capa para aumentar la exactitud del modelo [6].

Las neuronas son unidades computacionales las cuales contienen una función de activación que discriminan la información obtenida de otras neuronas y la selección de estas funciones dependen del problema y pueden cambiar en las diferentes capas de los modelos. Las neuronas están conectadas por pesos, estos representan el axón de las neuronas biológicas, que se estiman aleatoriamente mediante propagación hacia adelante y se actualizan con la propagación hacia atrás, utilizando un optimizador, el cual se encarga de buscar el mínimo de la función, con el fin de reducir el error que generó el modelo [5, 15] y esto se expresa matemáticamente en la ecuación 2, donde b es el bias del modelo, x_n las variables de entrada y w_n los pesos sinópticos de la red. En la figura 3 se muestra el diagrama de una ANN:

$$y = f(b + \sum_{i=1}^{n} x_n w_n). \tag{2}$$

3.4. Parámetros de ANN

Todos los modelos de ANN fueron entrenados con la función de pérdida de "Mean Squared Error", el optimizador aplicado fue "Adam", el cual tuvo los parámetros de tasa de aprendizaje de 0.001, un valor en β_1 de 0.9, β_2 de 0.999 y un valor de condición de paro epsilon de $1\,\mathrm{e}\text{-}07$.

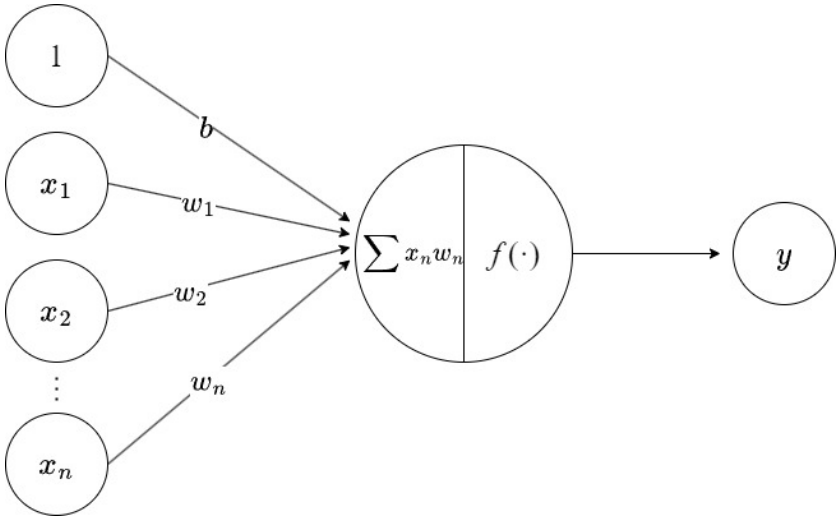


Fig. 3. Representación de una unidad de una ANN [16].

La arquitectura se baso en una capa de entrada con 12 neuronas usando la función de activación "ReLU", una capa oculta de 6 neuronas con la función de activación "ReLU" y una capa de salida con una neurona usando la función de activación "linear" con 150 épocas para cada modelo.

3.5. Métricas de evaluación

Las métricas utilizadas para la evaluación de los modelos son el Error Absoluto Medio (MAE, por sus siglas en inglés), en la ecuación 3, la Raíz del Error Cuadrático Medio (RMSE, por sus siglas en inglés), en la ecuación 4, finalmente utilizando el Coeficiente de Determinación (R^2), reportado en la ecuación 5 [12]:

$$\text{MAE} = \frac{\sum_{i=1}^{n} |E_i|}{n}, \tag{3}$$

$$\text{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} E_i^2}{n}}, \tag{4}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (P_i - A_i)}{\sum_{i=1}^{n} (A_i - \bar{A}_i)}, \tag{5}$$

donde A_i es el valor verdadero, P_i es el valor predicho, E_i es la diferencia entre el valor predicho y el valor verdadero, $\bar{A}_i$ es el promedio de los valores verdaderos y n se refiere al número total de valores.

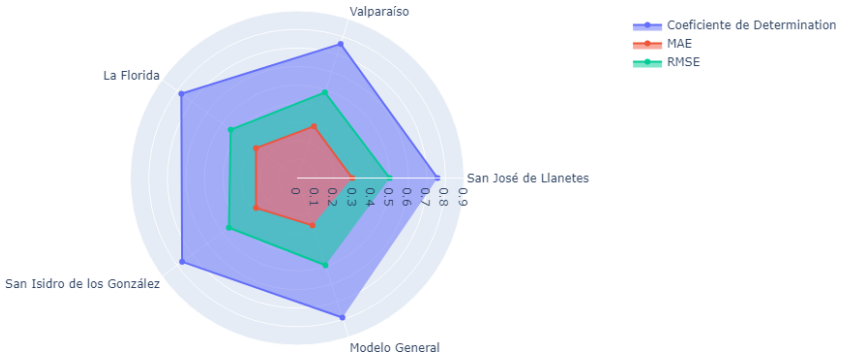


Fig. 4. Comportamiento de R^2, MAE y RMSE para los modelos de MLR.

4. Resultados

Se crearon 5 bases de datos diferentes, la primera fue con la estación de LF y las 3 estaciones restantes como entrada; la segunda fue implementada como salida la estación VP y las 3 restantes como entrada; la tercera tuvo como salida la estación SI y las 3 restantes como entrada; la cuarta con los datos de LL como salida y las tres restantes como entradas; el quinto modelo es la unión de las cuatro anteriores, de tal forma que los datos se cuadruplican, llamándolo Modelo General (MG). Para cada base de datos se creó un modelo MLR y un modelo ANN.

4.1. Resultados de MLR

Los resultados se pueden visualizar en la Tabla 2. Para el algoritmo MLR el modelo que mejor resultados dio fue en la estación SI con un R^2 de **0.790** y un MAE de **0.268**. El mejor puntaje en RMSE fue la estación de LF con **0.443**. El MG fue que mostro valores más bajos R^2 de 0.757, un MAE de 0.295 y un RMSE de 0.499. Las variaciones entre modelos fue del rango de 0.033 en R^2, 0.027 en MAE y 0.056 en RMSE Figura 4, donde se puede visualizar que existe una uniformidad entre los resultados, pues estos varían muy poco entre ellos, en cuanto a sus métricas de evaluación.

4.2. Resultados de ANN

Los resultados se muestran en la Tabla 3. El modelo que mejor resultados fue para la estación de LL con R^2 de **0.834**, **0.255** en MAE y **0.378** en RMSE. El modelo con los rendimientos más bajos en R^2 y MAE fue la estación de VP con 0.758 y 0.295 respectivamente mostrando la variación espacial y temporal de la lluvia.

El mayor error en RMSE se encontró fue la estación SI con 0.496, lo que indica la poca relación de los años analizados. Los modelos varían en un rango de 0.076 en R^2, 0.040 en MAE y 0.118 en RMSE Figura 5, donde se puede visualizar de ua mejor manera la relación que existe entre R^2 con MAE y RMSE, ya que mientras los últimos dos mencionados disminuyen, el primer R^2 aumenta.

Tabla 3. Resultados de los modelos ANN.

Estación Pluviométrica	R^2	**MAE**	**RMSE**
Modelo General	0.791	0.269	0.490
La Florida	0.810	0.266	0.404
Valparaíso	0.758	0.295	0.486
San Isidro de los González	0.788	0.257	0.496
San José de Llanetes	**0.834**	**0.255**	**0.378**

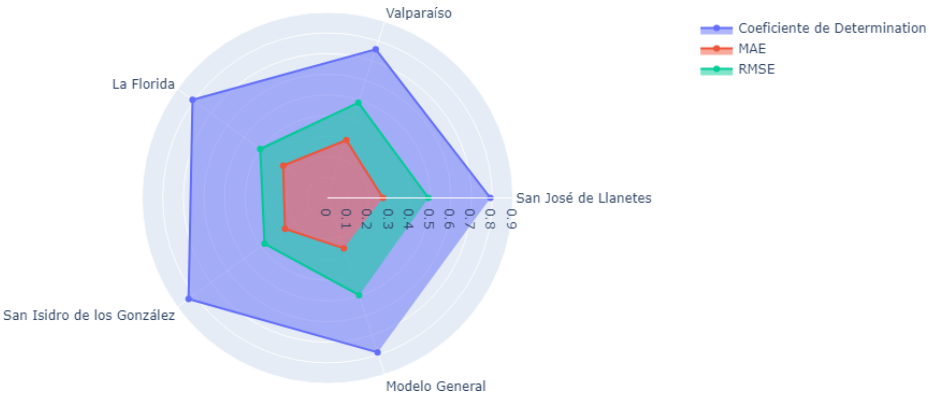


Fig. 5. Comportamiento de R^2, MAE y RMSE para los modelos de ANN.

Autores como [6, 14], han reportado valores similares a los encontrados en esta investigación, sobre todo en los valores de R^2. Sin embrago difieren de [11, 5], los cuales indican que se pueden mejorar algunas métricas de las utilizadas para regiones similares a esta, aunque puede influir que ellos utilizan datos diarios y otras características de entrada.

4.3. Comparación de los modelos de MLR y ANN

En la figura 6 se muestran las comparaciones de las métricas para ambos modelos. Para R^2 de los modelos MLR y ANN (a), donde se puede observar que para el MG, el modelo ANN superó al modelo MLR con 3.4 %; para la estación de LF, el modelo ANN también mejoró su rendimiento con respecto al modelo MLR con 3.8 %; caso contrario con la estación de VP y SI, donde los modelos ANN bajaron ligeramente su rendimiento con 1 % y 2 % respectivamente; finalmente la estación de LL, mostró el mejor desempeño para esta métrica, aumentando un 6.7 %.

Cabe destacar que para esta métrica, los modelos ANN mejoraron considerablemente en la estación LL, LF y en el MG. Los resultados de las diferencias de MAE de los modelos (b) demuestran que el resultado MG-ANN-MAE fue mejor que MG-MLR-MAE con 0.269 y 0.295 respectivamente; LF-ANN-MAE lanzó el resultado 0.266, siendo mejor que LF-MLR-MAE, pues este valor fue de 0.274; comparando VP-ANN-MAE y VP-MLR-MAE, el mejor fue el último modelo mencionado con 0.293, a diferencia de 0.295; se obtuvo un valor de 0.268 en SI-MLR-MAE, siendo

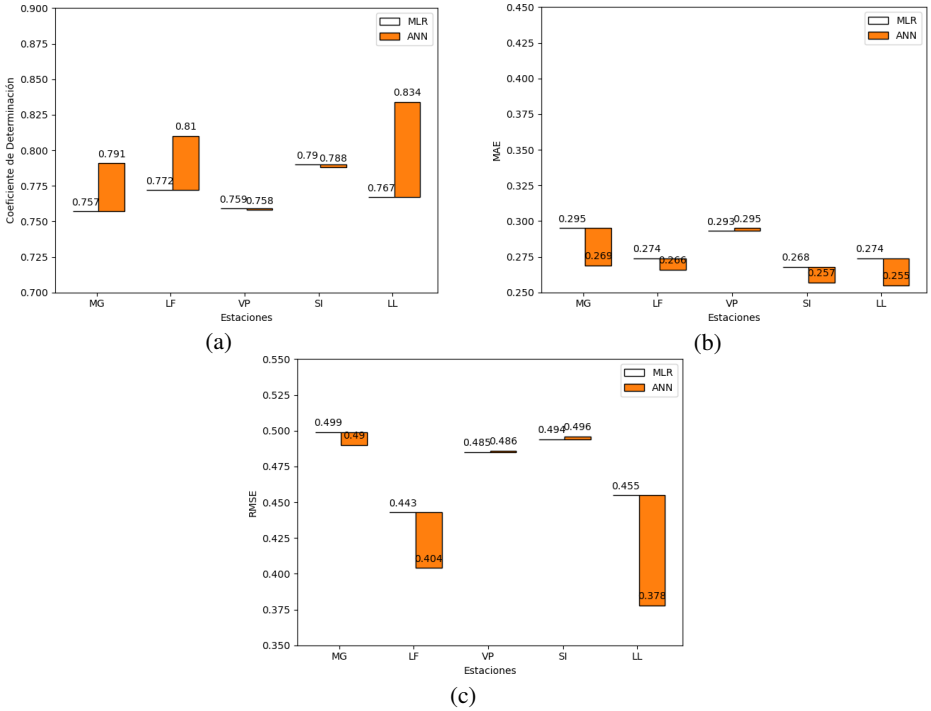


Fig. 6. Comparación de las métricas de los modelos MLR vs ANN.

mejorado por SI-ANN-MAE con 0.257; finalmente con LL-MLR-MAE el valor fue de 0.274, comparándolo con LL-ANN-MAE, que fue mejor con 0.255, además de que fue la mejor evaluación en MAE para todos los modelos implementados. En MAE los modelos ANN se demostraron muy similares e incluso en una estación siendo superado por los modelos MLR.

En la métrica MG-ANN-RMSE y MG-MLR-RMSE los resultados fueron 0.490 y 0.499, denotando una ligera mejora en el modelo ANN; para LF-MLR-RMSE el valor fue de 0.443, superado por LF-ANN-RMSE con 0.404; en VP-MLR-RMSE (0.485) y VP-ANN-RMSE (0.486) los valores fueron muy similares, con una diferencia mínima de 0.001; en SI-MLR-RMSE el valor fue de 0.494 y con menor rendimiento SI-ANN-RMSE fue de 0.496; en la estación LL, los resultados con LL-MLR-RMSE y LL-ANN-RMSE fueron 0.455 y 0.378, respectivamente, donde el modelo ANN para esta estación fue el mejor resultado en RMSE de todos los modelos. Para esta métrica, las estaciones LF y LL mejoraron considerablemente, en comparación con el resto de estaciones, esto se puede observar en (c).

Los datos obtenidos dan elementos para entender el comportamiento espacio-temporal de la lluvia en un área estudiada de 1,771 km^2, mostrando que los registros de lluvia en las estaciones son diferentes, además, influenciados por la altitud. Por otro lado, el modelo ANN, ajusta mejor los datos para estimar con menor incertidumbre la lluvia, logrando con ello obtener datos más confiables para el cálculo del volumen de agua disponible.

5. Conclusiones

En esta investigación se planteó como objetivo estimar la lluvia mensual promedio aplicando los algoritmos de MLR y ANN a partir de datos de lluvia registrados en 4 estaciones pluviométricas de una cuenca semiárida y un modelo general, que es una combinación de las 4 estaciones.

El modelo ANN mostró los mejores resultados con respecto a R^2 de 0.834 para la estación LL, con una mínima diferencia (0.049), entre las estaciones VP, LF y SI, esta pequeña diferencia se puede asumir que es debida a la variación espacio-temporal de la lluvia. El MAE y RMSE mostró un comportamiento similar a R^2 con respecto a las estaciones, con un rango de 0.255-0.295 y 0.378-0.496 respectivamente.

El modelo general de ANN mejora la estimación de lluvia en la cuenca, con respecto al MLR, por lo tanto se sugiere que la estimación de la lluvia podría realizarse con el método ANN con menor incertidumbre. Se recomienda que trabajos futuros se enfoquen a implementar esta metodología con más algoritmos de ML, con intervalos de tiempo más cortos (diarios y anuales).

Agradecimientos. Los autores agradecen al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico a través de la convocatoria "Becas Nacional (Tradicional) 2021-3", para el alumno José Armando Rodríguez Carrillo, facilitando el desarrollo de la presente investigación y a la Comisión Nacional del Agua (CONAGUA) por los datos proporcionados.

Referencias

1. Carrasquilla-Batista, A., Chacón-Rodríguez, A., Núñez-Montero, K., Gómez-Espinoza, O., Valverde, J., Guerrero-Barrantes, M.: Regresión lineal simple y múltiple: Aplicación en la predicción de variables naturales relacionadas con el crecimiento microalgal. Revista Tecnología en Marcha, vol. 29, pp. 33–45 (2016) doi: 10.18845/tm.v29i8.2983
2. Chiacchiera, A., Sai, F., Salvetti, A., Guariso, G.: Neural structures to predict river stages in heavily urbanized catchments. Water, vol. 14, no. 15, pp. 2330 (2022) doi: 10.3390/w14152330
3. CONAGUA: Comisión nacional del agua. (2023)
4. Diez-Sierra, J., Del Jesus, M.: Long-term rainfall prediction using atmospheric synoptic patterns in semi-arid climates with statistical and machine learning methods. Journal of Hydrology, vol. 586, pp. 124–789 (2020) doi: 10.1016/j.jhydrol.2020.124789
5. Endalie, D., Haile, G., Taye, W.: Deep learning model for daily rainfall prediction: Case study of Jimma, Ethiopia. Water Supply, vol. 22, no. 3, pp. 3448–3461 (2022) doi: 10.2166/ws.2021.391
6. Gu, J., Liu, S., Zhou, Z., Chalov, S. R., Zhuang, Q.: A stacking ensemble learning model for monthly rainfall prediction in the Taihu Basin, China. Water, vol. 14, no. 3, pp. 492 (2022) doi: 10.3390/w14030492
7. INEGI: Instituto nacional de estadística, geografía e informática (2023)
8. Kala, A., Vaidyanathan, S. G.: Prediction of rainfall using artificial neural network. In: International Conference on Inventive Research in Computing Applications, pp. 339–342 (2018)

9. Kamruzzaman, M., Shahriar, M. S., Beecham, S.: Assessment of short term rainfall and stream flows in South Australia. Water, vol. 6, no. 11, pp. 3528–3544 (2014) doi: 10.3390/w6113528
10. Lee, J., Kim, C. G., Lee, J. E., Kim, N. W., Kim, H.: Application of artificial neural networks to rainfall forecasting in the Geum River Basin, Korea. Water, vol. 10, no. 10, pp. 1448 (2018) doi: 10.3390/w10101448
11. Liyew, C. M., Melese, H. A.: Machine learning techniques to predict daily rainfall amount. Journal of Big Data, vol. 8, pp. 1–11 (2021) doi: 10.1186/s40537-021-00545-4
12. Naser, M. Z., Alavi, A.: Insights into performance fitness and error metrics for machine learning (2020) doi: 10.48550/arXiv.2006.00887
13. Pham, B. T., Le, L. M., Le, T. T., Bui, K. T. T., Le, V. M., Ly, H. B., Prakash, I.: Development of advanced artificial intelligence models for daily rainfall prediction. Atmospheric Research, vol. 237 (2020) doi: 10.1016/j.atmosres.2020.104845
14. Ridwan, W. M., Sapitang, M., Aziz, A., Kushiar, K. F., Ahmed, A. N., El-Shafie, A.: Rainfall forecasting model using machine learning methods: Case study Terengganu, Malaysia. Ain Shams Engineering Journal, vol. 12, no. 2, pp. 1651–1663 (2021) doi: 10.1016/j.asej.2020.09.011
15. Sarasa-Cabezuelo, A.: Prediction of rainfall in Australia using machine learning. Information, vol. 13, no. 4, pp. 163 (2022) doi: 10.3390/info13040163
16. Sharma, P.: Introduction to the neural network model, glossary, and backpropagation (2022) www.analyticsvidhya.com/blog/2022/01/introduction-to-the-neural-network-model-glossary-and-backpropagation/
17. Tran-Anh, D., Duc-Dang, T., Pham-Van, S.: Improved rainfall prediction using combined pre-processing methods and feed-forward neural networks. J — Multidisciplinary Scientific Journal, vol. 2, no. 1, pp. 65–83 (2019) doi: 10.3390/j2010006
18. Uyanık, G. K., Güler, N.: A study on multiple linear regression analysis. Procedia-Social and Behavioral Sciences, vol. 106, pp. 234–240 (2013) doi: 10.1016/j.sbspro.2013.12.027
19. Verma, S. P.: Estadística básica para el manejo de datos experimentales: Aplicación en la geoquímica (geoquimiometría). Universidad Nacional Autónoma de México (2005)

ISSN 1870-4069

Agrupamiento automático de datos magnéticos en prospección geofísica para arqueología

Manuel Ortiz Osio[1], Erik Molino Minero Re[2]

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

[2] Universidad Nacional Autónoma de México,
Unidad Académica Yucatán,
México

manuel.ortiz@ingenieria.unam.edu,
erik.molino@iimas.unam.mx

Resumen. Uno de los objetivos de la geofísica es la interpretación de imágenes asociadas con variables físicas de estructuras presentes en el subsuelo. En este trabajo proponemos aplicar a los datos geofísicos una metodología de agrupamiento usando herramientas de aprendizaje computacional, con el fin de brindar una herramienta cuantitativa para la interpretación. Debido a que no siempre es posible conocer la distribución verdadera de las estructuras buscadas, se eligió un enfoque basado en el aprendizaje no supervisado, en donde los algoritmos k-medias, k-medianas y mapas auto-organizados fueron aplicados a levantamientos magnetométricos con enfoque arqueológico.

Palabras clave: Prospección geofísica, magnetometría, arqueología, interpretación, agrupamiento automático, aprendizaje no supervisado.

Automatic Clustering of Magnetic Data in Geophysical Prospecting for Archeology

Abstract. The interpretation in geophysical prospecting is one of the main objectives, done by analyzing multiple images associated with physical propieties of buried structures. In this paper we propose to apply a clustering methodology to geophysical data using machine learning tools, in order to provide a quantitative tool for interpretation. Since it is not always possible to know the true distribution of the buried structures, an approach based on unsupervised learning was chosen, where the algorithms k-means, k-medians and self-organizing maps were applied to magnetometric surveys with an archaeological approach.

Keywords: Geophysical prospecting, magnetic method, archeology, interpretation, automatic clustering, unsupervised learning.

1. Introducción

La interpretación de datos geofísicos provenientes de los métodos prospectivos consiste en el análisis de múltiples imágenes dentro de un dominio restringido. El objetivo es inferir una distribución de propiedades físicas que expliquen las variables registradas, y por ende, la información recolectada debe ser interpretada dentro de un marco delimitado por las características del objetivo y del contexto geológico, hidrogeológico, arqueológico, etc.

Las interpretaciones generalmente se realizan de forma cualitativa, describiendo geometrías y patrones, y las conclusiones dependen de la experiencia del intérprete y su habilidad de analizar datos multidimensionales.

Algunas de las imágenes usadas para realizar las interpretaciones son el resultado de un modelo inverso o de la aplicación de filtros de realce de anomalías geofísicas. El flujo de trabajo que se aplica de forma general a los levantamientos geofísicos se describe en la literatura, como en [7], y puede resumirse en los siguientes pasos:

1. Adquisición: los datos se registran durante un levantamiento geofísico. Deben adquirirse con la mayor calidad posible.
2. Correcciones y preprocesamiento: los datos de algunos métodos requieren ser preprocesados para remover efectos no deseados. Se puede aplicar una metodología de filtrado para atenuar artefactos dentro de las señales registradas.
3. Procesamiento: los datos se procesan para obtener las imágenes a interpretar.

La magnetometría consiste en realizar mediciones del campo geomagnético en la superficie terrestre. Su objetivo es determinar la distribución de las posibles estructuras que son fuente de los datos registrados, cada dato se atribuye a un punto en superficie.

El procesamiento general aplicado a datos de magnetometría puede consultarse en [13, 2, 14]. Algunos detalles relevantes para este trabajo se revisan en la Sección 2 de este documento. En este trabajo proponemos una metodología basada en algoritmos de aprendizaje no supervisado para agrupar datos magnéticos, ya que no siempre es posible conocer la distribución real de las anomalías en el subsuelo.

Las herramientas de aprendizaje no supervisado, dentro del área de aprendizaje computacional, son un conjunto de metodologías en las que los modelos generados se crean a partir de las entradas al sistema, desconociendo total o parcialmente las salidas esperadas.

El objetivo es diseñar una herramienta computacional que apoye las conclusiones de un intérprete, obteniendo a la salida una imagen que muestra la distribución espacial de los grupos o conjuntos resultantes de la aplicación de algoritmos basados en k-medias, k-medianas y mapas auto-organizados. Los levantamientos analizados provienen de estudios realizados en dos zonas arqueológicas en México: Xalasco, Tlaxcala y La Ferrería, Durango, Figura 1.

Las bases de datos contienen las variables de algunos de los filtros de realce de anomalías más usados: gradiente vertical, reducción al polo, señal analítica, gradiente horizontal, derivada inclinada y gradiente horizontal de la derivada inclinada, además de la anomalía de campo total.

La etapa de agrupamiento propuesta se añade en la posición 4 al flujo de trabajo geofísico descrito más arriba, después de la generación de las imágenes finales. El agrupamiento resultante muestra una posible distribución de las estructuras que producen la respuesta observada en superficie, a través de las similitudes entre vectores y entre bases de datos.

Los resultados se validan correlacionándolos con los estudios arqueológicos previos liderados por interpretaciones de expertos en el área, mostrando que esta metodología puede aplicarse para apoyar a la interpretación geofísica. La implementación, escrita en scripts en python, se encuentra disponible en [11].

1.1. Revisión de literatura

Con la finalidad de reducir la incertidumbre en la interpretación de datos geofísicos multidimensionales, algunos autores han desarrollado métodos como la denominada inversión conjunta [10]. En este trabajo los autores asumen que dos métodos espacialmente coincidentes deben ser sensibles a la misma geología. Desarrollaron una relación de gradientes cruzados para evaluar características estructurales existentes en ambos métodos.

Por otro lado, se han aplicado técnicas de aprendizaje computacional para la clasificación automática de litología. En [16] se realiza una revisión de herramientas de aprendizaje automático aplicadas a datos de percepción remota. Los autores estudiaron cinco conjuntos de metodologías: reducción de dimensionalidad, clasificación, agrupamiento, regresión y aprendizaje profundo.

Cada conjunto estudiado parece ser una buena alternativa para procesar datos de percepción remota, resolviendo inconvenientes asociados con el mapeo de estructuras geológicas de gran escala y que pueden ser verificadas. En [9], los autores hicieron un mapeo predictivo de enjambres de diques al NE de Brazil, aplicando un algoritmo de SOM a datos de magnetometría aérea y de espectrometría de rayos gamma.

Esta técnica semiautomática permite una clasificación de unidades litológicas, los resultados fueron correlacionados con trabajo de campo. Otros autores han desarrollado herramientas computacionales para automatizar, o semi-automatizar, el mapeo geológico y litológico mediante la aplicación de bosques aleatorios. En [8] se usan estas metodologías para clasificar estructuras de gran escala, cuya existencia puede ser verificada.

2. Teoría

2.1. Geofísica

De acuerdo con [15], la geofísica aplicada consiste en la medición e interpretación de las propiedades físicas de la Tierra para determinar las condiciones del subsuelo. Los métodos geofísicos responden a los contrastes de las propiedades físicas de estructuras enterradas. Ningún método concluye con una interpretación única, por lo que es recomendable aplicar más de un método, o calcular más de un atributo, para facilitar la detección de la anomalía buscada, información al respecto se puede consultar en [3].

Es de suma importancia que los datos sean interpretados dentro de un marco definido por las características de la zona de estudio. La interpretación se lleva a cabo analizando múltiples imágenes que pertenecen al mismo dominio espacial.

Estas imágenes pueden provenir de diferentes métodos o de la aplicación de filtros de realce de anomalías. La interpretación normalmente es un proceso cualitativo, por lo que puede haber múltiples interpretaciones para un mismo levantamiento.

La magnetometría [1] es un método de exploración geofísica basado en la medición del campo geomagnético con el objetivo de inferir la distribución de cuerpos o estructuras magnetizables. Las lecturas se realizan en la superficie terrestre con magnetómetros, donde cada dato se atribuye a un punto en la superficie. Los magnetómetros de campo total miden la magnitud del campo geomagnético, incluyendo las contribuciones de campos internos y externos.

Para estudiar las anomalías locales es necesario retirar las contribuciones del campo principal y de los campos externos. Este proceso se realiza numéricamente en un preprocesamiento. Comúnmente, para interpretar datos provenientes de levantamientos magnéticos, se analizan las imágenes resultantes de la aplicación de los siguientes filtros:

- **Reducción al polo**: Cambia matemáticamente el ángulo de inclinación del campo geomagnético, obteniendo entonces la respuesta de las anomalías a una magnetización vertical.
- **Gradiente vertical**: Define la taza de cambio vertical del campo geomagnético.
- **Gradiente horizontal**: Es el módulo de las derivadas en dirección x y en dirección y.
- **Señal analítica**: Se puede ver como la combinación de los gradientes vertical y horizontal, calculada mediante la transformada de Hilbert.
- **Derivada inclinada**: $\mathrm{TDR} = \tan^{-1}\left(\dfrac{\mathrm{GV}}{\mathrm{GH}}\right)$.
- **Gradiente horizontal de la derivada inclinada**: cuyo cálculo es equivalente al del gradiente horizontal.

Debido a la complejidad de los mapas magnéticos resultantes, la interpretación se lleva a cabo de forma cualitativa analizando patrones, continuidad y geometría de las anomalías resaltadas, características que dependen del filtro aplicado.

Los datos pueden ser interpretados como perfiles individuales (curvas en dos dimensiones) o como mapas (imágenes en dos dimensiones cuyos ejes son las coordenadas y el valor del campo se representa en falso color). Algunos ejemplos sobre la aplicabilidad del método en arqueología se pueden consultar en [5, 6].

2.2. K-medias y K-medianas

Los algoritmos de k-medias y k-medianas agrupan los datos usando la distancia como métrica para estimar la similitud entre los vectores de entrada. Estos algoritmos iteran para actualizar la posición de centroides considerando la media o mediana de las distancias de sus vectores más cercanos. Los grupos resultantes están definidos por los vectores más cercanos a cada centroide.

(a) Localización de Xalasco. (b) Localización de La Ferrería.

Fig. 1. Ubicación de las zonas de estudio, resaltadas por un cuadro azul.

La base de datos con vectores agrupados muestra entonces los patrones generales de acuerdo a las similitudes inferidas a partir del cálculo de la media o mediana de distancias, en este trabajo usamos la implementación de scikit-learn [12].

2.3. Mapas auto-organizados

Los Mapas Auto-Organizados, también conocidos como SOM (por su nombre en inglés), son una herramienta de aprendizaje no supervisado que se usa para mapear y visualizar datos multidimensionales en un espacio comúnmente bidimensional, preservando su estructura topológica.

Cada neurona, elemento del mapa de la SOM, tiene un vector de pesos con la misma dimensión que el espacio de características de los datos. Los pesos de las neuronas se actualizan considerando la distancia entre los vectores de entrada y sus respectivas neuronas ganadoras (Best Matching Unit - BMU), así como de la vecindad y de una taza de aprendizaje. La elaboración matemática se puede consultar en [4].

Esta herramienta puede usarse para agrupar datos, donde los grupos resultantes son definidos por las BMU. Comparado con los algoritmos de k-medias y k-medianas, este método ofrece un agrupamiento más robusto, añadiendo la sensibilidad a neuronas vecinas. En este trabajo usamos la implementación minisom [18].

3. Metodología propuesta

3.1. Configuración de los datos

En este paso se eligen las variables que se usarán como entrada. La selección de estas depende de las necesidades del intérprete, pudiendo discriminar entre datos de uno o varios sensores con sus respectivos atributos calculados. Nuestro sistema usa el algoritmo de PCA para reducir la dimensionalidad en las bases de datos. Las variables elegidas para cada conjunto pueden ser aquellas que tengan un comportamiento similar, por ejemplo datos que provienen de sensores independientes. Con este procedimiento se busca disminuir la colinealidad entre variables, de acuerdo con las características

Tabla 1. Enumeración de las variables magnéticas usadas para entrenar y agrupar.

Identificador	Variable	Sensor
1	Campo magnético total	Superior
2	Campo magnético total	Inferior
3	Gradiente vertical	N/A
4	Reducción al polo	Superior
5	Reducción al polo	Inferior
6	Gradiente horizontal	Superior
7	Gradiente horizontal	Inferior
8	Señal analítica	Superior
9	Señal analítica	Inferior
10	Derivada inclinada	Superior
11	Derivada inclinada	Inferior
12	Gradiente horizontal de la derivada inclinada	Superior
13	Gradiente horizontal de la derivada inclinada	Inferior

físicas del levantamiento (cuando se usan dos o más sensores para la adquisición) o de las propiedades de los filtros de realce (pudiendo existir una dependencia prácticamente lineal entre sus salidas). Por otro lado, los datos son escalados antes de la aplicación de los algoritmos de aprendizaje computacional.

3.2. Entrenamiento y agrupamiento

Con el fin de aumentar el desempeño de las herramientas de aprendizaje no supervisado se aplicó un proceso extra para cada caso.

Reentrenamiento para k-medias y k-medianas. Estos métodos pueden enmascarar algunas características en los grupos resultantes. Los datos de magnetometría tienden a agruparse en dos conjuntos principales: los puntos con valores positivos y los puntos con valores negativos.

Añadir más centroides al entrenamiento no mejora la resolución en los conjuntos de salida, tendiendo siempre a los mismos dos conjuntos generales y conjuntos adicionales que describen los vectores atípicos. Para evitar el efecto de sobre-ajuste usamos el siguiente algoritmo:

1. Entrenar c centroides usando el conjunto de entrenamiento completo.
2. Retirar del conjunto de entrenamiento los vectores pertenecientes al centroide más poblado.
3. Entrenar c nuevos centroides usando el conjunto de entrenamiento filtrado resultante del punto anterior.

Con este algoritmo evitamos que los centroides se ubiquen únicamente en las porciones que definen la anomalía principal, pudiendo ahora discriminar estructuras de transición al efecto regional.

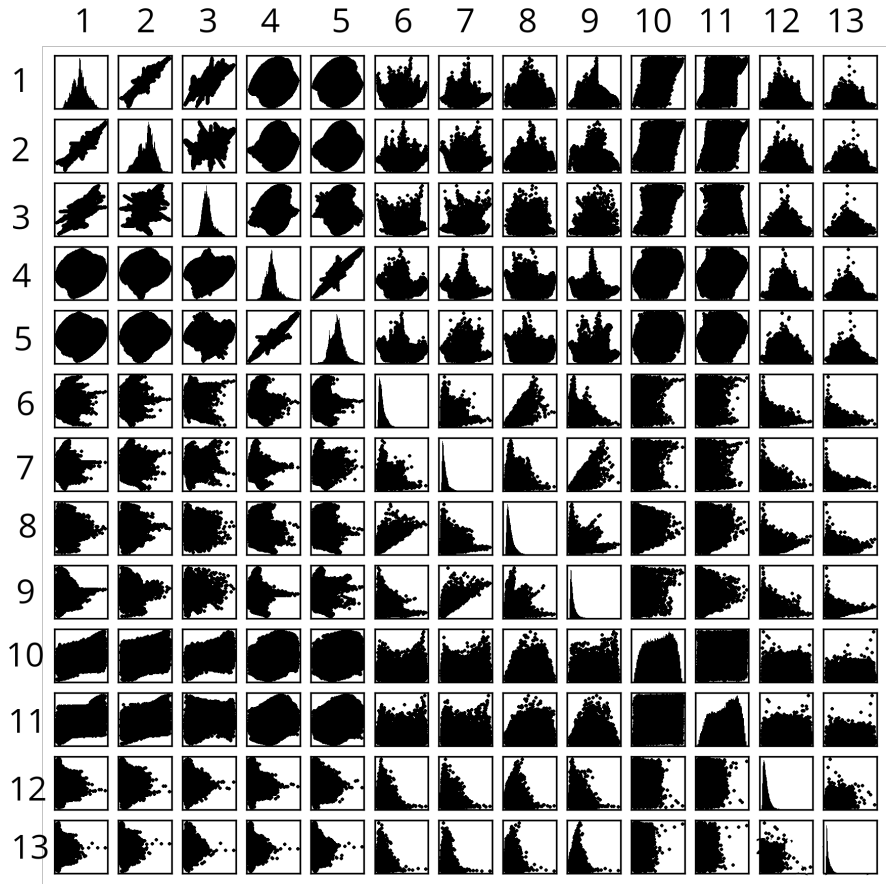


Fig. 2. Matriz de dispersión de la base de datos, en la diagonal se muestra el histograma de cada variable.

Promedio de m SOMs. Los SOM dependen de la inicialización de las neuronas, en consecuencia no siempre los mismos vectores se agruparán en la misma neurona en cada nuevo mapa. Para aumentar la sensibilidad de los SOM, buscando distinguir grupos que pueden comportarse como transición a otros grupos, usamos el siguiente algoritmo:

1. Entrenar m mapas con inicialización aleatoria independiente.
2. Usamos el algoritmo de k-medoides para agrupar las neuronas en c grupos de acuerdo con la similitud entre sus vectores de peso, el número de grupos es elegido por el usuario. A cada conjunto de neuronas se le asigna de forma automática un matiz o color, a partir del cual las neuronas que pertenecen al mismo grupo se diferencian entre sí presentando una saturación de color diferente. Cada vector del conjunto de agrupamiento tendrá una BMU caracterizada por un color: los vectores que tengan el mismo matiz tendrán características similares, siendo del mismo grupo aquellos que además presenten la misma saturación.

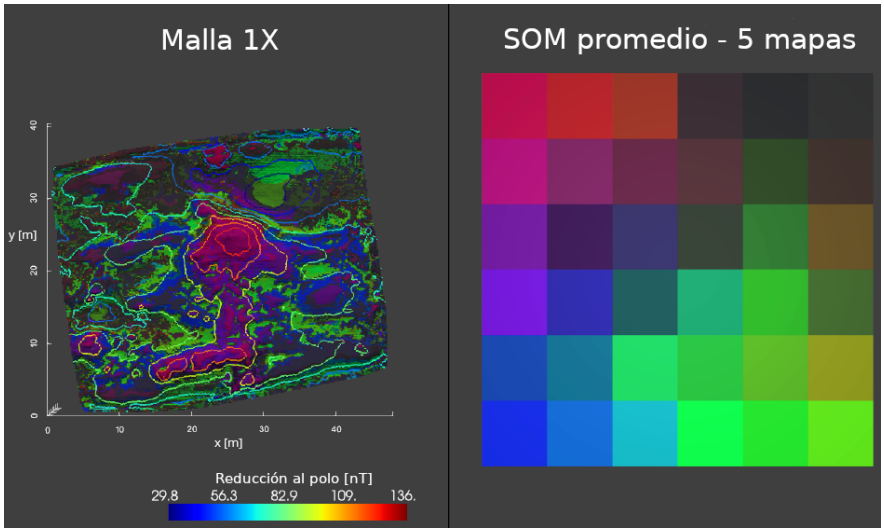


Fig. 3. Imagen de una aplicación del sistema propuesto a la malla 1 de Xalasco, Tlaxcala. Derecha: SOM obtenida tras la aplicación del algoritmo de entrenamiento propuesto. Izquierda: mapa del levantamiento que muestra las estructuras principales inferidas del agrupamiento aplicado, se sobrepone la variable de reducción al polo como mapa de contornos.

3. Para cada vector del conjunto de agrupamiento se computa el color promedio de sus BMU en los m mapas. En este punto se diferencian los vectores que tienen BMUs con matices y saturaciones diferentes.
4. Se entrena un nuevo SOM con los colores obtenidos en el paso anterior. El mapa resultante es usado para agregar la etiqueta de color a los vectores del conjunto de agrupamiento. Con este proceso se reduce la paleta de colores y en consecuencia simplificamos las estructuras resaltadas por las similitudes encontradas en la base de datos.

3.3. Generación de imágenes

En esta etapa usamos la implementación de pyvista [17] para generar la figura final, esta se divide en dos áreas:

1. Distribución de los grupos o el SOM con etiquetas de color para cada elemento.
2. Un mapa con la distribución espacial del levantamiento de la base de datos agrupada. El color de cada píxel corresponde unívocamente al grupo al que pertenece de acuerdo al punto anterior. La continuidad en el color de los píxeles en esta imagen describe las posibles estructuras, debido a que se puede atribuir como la persistencia espacial de las similitudes en la base de datos.

Gráfico de distribución de centroides. El número de centroides es elegido por el usuario en la etapa de entrenamiento, cada uno existe en un espacio de dimensión igual al número de características de la base de datos.

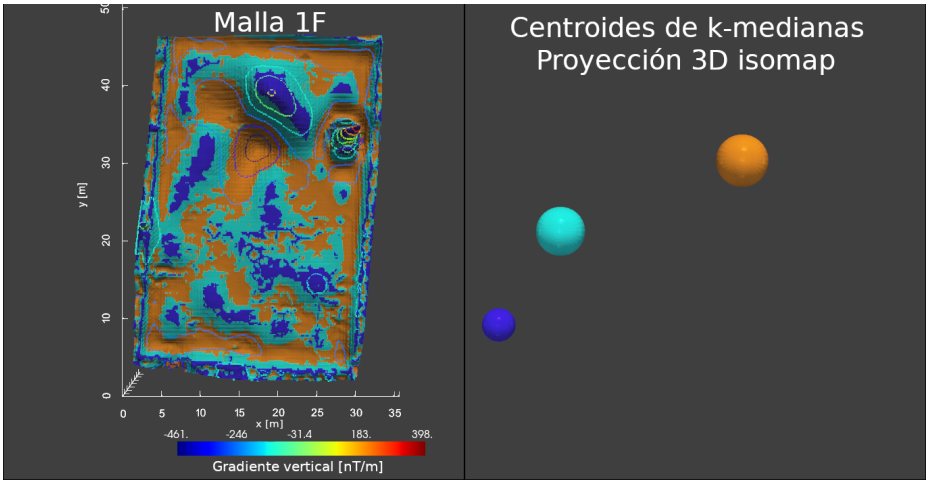


Fig. 4. Imagen de una aplicación del sistema propuesto a la malla 1 de La Ferrería, Durango. Derecha: distribución de centroides proyectados de un espacio de siete dimensiones a un espacio de tres dimensiones. Izquierda: mapa del levantamiento que muestra las estructuras principales inferidas del agrupamiento aplicado, se sobrepone la variable de gradiente vertical como mapa de contornos.

Para visualizar la distribución de los centroides resultantes aplicamos el algoritmo de reducción de dimensionalidad isomap, buscando representar a cada centroide como una esfera en un espacio tridimensional, cuyo radio es proporcional al número de vectores que pertenecen al centroide respectivo, el color que define a cada esfera se asigna de forma automática.

El usuario puede cambiar el nivel de acercamiento y el ángulo de visualización de los centroides, con el fin de analizar las posiciones relativas entre ellos y concluir sobre su similitud o cercanía.

Gráfico del SOM. El algoritmo implementado construye una malla rectangular o hexagonal de tamaño $n \times m$, de acuerdo a la topología ingresada por el usuario. Después de la asignación del color final se construye la malla de forma automática.

Gráfico del mapa. Aquí se sustituye el valor de las variables, cuyas coordenadas están definidas por los puntos de adquisición del levantamiento magnetométrico, por un píxel cuyo color corresponde al grupo determinado por los métodos de agrupamiento.

En esta imagen se muestra el resultado del agrupamiento de los datos analizados, la continuidad y geometría de los elementos del mismo color pueden interpretarse como elementos de una misma estructura, ya que estos píxeles tienen un comportamiento similar.

Los colores deben ser interpretados junto con la distribución de los centroides o junto con el SOM: la definición de las estructuras resaltadas depende de la distribución de los centroides o de las neuronas. Para apoyar las conclusiones del intérprete añadimos la opción de añadir un mapa de contornos sobre el mapa con vectores agrupados (e.j. componente total, reducción al polo, gradiente vertical, etc).

4. Base de datos

La metodología descrita anteriormente se aplicó a datos reales, en esta sección se describirá de forma breve la ubicación de los levantamientos y el comportamiento de las variables que se usaron para entrenar y agrupar.

4.1. Zonas de prueba

Las bases de datos usadas en este trabajo provienen de dos sitios arqueológicos: Xalasco, Tlaxcala (figura 1a) y La Ferrería, Durango (figura 1b), en México. Contamos con seis mallas de magnetometría para cada localidad, datos que fueron brindados por el departamento de geofísica de la Facultad de Ingeniería de la UNAM.

En todos los levantamientos se usó un gradiómetro, este instrumento puede definirse como un magnetómetro con dos sensores. Se tienen entonces dos lecturas de componente de campo total, dos componentes resultantes de cada filtro de realce de anomalías (uno para cada sensor) y la lectura directa del gradiente vertical. Cada malla fue corregida y preprocesada apropiadamente.

4.2. Configuración de los datos

La figura 2 muestra los gráficos de dispersión e histogramas de una muestra aleatoria del $60\,\%$ de la base de datos completa, las variables se enumeran en el cuadro 1. Los histogramas se encuentran bien distribuidos y los gráficos de dispersión son uniformes sin mostrar dependencias fuertes, excepto para las componentes de campo total y la reducción al polo en ambos sensores.

Al contar con dos versiones para cada variable (excepto para el gradiente vertical), contamos con una base de datos con 13 características y de $1,416,808$ vectores considerando las 12 mallas.

5. Resultados y discusión

Se aplicó la metodología propuesta en la sección 3 a la base de datos descrita en la sección 4. El proceso se repitió varias veces con distintas configuraciones de parámetros, se recurrió a un análisis cualitativo para elegir la mejor imagen. Las características consideradas fueron:

1. Detalle: la imagen muestra elementos que pueden ser interpretados como zonas de transición de las estructuras principales, estas últimas definidas por su geometría, relacionándose con el objetivo del estudio.
2. Definición de anomalías geofísicas: la geometría de las estructuras visualizadas tienen una tendencia espacial clara.
3. Paleta de colores: los colores de la imagen resultante tienen un buen contraste que facilita la visualización.

4. Distribución de los centroides o estructura del SOM: para los algoritmos k-medias y k-medianas, los centroides resultantes no se enciman; para SOM, la u-matrix (matriz de distancias unificada) y el hit-map (mapa de aciertos) tienen un comportamiento homogéneo.
5. Correlación con evidencia arqueológica: las estructuras definidas por el agrupamiento tienen correlación directa con estudios arqueológicos previos.

5.1. Malla 1 Xalasco, Tlaxcala

La figura 3 muestra el resultado de un agrupamiento, usando el algoritmo SOM, aplicado a la malla 1 de Xalasco, nombrada en este trabajo como 1X. Los parámetros con los que se obtuvo el mejor resultado fueron:

– El entrenamiento se realizó sobre las variables del sensor inferior y el gradiente vertical, ya que los objetivos arqueológicos son someros. Se eligió de forma aleatoria al conjunto de entrenamiento, conformada por el $60\,\%$ de la base de datos completa.

– Relativo al SOM:

  – Se usó una red rectangular de 6×6, con la que se obtuvieron errores bajos y u-matrix con grupos claramente definidos. El SOM resultante se obtuvo a partir de cinco SOM con diferentes inicializaciones, en los que sus neuronas fueron respectivamente agrupadas en 3 conjuntos con tonos rojos, verdes y azules.

  – Se realizaron 5000 iteraciones, obteniendo con esto errores topológicos y de cuantización de $25\,\%$ y de 0.1 respectivamente.

  – Radio de vecindad del $30\,\%$.

  – Taza da aprendizaje de 0.1.

Respecto a la validación cualitativa:

– Detalle: se aprecian zonas de transición coherentes que delimitan las estructuras principales.
– Definición de anomalías geofísicas: los grupos resultantes preservan geometrías continuas y patrones estructurados.
– Paleta de colores: los colores principales son azul, lila y verde, la distribución de tonos resulta en una imagen con suficiente contraste para ser analizada.
– Estructura del SOM: los grupos resultantes de la aplicación del SOM se encuentran bien distribuidos, siendo capaz de definir de forma clara cuatro conjuntos principales.
– Correlación arqueológica: de acuerdo con [6] las excavaciones revelan la existencia de una estructura antropogénica al centro de la malla, corresponde con un suelo fragmentado con orientación norte-sur. Esta estructura tiene correlación con los elementos definidos por los tonos lila.

De acuerdo con la reducción al polo, los tonos lila y azul corresponden a la anomalía principal, que de acuerdo al SOM son elementos muy diferentes a los elementos definidos por los los tonos verdes.

La metodología de agrupamiento distingue dos estructuras principales, definidas por los cambios de lila a azul. Existe correlación con evidencia arqueológica mostrando discontinuidades que pueden atribuirse a la destrucción del suelo como resultado de actividad humana contemporánea.

5.2. Malla 1 La Ferrería, Durango

La figura 4 muestra el resultado de un agrupamiento, usando el algoritmo k-medianas, aplicado a la malla 1 de La Ferrería, nombrada en este trabajo como 1F. Los parámetros con los que se obtuvo el mejor resultado fueron:

- El entrenamiento se realizó sobre las variables del sensor inferior y el gradiente vertical, ya que los objetivos arqueológicos son someros. Se eligió de forma aleatoria al conjunto de entrenamiento, conformada por el $60\,\%$ de la base de datos completa.
- Relativo al algoritmo de k-medianas:
  - Se entrenó el sistema con tres centroides, esperando un modelo con tres estructuras principales: una anomalía principal, una zona de transición y una estructura con comportamiento regional.
  - Los índices de Davies-Boudin y de Silhouette resultantes de este procedimiento fueron 1.5 y 0.18 respectivamente.
- Respecto a la validación cualitativa:
  - Detalle: a pesar de que k-medianas es menos robusto que SOM, la imagen resultante muestra estructuras continuas sin perder continuidad por elementos de una clase diferente con comportamiento aparentemente aleatorio. Este comportamiento probablemente se deba a que las estructuras reales en el subsuelo son más profundas.
  - Definición de anomalías geofísicas: el agrupamiento resultante define de forma clara tres estructuras diferentes que corresponden a la anomalía principal, a una anomalía con comportamiento regional y a una estructura de transición entre ambas.
  - Paleta de colores: la imagen agrupada se describe con tres colores: azul, cyan y naranja, cuyo contraste es suficiente para analizar las estructuras.
  - Distribución de los centroides: de la proyección con isomap se concluye que los centroides no se sobreponen entre sí, siendo elementos que pueden describir el fenómeno analizado. Gracias a este entorno interactivo podemos concluir que la inferida estructura de transición corresponde a los colores del centroide cyan, ya que se localiza entre los otros dos.
  - Correlación arqueológica: aún no hay evidencia en trabajos publicados.

De acuerdo con el gradiente vertical, se puede atribuir que el color azul corresponde a la anomalía principal. La estructura conformada por estos elementos presentan alineaciones con orientación Sur-Norte hacia el oeste, y oeste-este hacia el sur.

Los resultados obtenidos en ambas zonas de estudio muestran que el sistema planteado en este trabajo es capaz de resaltar patrones en las bases de datos, pudiendo correlacionarse con evidencia de excavaciones arqueológicas.

La continuidad en las similitudes de los vectores que conforman las mallas de magnetometría se interpretan como las estructuras que son fuente de este método geofísico, cuyas geometrías y diferencias entre sí (identificadas por los cambios en tono o color) comprenden desde propiedades físicas diferentes (e.j. permeabilidad magnética) hasta la relación con la profundidad.

6. Conclusiones

La metodología propuesta es capaz de generalizar las estructuras y patrones en las bases de datos analizadas. Las imágenes resultantes muestran estructuras que resultan lógicas de acuerdo con el objetivo. Las características resaltadas, producto del agrupamiento realizado con herramientas de aprendizaje no supervisado, pueden ayudar a las conclusiones de un intérprete, quien añade el contexto físico al resultado de la metodología aquí propuesta.

Los resultados del agrupamiento muestran correlación con las excavaciones realizadas en la zona de Xalasco, brindando un resultado cuantitativo para apoyar en la decisión de áreas para excavaciones futuras. Respecto a la zona de La Ferrería, no hay información sobre excavaciones en los sectores estudiados, sin embargo se pueden observar alineaciones que pueden ser lógicas de acuerdo a las estructuras antropogénicas esperadas.

Los algoritmos k-medias y k-medianas producen resultados que muestran patrones muy generales, comportamiento que es esperado cuando las estructuras buscadas no son superficiales, estos algoritmos tienden a enmascarar estructuras pequeñas.

Por otro lado, el SOM es capaz de detectar cambios sutiles, comportamiento que es esperado cuando las estructuras buscadas son someras, sin embargo tiende a presentar un comportamiento ruidoso cuando las estructuras son profundas. Ambas metodologías son aplicables para apoyar a la interpretación geofísica, aplicadas de forma conjunta pueden reducir la incertidumbre en la toma de decisiones.

En este trabajo se analizan bases de datos multidimensionales provenientes únicamente de datos magnéticos, sin embargo no se descarta que la metodología descrita tenga un buen desempeño al mezclar datos de distintas prospecciones, siendo cada una sensible a distintas propiedades de la materia.

Si los distintos levantamientos geofísicos son espacialmente coincidentes, es lógico pensar que la respuesta de cada uno es sensible a la existencia de una distribución de propiedades fija, la metodología de agrupamiento buscaría entonces la similitud entre cada distribución asumiendo que provienen de un mismo fenómeno en el subsuelo, siendo entonces una alternativa a la inversión conjunta.

Como propuesta de trabajo futuro se propone la implementación de este sistema que reciba datos de distintas prospecciones a la entrada, además de la posibilidad de generar modelos a partir de datos sintéticos para añadir una etapa de validación adicional.

Agradecimientos. Investigación realizada gracias al Programa PAPIIT - UNAM IG101222.

Referencias

1. Blakely, R. J.: Potential theory in gravity and magnetic applications, Cambridge university press (1996)
2. Bournas, N., Bake, H. A.: Interpretation of magnetic anomalies using the horizontal gradient analytic signal. Annals of Geophysics, vol. 44, no. 3 (2009) doi: 10.4401/ag-3572
3. Butler, D. K., Bennett, H. H., Ballard, J. H.: Overview of multimethod geophysical system development for enhanced near-surface target detection, discrimination, and characterization. The Leading Edge, vol. 25, no. 3, pp. 352–356 (2006) doi: 10.1190/1.2184105
4. Davydov, A., Larionov, A., Nagul, N.: The formal description of discrete-event systems using positively constructed formulas. In: 40th International Convention on Information and Communication Technology, Electronics and Microelectronics (2017) doi: 10.23919/mipro.2017.7973599
5. Fassbinder, J. W. E.: Magnetometry for archaeology. Encyclopedia of Geoarchaeology, pp. 499–514 (2017) doi: 10.1007/978-1-4020-4409-0_169
6. Juárez, K., López-García, P., Argote-Espino, D. L., Tejero-Andrade, A., Chávez, R. E., García-Serrano, A.: Magnetic and electrical prospections in the archaeological site of Xalasco northeast of Tlaxcala, Mexico. Global Journal of Archaeology and Anthropology, vol. 2, no. 2, pp. 555–581 (2017) doi: 10.19080/GJAA.2017.02.555581
7. Khesin, B. E., Alexeyev, V. G., Eppelbaum, L. V.: Interpretation of geophysical fields in complicated environments. Springer Science and Business Media, vol. 14 (2013) doi: 10.1007/978-94-015-8613-9
8. Kuhn, S., Cracknell, M. J., Reading, A. M.: Lithologic mapping using random forests applied to geophysical and remote-sensing data: A demonstration study from the eastern goldfields of Australia. Geophysics, vol. 83, no. 4, pp. B183–B193 (2018) doi: 10.1190/geo2017-0590.1
9. Melo, A. C. C., de Castro, D. L., Fraser, S. J., Filho, A. A. M.: Using self-organizing maps in airborne geophysical data for mapping mafic dyke swarms in NE Brazil. Journal of Applied Geophysics, vol. 192, pp. 104377 (2021) doi: 10.1016/j.jappgeo.2021.104377
10. Oliver-Ocaño, F., Gallardo, L., Romo-Jones, J., Pérez-Flores, M.: Structure of the cerro prieto pull-apart basin from joint inversion of gravity, magnetic and magnetotelluric data. Journal of Applied Geophysics, vol. 170, pp. 103835 (2019) doi: 10.1016/j.jappgeo.2019.103835
11. Ortiz, M.: Interprete geofisica. (2023) github.com/CecilRamza/Interprete_geofisica
12. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in python. the Journal of machine Learning research, vol. 12, pp. 2825–2830 (2011)
13. Rajagopalan, S.: Analytic signal vs. reduction to pole: Solutions for low magnetic latitudes. Exploration Geophysics, vol. 34, no. 4, pp. 257–262 (2003)
14. Salem, A., Williams, S., Fairhead, D., Smith, R., Ravat, D.: Interpretation of magnetic data using tilt-angle derivatives. Geophysics, vol. 73, no. 1, pp. L1–L10 (2008) doi: 10.1190/1.2799992
15. Sheriff, R. E.: Encyclopedic dictionary of applied geophysics. Society of exploration geophysicists, pp. 442 (2002) doi: 10.1190/1.9781560802969

16. Shirmard, H., Farahbakhsh, E., Müller, R. D., Chandra, R.: A review of machine learning in processing remote sensing data for mineral exploration. Remote Sensing of Environment, vol. 268, pp. 112750 (2022) doi: 10.1016/j.rse.2021.112750
17. Sullivan, C., Kaszynski, A.: Pyvista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (vtk). Journal of Open Source Software, vol. 4, no. 37, pp. 1450 (2019) doi: 10.21105/joss.01450
18. Vettigli, G.: MiniSom: Minimalistic and numpy-based implementation of the self organizing map (2018) github.com/JustGlowing/minisom/

ISSN 1870-4069

Aprendizaje de similitud semántica para el reconocimiento del alfabeto de lengua de señas

Atoany Nazareth Fierro Radilla[1], Regina Alexia Blas Flores[1], Emiliano Vivas Rodriguez[1], Karina Ruby Perez Daniel[2], Gibran Benitez-Garcia[3]

[1] Tecnológico de Monterrey, Campus Cuernavaca, México

[2] Universidad Panamericana, Facultad de Ingeniería, México

[3] The University of Electro-Communications, Japón

afierror@tec.mx, karinaperez@up.edu.mx, gibran@ieee.org

Resumen. La lengua de señas es un importante método de comunicación que utilizan las personas que padecen alguna enfermedad auditiva, especialmente personas con problemas del habla y/o del escucha. En los Estados Unidos, aproximadamente dos millones de personas que viven con discapacidad auditiva utilizan ASL (por sus siglas en ingles American Sign Language). Por lo tanto, el objetivo de este estudio es investigar y desarrollar un sistema de reconocimiento para el alfabeto ASL, haciendo uso de redes neuronales convolucionales (VGG16 y Mobilenet) siamesas (dos arquitecturas iguales) para poder mejorar la comunicación y las relaciones interpersonales con las personas que viven con discapacidades auditivas. La propuesta se basa en implementar el aprendizaje de similitud semántica para reducir la variación intraclase y la similitud interclase de imágenes del alfabeto de la lengua de señas en un espacio euclidiano. Los resultados muestran que el sistema propuesto tiene una precisión promedio de 99 % y 90.1 % en el conjunto de datos MINIST y ASL, respectivamente. Utilizando la técnica de análisis estadística llamada t-SNE se puede demostrar por qué nuestra propuesta supera los algoritmos reportados en la literatura.

Palabras clave: ASL, lengua de señas, aprendizaje de similitudes, CNN, red siamesa.

Semantic Similarity Learning for American Sign Language Alphabet Recognition

Abstract. Sign language is an important manner to convey information among deaf community, and it is primarily used by people who have hearing or speech impairments. In USA, approximately two million of deaf people use ASL (American Sign Language). Therefore, the purpose of this study is to

investigate and develop a system for ASL alphabet recognition using two Convolutional Neural Networks (CNN), such as VGG16 and Mobilenet. The proposal is to implement semantic similarity learning in order to reduce the high intra-class variation and the high inter-class similarity in an euclidean space of sign images. The results show that the proposed system improves the ASL alphabet recognition in a considerable way, yielding an average accuracy of 99% and 90.1% on MNIST Dataset and ASL Dataset, respectively. Using an statistical analysis technique, called t-SNE we can demonstrate why our proposal outperform the methods reported in literature.

Keywords: ASL, sign language, semantic similarity, CNN, siamese network.

1. Introducción

La forma en la que nos conectamos físicamente con el mundo es a través de las manos; realizamos la mayoría de las tareas diarias con ellas. Por otro lado, utilizamos dispositivos periféricos como mouse y el joystick para trabajar con una computadora (Interacción humano-máquina) [24], dicha interacción es un área de investigación multidisciplinaria con diversas aplicaciones en control, robótica, estudio de comportamiento psicológico, realidad virtual, reconocimiento de lengua de señas, visualización científica y, en el Metaverso [24, 2].

Los sistemas de Reconocimiento de Lengua de Señas (RLS) se realizan por medio de la Interacción Humano-Computadora, convirtiendo los gestos y movimientos de manos en comandos de texto y/o de voz, permitiendo la comunicación entre los seres humanos a través de una computadora, entre las personas que presentan una discapacidad auditiva y las personas oyentes [2, 26].

En los Estados Unidos (EE.UU) hay aproximadamente dos millones de personas sordas, algunos de ellos nacen con pérdida auditiva en ambos oídos, mientras que otros pierden la audición debido a factores como la rubéola y la meningitis [24].

La lengua de señas americana (ASL) es el segundo idioma distinto del inglés más utilizado en los EE. UU. después del español, tiene 36 formas de manos, 26 letras y alrededor de 6000 palabras, que consisten en movimientos corporales complejos. Las señas se crean usando la mano derecha, la mano izquierda, ambas manos y expresiones faciales y/o corporales [24, 23].

A pesar de que ASL es el principal modo de comunicación para la mayoría de las personas sordas en EE. UU., siguen existiendo problemas de comunicación con las personas oyentes ya que no comprenden el lenguaje ASL. Si el ASL se pudiera traducir automáticamente en texto o voz en inglés y/o español, será mucho más fácil para las personas sordas sentirse incluidos y tener mayor comunicación [24]. Los prinpales aportes de este trabajo son:

- Desarrollo de un algoritmo para el reconocimiento del alfabeto ASL.
- Comparación de los resultados de clasificación entre arquitecturas simples y siamesas.
- Comprobación de nuestra hipótesis utilizando la técnica t-SNE.

2. Sistema de reconocimiento del alfabeto de lengua de señas

Desde hace más de dos décadas, se han publicado diversos trabajos de investigación para el reconocimiento de la lengua de señas con grandes avances, en especial para los estadounidenses [26], australianos [12], los coreanos [17] y el chino [14], sin embargo, existen grandes dificultades debido a la complejidad de los movimientos de manos y cuerpo en expresiones en lengua de señas (LS) [24].

Los enfoques utilizados para resolver los problemas de SLR se pueden clasificar en dos métodos principales, sensores y visión por computadora [2, 26]. En los enfoques basados en sensores, se suele usar un guante o sensor especial para rastrear la orientación, la posición, la rotación y los movimientos de la mano [24, 26, 3], los cuales proporcionan información precisa de la posición de la mano [24, 23].

Sin embargo, son demasiado pesados e incómodos para el uso diario [26]. Por otro lado, los métodos basados en visión por computadora consisten en técnicas de procesamiento de imágenes y aprendizaje automático para capturar y clasificar el movimiento del cuerpo y la forma de la mano usando imágenes a color sin necesidad de sensores conectados al ser humano [24, 3]. Este documento se centra únicamente en el enfoque basado en la visión.

3. Deletreo en la lengua de señas

El deletreo con los dedos es la representación de cada letra del alfabeto mediante una seña. Los usuarios de ASL usan el alfabeto en inglés deletreado con los dedos y manos (AFA, American Finger-spelled Alphabet), mientras que usuarios de otro alfabeto, como por ejemplo el mexicano o el chino, utilizan diferentes variaciones de deletreo. El AFA consta de 22 configuraciones de mano que cuando se mantienen en ciertas posiciones y/o se producen ciertos movimientos, se representan las 26 letras del alfabeto inglés [24, 23].

Las investigaciones coinciden en que el deletreo en lengua de señas está integrado en ASL de manera muy sistemática [9]. Uno de los principales usos del letreo es principalmente para representar nombres propios, nombre de medicamentos o palabras en inglés sin equivalentes en lengua de señas [5].

Además, el deletreo es una parte importante de la lengua de señas para los nuevos usuarios y ayuda a las personas a abreviar señas más largas, a comunicar dos palabras compuestas [4] y a cerrar la brecha entre el léxico ASL a través de la geografía y las culturas [3]. La incorporación del deletreo en ASL resulta más conveniente en escenarios críticos, como es en el ámbito de la medicina.

4. Retos en el reconocimiento del alfabeto ASL

La clasificación de los signos depende de las configuraciones de las manos, las cuales son captadas por una cámara de color y/o profundidad. Las complejidades de las señas hacen que el reconocimiento del alfabeto ASL sea una tarea difícil debido a dos factores principales, la similitud entre clases y las variaciones dentro de las clases.

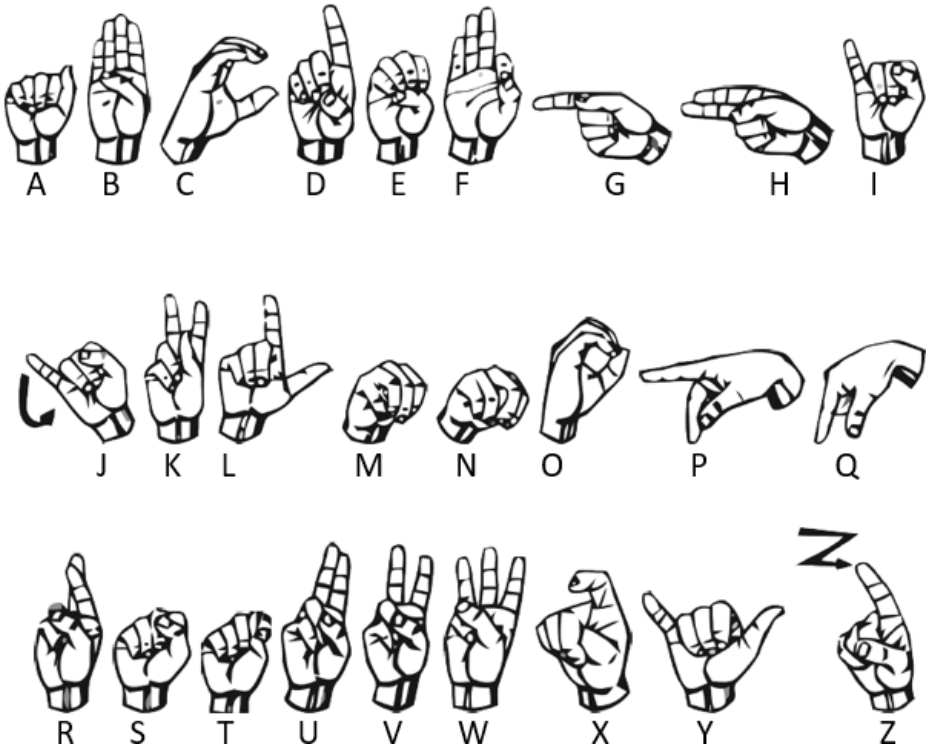


Fig. 1. Alfabeto de la langua de señas americana. Las letras J y Z involucran movimiento.

La similitud entre clases significa que algunas letras están muy estrechamente relacionadas con otras y difieren muy poco en la ubicación de los dedos. Por ejemplo, las letras M y N solo se diferencian entre sí, si el pulgar está entre el primer y el segundo o entre el segundo y el tercer dedo.

Las grandes variaciones intraclase significan que, dentro de una clase, existen diferencias entre las muestras, como la iluminación, el color de la piel, las variaciones del fondo y la posición relativa del usuario con respecto a la cámara.

5. Trabajos relacionados

El reconocimiento del alfabeto ASL se basa principalmente en dos subtareas: extracción de características y clasificación multiclase [26]. En la primer subtarea se realiza la detección y extracción de características locales.

Por otro lado, en la segunda subtarea, éstas características extraídas son comprendidas y caracterizadas para clasificar las muestras [22]. Para el reconocimiento del alfabeto ASL, en la literatura se pueden encontrar dos enfoques que se han utilizado: métodos tradicionales y basados en CNN.

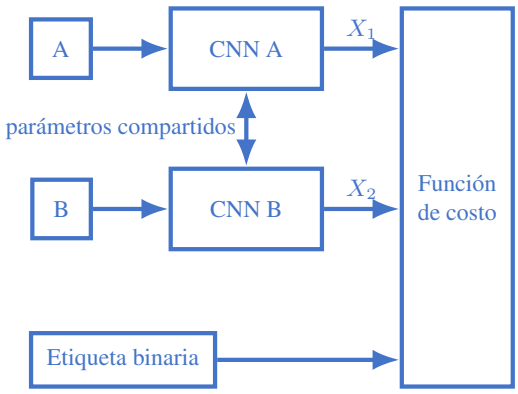


Fig. 2. La arquitectura siamesa está compuesta por dos redes idénticas (Red A y Red B). Estas redes se alimentan con las imágenes A y B. La capa de función de costo se alimenta con los descriptores visuales generados por dichas redes y una etiqueta binaria; esta etiqueta binaria indica si el par es positivo o negativo.

5.1. Métodos tradicionales

En inglés se les conoce como Handcrafted Methods ya que el algoritmo de extracción de características ha sido manualmente construido [16]. Para los enfoques tradicionales es necesario diseñar el mejor algoritmo de extracción de características que mejor se adapte; además, una vez que las características se seleccionan, se debe elegir un clasificador de tal manera que se ajuste con la etapa de extracción de características.

Uno de los primeros sistemas de reconocimiento del alfabeto ASL utiliza los filtros de Gabor para la selección de características y random forest para la clasificación [22], donde los autores obtuvieron un 49 % de precisión.

En [27] los autores popusieron utilizar SP-EMD (Super Pixel Earth Movers Distance, en inglés), el cual es un algoritmo que mide la similitud de las características de forma, textura y profundidad para las imágenes de señas, obteniendo una precisión del 75.8 %. Por otro lado, en lugar de extraer similitudes, los autores en [20] extrajeron textura utilizando espaciogramas volumétricos a partir de los patrones binarios locales (VS-LBP) y utilizando una máquina de soporte vectorial (SVM) como clasificador, obtuvieron una precisión del 83.7 %.

Algunos otros autores [6, 21, 18] consideraron trabajar con la información de profundidad en lugar del color y como clasificador propusieron utilizar bosques aleatorios, obteniendo una precisión de clasificación del 81.1 %, 87 % y 90 %, respectivamente. Los trabajos relacionados mencionados arriba básicamente consisten de dos tareas separadas, extracción de características y clasificación.

Esto produce lo que nosotros llamamos ”fenómeno de desacoplamiento”, donde cierta información se pierde en la etapa de la extracción de características y no logra propagarse hacia la tarea de clasificación.

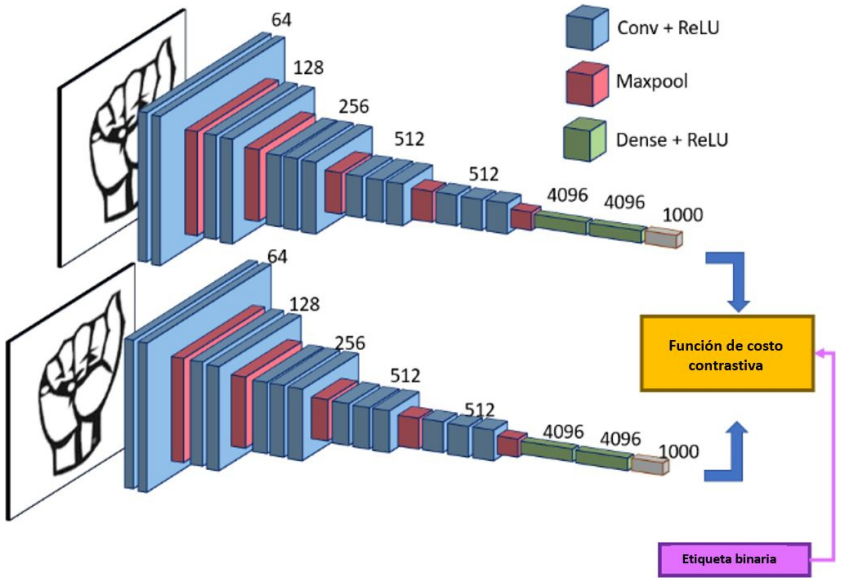


Fig. 3. Arquitectura VGG siamesa. La etiqueta binaria indica la similitud de la imagen. La función de costo contrastiva permite generar descriptores visuales de acuerdo a la similitud entre las imágenes.

5.2. Métodos basados en redes neuronales convolucionales

En el año 2012, las redes neuronales convolucionales (CNN por sus siglas en Inglés) se volvieron muy populares entre la comunidad de visión por computadora debido al éxito de la red Alexnet en la competencia ILSVRC (ImageNet Large Scale Visual Recognition Challenge).

Una de las ventajas de utilizar redes CNN con respecto a los métodos tradicionales es que la extracción de características y la clasificación se realiza en un solo algoritmo sin la intervención humana. En otras palabras, la red aprende qué características son las mejores para extraerse y tener un mejor resultado en la clasificación.

Una red CNN está compuesta por capas convolucionales y capas densas; las primeras permiten obtener representaciones no lineales de imágenes para la extracción de características, mientras que las capas densas son las responsables de la clasificación. Uno de los métodos basados en CNN encontrado en la literatura es [1], donde se propone utilizar una red CNN de dos entradas, una para las imágenes de color y la otra para imágenes de profundidad.

Las convoluciones de estas dos etradas se concatenan y se introducen a las capas densas las cuales obtienen una precisión de clasificación del 80.3 %. Por otro lado, los autores en [26] proponen una estrategia novedosa la cual utiliza solamente imágenes de profundidad para generar una nube de puntos en 3 dimensiones.

En [3] las información de profundidad es también utilizada; los autores utilizaron un sensor de Microsoft Kinnect como extractor de características las cuales fueron clasificadas utilizando PCANet, obteniendo una precisión de 84.5 %.

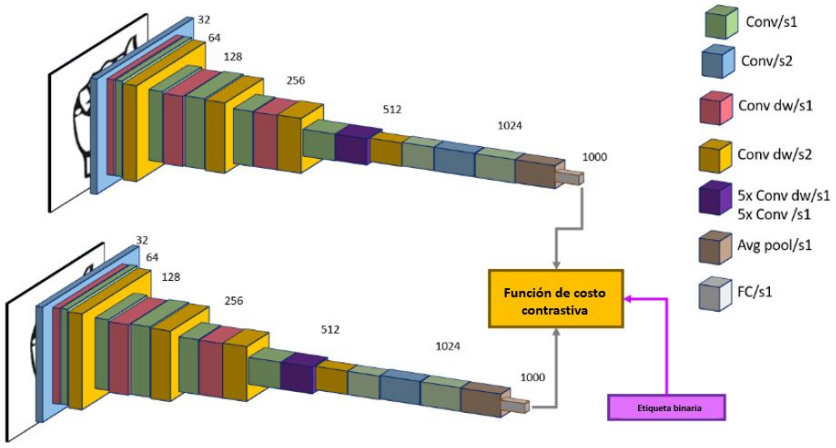


Fig. 4. Arquitectura Mobilenet siamesa. La etiqueta binaria indica la similitud de la imagen. La función de costo contrastiva permite generar descriptores visuales de acuerdo a la similitud entre las imágenes. La última capa coloreada en gris se utiliza como descriptor visual.

A diferencia de los métodos arriba mencionados, en este trabajo utilizamos una arquitectura doble la cual nos entrega dos descriptores cuya distancia entre ellos depende de la similitud de las imágenes. Esta arquitectura doble, mejor conocida como red siamesa se explica a continuación.

6. Arquitectura CNN siamesa

Las redes CNN necesitan entrenarse con una gran cantidad de información, sin embargo, es muy dificil en algunas aplicaciones conseguir tal cantidad de información. En una red neuronal siamesa, el set de entrenamiento se genera mediante combinaciones de pares de imágenes; de esta manera, no se necesita una gran cantidad de muestras. Estos pares de imágenes pueden ser positivos (imágenes de la misma clase) o negativos (imágenes de diferente clases).

Esto hace que las redes neuronales siamesas sean más robustas frente conjuntos de datos sin balancear. Una red siamesa consiste en dos redes neuronales convolucionales idénticas que comparten sus parámetros y se utilizan para aprender similitudes semánticas. La hipótesis de este artículo es que, usando una arquitectura siamesa, se puede reducir la alta similitud interclase y las altas variaciones intraclase, mejorando así el reconocimiento del alfabeto de lengua de señas.

7. Sistema propuesto

Los experimentos se realizaron utilizando dos arquitecturas CNN diferentes, VGG16 y Mobilenet, así como sus versiones siamesas, en dos conjuntos de datos diferentes, Alphabet Dataset [15] y Sign Language MNIST [25]. El entrenamiento se hizo utilizando Keras y Tensorflow en una máquina Intel Core i7 con una GPU NVIDIA GeForce RTX 2070 SUPER.

Tabla 1. Reporte de clasificación de la arquitectura VGG16.

	VGG16		VGG16 Siamesa	
Métrica	**MNIST**	**ASL Alphabet**	**MNIST**	**ASL Alphabet**
Exactitud	0.58	0.30	0.99	0.89
Precisión	0.71	0.47	0.99	0.88
Exhaustividad	0.58	0.30	0.99	0.89
Puntaje F1	0.56	0.28	0.99	0.89

7.1. Datasets

Para los experimentos, se utilizaron los conjuntos de datos del lenguaje de señas MNIST y ASL Alphabet. El conjunto de datos MNIST de lenguaje de señas está compuesto por 34,627 imágenes de tamaño 28x28 píxeles divididas en 24 clases (de la A a la Z, no contienen muestras para J y Z debido involucran movimientos de gestos). Por otro lado, el conjunto de datos de ASL Alphabet está compuesto por 87,000 imágenes de 200x200x3 divididas en 29 clases (de la A a la Z) y 3 clases más etiquetadas como ”SPACE”, ”DEL” y ”NOTHING”; en este trabajo, “J” y “Z” se consideran signos estáticos.

7.2. VGG16

VGG16 se propuso por primera vez en [25] y logró una precisión de prueba del 92,7 % en ImageNet [7], que es un conjunto de datos compuesto por más de 14 millones de imágenes de 1000 clases diferentes. Este modelo participó en el reto ILSVRC-2014, mejorando el rendimiento de Alexnet al reducir el tamaño de los kernel a 3×3. La función de pool utilizada por la red VGG16 es una capa de de 2×2 con un paso de 2.

Esta arquitectura tiene 3 capas densas, seguidas de una capa softmax como salida. La principal contribución de VGG16 es que, en lugar de tener una gran cantidad de hiperparámetros, los autores se enfocaron en tener capas convolucionales de 3×3 con un stride de 1 ($s = 1$). En el entrenamiento de ambos conjuntos de datos, las imágenes se redimensionaron a $224 \times 224 \times 3$ debido a los requisitos de entrada de la red.

La red se entrenó primero en el conjunto de datos del lenguaje de señas del MNIST durante 100 épocas. Se utilizaron 24,720 imágenes como conjunto de entrenamiento y 2,735 para el conjunto de validación, logrando una precisión de entrenamiento de 0.9231, pérdida de entrenamiento de 5.3653, precisión de validación de 0.9872 y pérdida de validación de 0.3970.

Por otro lado, para el entrenamiento con el conjunto de datos ASL Alphabet, el número de épocas fue de 50 debido a la capacidad de la memoria de la máquina. En este experimento se utilizaron 78,300 imágenes de $224 \times 224 \times 3$ como conjunto de entrenamiento y 8,700 imágenes de $224 \times 224 \times 3$ como conjunto de validación. Los resultados utilizando el conjunto de entrenamiento fueron: exactitud de 0.9285, y error de 5.5736. Para el conjunto de validación se obtuvo una exactitud de 0.8477 y un error de 16.2988.

Tabla 2. Reporte de clasificación de la arquitectura Mobilenet.

	Mobilenet		**Mobilenet Siamesa**	
Métrica	**MNIST**	**ASL Alphabet**	**MNIST**	**ASL Alphabet**
Exactitud	0.08	0.04	1.00	0.91
Precisión	0.08	0.07	1.00	0.91
Exhaustividad	0.08	0.04	1.00	0.91
Puntaje F1	0.04	0.01	1.00	0.91

7.3. Mobilenet

Mobilenet fue propuesto en [13]. Este modelo ligero utiliza convoluciones separables en profundidad, lo que reduce el número de parámetros en comparación con las redes con convoluciones regulares con la misma profundidad. Además, en Mobilenet se realiza una sola convolución en cada canal de color en lugar de combinarlos y aplanarlos. Como aplica su nombre, Mobilenet está diseñado para ser utilizado en aplicaciones móviles.

Las dimensiones de las imágenes fueron de $224 \times 224 \times 3$ para realizar los experimentos en las mismas condiciones que en VGG16. Para el conjunto de datos de lengua de señas de MNIST, se utilizaron 24,720 imágenes como conjunto de entrenamiento y 2,735 imágenes como conjunto de validación; utilizando el conjunto de entrenamiento se obtuvo una exactitud de 0.9989 y un error de 0.0042.

Con el conjunto de validación se obtuvo un valor de exactitud de 1.00 y un error de 0.0003. En el caso del conjunto de datos del Alfabeto ASL, el cual está conformado por 78,300 imágenes de entrenamiento y 8,700 imágenes de validacion. Utilizando el set de entranamiento se obtuvo una exactitud de 0.9947 y un error de 0.0158. Por otra parte, con el set de validación se obtuvo un valor de exactitud de 0.9291 y un error de 0.2754.

7.4. VGG siamesa

Para el aprendizaje de similitud semántica, se utilizaron dos VGG16 tipo D idénticas; estas dos redes comparten sus parámetros. La etiqueta binaria indica la similitud del par de imágenes. El número de neuronas en la última capa es 1000 para ambos conjuntos de datos.

Aquí, en el entrenamiento siamés, la última capa no contiene la probabilidad de que una imagen pertenezca a una determinada clase, sino un descriptor visual el cual es una codificación para representar el contenido de la imagen en un espacio euclidiano. Cuanto mayor sea la dimensión de la codificación de la imagen, mejor será la representación de la imagen; sin embargo, los descriptores visuales de grandes dimensiones representan una mayor demanda de recursos informáticos.

Después de varios experimentos, 1000 neuronas en la última capa mostraron el mejor rendimiento teniendo en cuenta la compensación entre la complejidad del cálculo, la precisión y las limitaciones del hardware.

En el caso del conjunto de datos de lenguaje de señas del MNIST, el tamaño de la imagen original es $28 \times 28 \times 1$; sin embargo, el tamaño de entrada mínimo para VGG16 es $32 \times 32 \times 1$ y, debido a que VGG16 es una red profunda, se decidió usar $64 \times 64 \times 1$

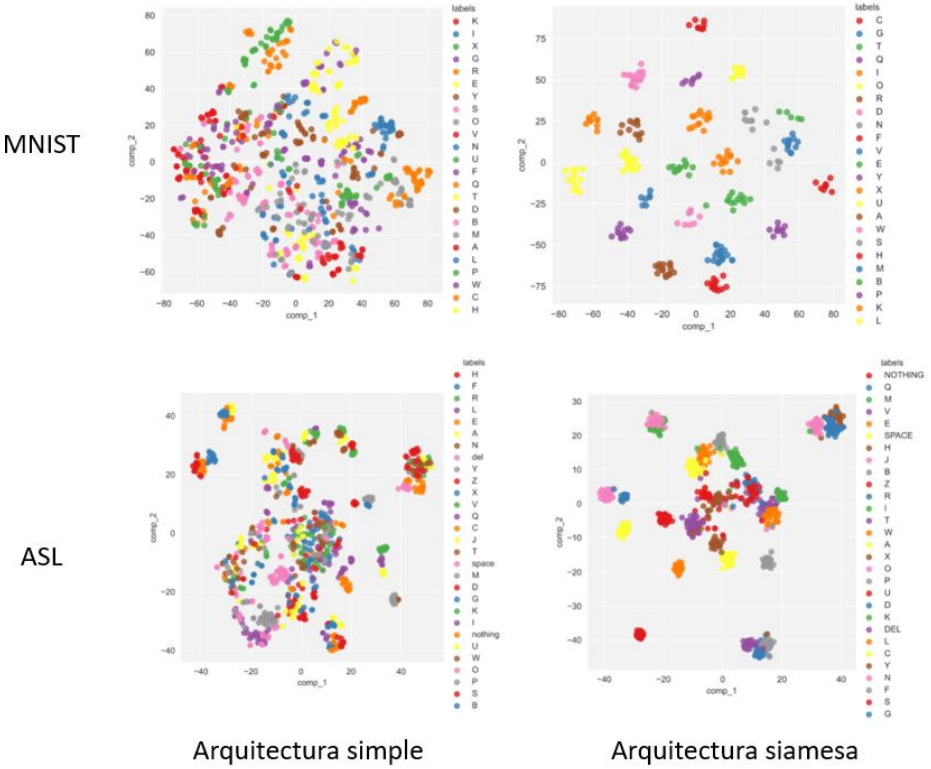


Fig. 5. TSNE.

como tamaño de imagen de entrada para para no perder información importante a través de las capas convolucionales. Los conjuntos de entrenamiento y validación están compuestos por 24,720 y 2,735 imágenes. La versión siamesa de VGG16 entrenada en el conjunto de datos MNIST Sing Language obtuvo un valor de exactitud de 0.9861 y un valor de 0.0199 de error, utilizando el set de entrenamiento. Utilizando el conjunto de validación se obtuvo una exactitud de 0.9834 y un error de 0.0237.

7.5. Mobilenet siamesa

Se utilizaron dos redes Mobilenet idénticas; la última capa contiene 1000 neuronas. Debido a limitaciones de hardware, las imágenes se redimensionaron a 64×64 para ambos conjuntos de datos; los hiperparámetros son básicamente los mismos que se usaron para la VGG16 siamesa, excepto por la cantidad de épocas.

Para el conjunto de datos de lengua de señas de MNIST, el conjunto de entrenamiento y validación estuvo compuesto por 24,709 y 2,746 imágenes, respectivamente. Los resultados de utilizando el conjunto de entrenamiento son los siguientes: error de 4.42E-4 y exactitud de 1.0. Con el conjunto de validación se obtuvo un error de 0.0061 y una exactitud de 1.0.

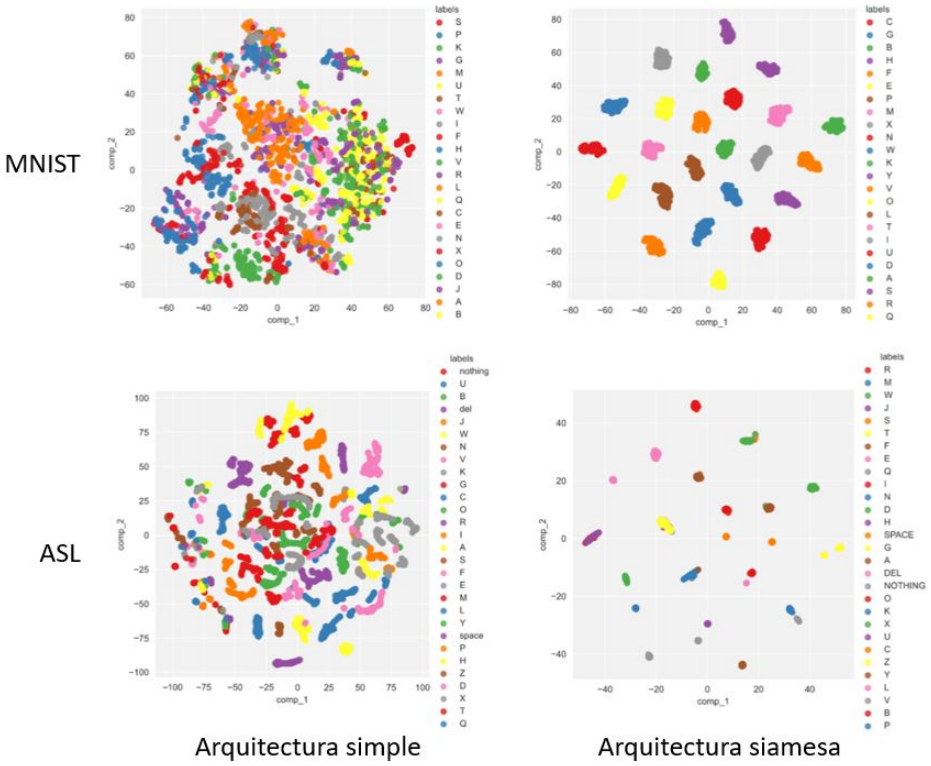


Fig. 6. TSNE.

Por otro lado, para el conjunto de datos de ASL Alphabet, solo se usó el 10 % del conjunto de datos debido a limitaciones de hardware; el conjunto de entrenamiento estuvo compuesto por 7,830 imágenes mientras que el conjunto de validación de 870 imágenes. Los resultados utilizando el conjunto de entrenamiento son los siguientes: error de 0.0064 y exactitud de 0.9925. Por otro lado, utilizando el conjunto de validación, se obtuvo un error de 0.0102 y una exactitud de 0.9888.

8. Resultados experimentales

El reconocimiento del alfabeto ASL se realiza mediante una tarea de clasificación, y para ello se utilizaron los modelos generados del entrenamiento de las redes mencionadas anteriormente. Para los modelos VGG16 y Mobilenet, la tarea de clasificación tiene como base alimentar una imagen del conjunto de prueba (muestras nunca vistas por la red) y dejar que la red prediga el alfabeto.

Las predicciones se compararon con los ejemplos reales. El rendimiento de la clasificación se midió utilizando las métricas más comúnmente utilizadas para la evaluación de la clasificación, como la exactitud, la precisión, recall y puntaje F1. En las Tablas 1 y 2 se muestran los resultados cuantitativos por parte de las arquitecturas simples y siamesas utilizando las dos bases de datos anteriormente mencionadas.

Tabla 3. Comparación de trabajos.

Referencia	**Trabajo**	**Conjunto** de **datos**	**Exactitud**
García et al. [8]	GoogleNet	ASL Alphabet	0.70
Hao et al. [10]	Autodestilación	ASL Alphabet	0.80
Método propuesto	VGG16 Siamesa	ASL Alphabet	0.89
Método propuesto	Mobilenet Siamesa	ASL Alphabet	0.91
LeCun et al. [19]	LeNet	MNIST	0.89
Krizhevsky et al. [16]	Alexnet	MNIST	0.94
Kaiming et al. [11]	ResNet18	MNIST	0.98
Kaiming et al. [11]	ResNet50	MNIST	0.98
Método propuesto	VGG16 Siamesa	MNIST	0.99
Método propuesto	Mobilenet Siamesa	MNIST	1.00

Se puede observar que para el caso de las arquitecturas siamesas, el resultado de la clasificación es mejor que el obtenido utilizando las arquitecturas simples, esto es debido a que el aprendizaje de similutd reduce la alta similitud entre clases y la alta variación dentro de la misma clase.

Una manera de comprobar esto es haciendo uso de la técnica t-SNE (t-distributed Stochastic Neighbor Embedding, en inglés) la cual es un algoritmo el cual permite visualizar datos multidimensionales. En la Figura 5 se presentan los resultados de t-SNE sobre la red VGG16 en su versión simple (primer columna) y su versión siamesa (columna 2) En cada renglón se presenta el resultado para cada uno de los conjuntos de datos.

Se puede observar que la arquitectura siamesa genera descriptores visuales de acuerdo a la similitud entre las imágenes, teniendo como resultado una disminución en la similitud entre clases (los grupos de cada letra estan más separados) y la reducción de la variación dentro de la misma clase (los grupos de descriptores visuales de una misma clase están más cerca).

En la Figura 6 podemos observar algo similar, los descriptores visuales que representan imágenes de la misma letra del alfabeto de la lengua de señas aparecen más cerca (se redujo la variación intra clase) y los descriptores visuales de diferentes letras aparacen separados (se redujo la similitud entre clase).

Para la predicción de una letra del alfabeto de la lengua de señas, se calculó el centroide de los descriptores visuales de las imágenes de entrenamiento de cada clase los cuales fueron generados por las redes siamesas. Para la predicción de un alfabeto de la lengua de señas se introdujo una imagen de prueba a la red siamesa, se obtuvo su vector característico y se calculo la distancia euclidiana con cada uno de los centroides de cada clase.

El centroide que esté mas cerca del descriptor visual de la imagen de prueba determinará qué seña el usuario está haciendo. El orden de complejidad de nuestro algoritmo para reconocer una letra nueva es de O(n^2). El sistema propuesto se evaluó con trabajos publicados en la literatura. En la Tabla 3 se muestra la comparación utilizando el conjunto de entrenamiento ASL Alphabet.

9. Conclusiones

La lengua de señas es la forma en que las personas con discapacidades auditivas y del habla se comunican con los demás, sin embargo, la mayoría de las veces, las personas oyentes no conocen esta lengua. Por lo tanto, existe una brecha de comunicación que impacta negativamente a la comunidad sorda. Por lo tanto, en este artículo, se presenta un método para el reconocimiento del alfabeto ASL.

Uno de los mayores desafíos en el reconocimiento del alfabeto ASL es la gran variación intraclase y la gran similitud entre las imágenes. Los métodos se enfocan en encontrar patrones para la clasificación alfabética de ASL. Para solucionar esto, se planteó la hipótesis de que, en caso de que sea posible generar patrones de acuerdo a la similitud de las imágenes, se podría mejorar el reconocimiento del alfabeto ASL.

Para ello, se implementó un aprendizaje de similitud semántica utilizando redes siamesas, que en pocas palabras, son dos redes idénticas que comparten sus parámetros. Los experimentos muestran que los descriptores visuales generados por las arquitecturas siamesas representan mejor a las imágenes al tener en cuenta las similitudes y diferencias entre ellas.

Otro hallazgo en los experimentos fue que a pesar de que el conjunto de entrenamiento para la CNN siamesa fue mucho menor en comparación con el conjunto de entrenamiento utilizado en una CNN simple, las redes siamesas no presentaron sobreajuste, esto se debe a la ayuda del aprendizaje one-shot. El aprendizaje one-shot permite que la red aprenda de solo unas pocas imágenes por clase.

Además, las redes siamesas son resistentes al desequilibrio de clases debido a que, al final del día, la red intenta aprender solo dos clases, pares de imágenes similares y no similares. El tiempo de entrenamiento y los requerimientos de memoria de hardware de las redes siamesas son los mayores inconvenientes porque involucra pares cuadráticos de muestras para aprender.

Referencias

1. Ameen, S., Vadera, S.: A convolutional neural network to classify American Sign Language fingerspelling from depth and colour images. Expert Systems, vol. 34, no. 3, pp. e12197 (2017) doi: 10.1111/exsy.12197
2. Assaleh, K., Shanableh, T., Zourob, M.: Low complexity classification system for glove-based arabic sign language recognition. Neural Information Processing, Springer Berlin Heidelberg, pp. 262–268 (2012) doi: 10.1007/978-3-642-34487-9_32
3. Aly, W., Aly, S., Almotairi, S.: User-independent american sign language alphabet recognition based on depth image and PCANet features, vol. 7, pp. 123138–123150 (2019) doi: 10.1109/access.2019.2938829
4. Baker, S.: The importance of fingerspelling for reading. Visual Language and Visual Learning Science of Learning Center (2010)
5. Battison, R.: American sign language: Lexical borrowing in american sign language. University of California (1978)
6. Dong, C., Leu, M. C., Yin, Z.: American sign language alphabet recognition using Microsoft Kinect. In: IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 44–52 (2015) doi: 10.1109/cvprw.2015.7301347

7. Deng, J., Dong, W., Socher, R., Li, L. J., Kai, L., Fei-Fei, L.: ImageNet: A large-scale hierarchical image database. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 248–255 (2009) doi: 10.1109/cvpr.2009.5206848
8. Garcia, B., Viesca, S. A.: Real-time american sign language recognition with convolutional neural networks. Convolutional Neural Network and Visual Recognition, vol. 2, pp. 225–232 (2016)
9. Geddes, K. O., Czapor, S. R., Labahn, G.: Algorithms for Computer Algebra. Springer US (1992) doi: 10.1007/b102438
10. Hao, A., Min, Y., Chen, X.: Self-mutual distillation learning for continuous sign language recognition. In: IEEE/CVF International Conference on Computer Vision, pp. 11303–11312 (2021) doi: 10.1109/iccv48922.2021.01111
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition, pp. 770–778 (2015) doi: 10.48550/ARXIV.1512.03385
12. Holden, E. J., Lee, G., Owens, R.: Australian sign language recognition. Machine Vision and Applications, vol. 16, no. 5, pp. 312–320 (2005) doi: 10.1007/s00138-005-0003-1
13. Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: MobileNets: Efficient convolutional neural networks for mobile vision applications (2017) doi: 10.48550/ARXIV.1704.04861
14. Kim, J. S., Jang, W., Bien, Z.: A dynamic gesture recognition system for the Korean sign language (KSL). IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics), vol. 26, no. 2, pp. 354–359 (1996) doi: 10.1109/3477.485888
15. Kaggle: Dataset homepage (2023) www.kaggle.com/datasets/grassknoted/asl-alphabet
16. Krizhevsky, A., Sutskever, I., Hinton, G. E.: ImageNet classification with deep convolutional neural networks. Communications of the ACM, vol. 60, no. 6, pp. 84–90 (2017) doi: 10.1145/3065386
17. Kumar, P., Gauba, H., Pratim Roy, P., Prosad Dogra, D.: A multimodal framework for sensor based sign language recognition. Neurocomputing, vol. 259, pp. 21–38 (2017) doi: 10.1016/j.neucom.2016.08.132
18. Kuznetsova, A., Leal-Taixe, L., Rosenhahn, B.: Real-time sign language recognition using a consumer depth camera. In: IEEE International Conference on Computer Vision Workshops (2013) doi: 10.1109/iccvw.2013.18
19. Lecun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. In: Proceedings of the IEEE, vol. 86, no. 11, pp. 2278–2324 (1998) doi: 10.1109/5.726791
20. Maqueda, A. I., del-Blanco, C. R., Jaureguizar, F., García, N.: Human–computer interaction based on visual hand-gesture recognition using volumetric spatiograms of local binary patterns. Computer Vision and Image Understanding, vol. 141, pp. 126–137 (2015) doi: 10.1016/j.cviu.2015.07.009
21. Nai, W., Liu, Y., Rempel, D., Wang, Y.: Fast hand posture classification using depth features extracted from random line segments. Pattern Recognition, vol. 65, pp. 1–10 (2017) doi: 10.1016/j.patcog.2016.11.022
22. Nanni, L., Ghidoni, S., Brahnam, S.: Handcrafted vs. non-handcrafted features for computer vision classification. Pattern Recognition, vol. 71, pp. 158–172 (2017) doi: 10.1016/j.patcog.2017.05.025
23. Oz, C., Leu, M. C.: Linguistic properties based on american sign language isolated word recognition with artificial neural networks using a sensory glove and motion tracker. Neurocomputing, vol. 70, no. 16–18, pp. 2891–2901, Oct. 2007. doi: 10.1016 /j.neucom.2006.04.016
24. Oz, C., Leu, M. C.: Recognition of finger spelling of american sign language with artificial neural network using position/orientation sensors and data glove. Advances in Neural Networks, pp. 157–164 (2005) doi: 10.1007/11427445_25

25. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014) doi: 10.48550/ARXIV.1409.1556
26. Tao, W., Leu, M. C., Yin, Z.: American Sign Language alphabet recognition using convolutional neural networks with multiview augmentation and inference fusion. Engineering Applications of Artificial Intelligence, vol. 76, pp. 202–213 (2018) doi: 10.1016/j.engappai.2018.09.006
27. Wang, C., Liu, Z., Chan, S. C.: Superpixel-based hand gesture recognition with kinect depth camera. In: IEEE Transactions on Multimedia, vol. 17, no. 1, pp. 29–39 (2015) doi: 10.1109/tmm.2014.2374357

ISSN 1870-4069

Implementación de algoritmo de pruebas para la detección de comportamientos humanos utilizando el filtro de Kalman para inferencia de actividades mediante el uso de machine learning

Braulio Israel Alejo-Cerezo, Juan Pablo Pérez-Monje,
Selene Ramírez-Rosales, Alvaro Anzueto-Ríos,
Jorge Luis Pérez-Ramos

Universidad Autónoma de Querétaro,
Facultad de Informática,
México

{bralioiac, juanpabloperez0299, seleneramirezrosales}@gmail.com,
aanzuetor@ipn.mx, jorge.luis.perez@uaq.edu.mx

Resumen. Los sistemas de videovigilancia son herramientas tecnológicas que ayudan al ser humano a la monitorización de áreas de interés para fines de seguridad, supervisión y prevención de delitos. Estudios indican que los sistemas de videovigilancia no supervisados exhiben una ventaja al no poseer limitaciones humanas, como parte de ello, el objetivo principal de este trabajo recae en la implementación de un sistema de videovigilancia no supervisada, enfocado en la detección de comportamientos humanos asociados con actos delictivos, utilizando algoritmos de identificación y seguimiento de personas en apoyo con el filtro de Kalman, implementando finalmente una respuesta de alarma y notificación al usuario. Con ello, se obtuvieron los resultados del análisis de vídeos cuyas escenas fueron capturadas en diferentes entornos, validando así la efectividad del sistema para la búsqueda e identificación de comportamientos.

Palabras clave: Videovigilancia, patrones, comportamientos, detección, seguimiento, vídeos.

Implementation of Test Algorithm for Human Behaviors Detection Using the Kalman Filter for Activities Inference Using Machine Learning

Abstract. Surveillance systems are technological tools that help humans in monitoring regions of interest for security, supervision, and crime prevention tasks. Researchers say unsupervised surveillance systems are advantageous since they do not depend on human limitations. Therefore, the main objective of this document is the implementation of an unsupervised surveillance system focused on the detection of human behaviors related to criminal acts, all of this by using algorithms for people detection and tracking, and finally, implementing an alarm

response and user notification system. This system lets us analyze videos in which scenes were recorded with different characteristics, validating the system's effectiveness in searching and identifying behaviors.

Keywords: Surveillance, patterns, behaviours, detection, tracing, videos.

1. Introducción

Desde su aparición, los sistemas de videovigilancia han conformado un conjunto de herramientas esenciales para el resguardo de espacios públicos y privados, tal como lo son centros comerciales, bancos, instituciones gubernamentales, escuelas, negocios, casas, etc. En la actualidad estos sistemas han evolucionado con herramientas propias de Inteligencia Artificial, principalmente enfocadas a la reducción de las principales fuentes humanas de inefectividad en relación con las cámaras de seguridad [2].

Un sistema típico de videovigilancia consta de cinco partes: detección de objetos, clasificación de objetos, seguimiento de objetos, entendimiento y descripción de comportamientos, y la identificación [7]. Las cámaras empleadas en videovigilancia graban días enteros de actividades, lo cual resulta en una gran cantidad de datos de vídeo que hacen de los procesos de búsqueda una tarea laboriosa y altamente demandante para un observador humano [8], cuando pretende obtener información relevante.

Estos sistemas cuentan con una persona encargada de evaluar los movimientos registrados en cada escena, de acuerdo a sus conocimientos y criterio de cada situación. Sin embargo, analizar múltiples cámaras de manera simultánea como sistemas distribuidos aumenta su complejidad y genera dependencia a un operador en la detección de actos delictivos, ya que es necesario monitorizar múltiples pantallas de forma simultánea y sincronizada, prestando atención a los detalles de cada escena de forma continua.

Lo anterior presenta limitaciones en la efectividad de la monitorización debido al error humano, como ceguera no intencional a causa de errores por falta de atención [1, 9]. Se reporta que en México, la tasa de incidencia delictiva ha presentado un crecimiento en el delito de robo a casa habitación (rch), esto de acuerdo con cifras de la Encuesta Nacional de Victimización y Percepción sobre Seguridad Pública (ENVIPE).

Actualmente, esta cifra apunta a 1,880 delitos por rch$\times$ 100,000 hab, no obstante, la [6] reporta que, debido a la contingencia sanitaria generada por el virus SARS-CoV2 (causante de la COVID-19), a partir del levantamiento de la ENVIPE 2020, con año de referencia 2019, se han registrado cambios estadísticamente significativos con respecto a los ejercicios anteriores, presentando valores más bajos de lo normal.

A pesar de ello, el número de personas que pueden verse beneficiadas mediante el presente proyecto se puede conocer gracias a que la ENVIPE 2021 consideró una tasa de 1.3 delitos por víctima durante el 2020, proporción que, de aplicarse al número total de delitos por robo a casa habitación (1.6836 millones de delitos), da como resultado un total de 1.295 millones de víctimas.

Este mismo estudio permite estimar el costo total a consecuencia de la inseguridad por delito, y estima que, sólo en el año de 2020, el delito en hogares alcanzó un monto

de pérdidas aproximado de 277.6 mil mdp. Para ilustrar la gravedad de esta cifra, se estima que la misma representa 1.85 % del PIB [6].

El beneficio de proyectos proyectos enfocados a sistemas de vigilancia se fundamenta en la investigación realizada por [3], donde establece que la tecnología es una alternativa efectiva si se utiliza como técnica de prevención situacional al delito, esto quiere decir que los dispositivos tecnológicos implementados en sistemas de vigilancia influyen directamente en los riesgos que perciben los infractores, y por ello, inhibe la comisión del delito. Por lo anterior, se propone el desarrollo de un sistema de vigilancia para la detección de comportamientos humanos utilizando el filtro de Kalman para inferencia de actividades mediante el uso de Machine Learning.

2. Marco teórico

Esta sección explica los fundamentos teóricos en los que se basa la presente investigación. En primer lugar se tiene el Procesamiento Digital de Imágenes, que es el proceso por el cual una máquina puede interpretar la información de los elementos que conforman una imagen, y es ampliamente usado en campos como la medicina, física, arqueología, etc. Por otro lado, el filtro de Kalman es un predictor lineal, el cual es empleado en sistemas no lineales para obtener la dinámica del sistema y el movimiento de los objetos en la escena.

2.1. Procesamiento digital de imágenes

Una imagen digital puede definirse como una función de dos dimensiones $f(x, y)$ donde x y y son coordenadas espaciales de un plano, y la amplitud de f en cualquier par de coordenadas es llamado intensidad o nivel de gris de la imagen en ese punto [5], sin embargo, existen también representaciones de imágenes en donde cada píxel de color es una combinación de los colores (también llamados canales) rojo, verde y azul [11]. Si una coordenada (x,y), así como el valor de f son cantidades finitas y discretas, la función toma el nombre de imagen digital.

El procesamiento de imágenes hace referencia al procesamiento de imágenes digitales empleando equipos de cómputo y toma en consideración las técnicas utilizadas para desarrollar dicho procesamiento, cuyo fin será obtener una mejora en la imagen o extraer información que resulte útil para el propósito que se persigue [12]. Una imagen digital se compone de un número finito de elementos, cada uno con un valor y ubicación particular, estos elementos son llamados elementos de imagen o píxeles.

2.2. Filtro de Kalman

El filtro de Kalman es un conjunto de ecuaciones matemáticas que son capaces de proveer un método computacional recursivo eficiente para estimar el estado de un proceso, minimizando el error cuadrático medio [14, 10].

El filtro de Kalman aborda el problema de estimar el estado $x \in \mathbb{R}^n$ de un proceso discreto controlado que está definido por la siguiente ecuación diferencial estocástica [13]:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}. \tag{1}$$

En un tiempo k, con una medición $z \in \mathbb{R}^m$:

$$z_k = Hx_k + v_k, \tag{2}$$

donde w_k y v_k son variables aleatorias que representan el ruido del proceso y la medición del ruido respectivamente, y se asume que son independientes una de la otra, con distribución normal de probabilidad:

$$\begin{aligned} p(w) &\sim N(0, Q), \\ p(v) &\sim N(0, R), \end{aligned} \tag{3}$$

donde Q es covarianza del ruido del proceso y R la covarianza del ruido de la medición, las cuales se asumen constantes. La matriz A con dimensión $n \times n$ relaciona el estado x en el tiempo $k-1$ con el estado x en el tiempo actual k. La matriz B de dimensión $n \times l$ relaciona una entrada opcional $u \in \mathbb{R}^l$ al estado x.

Por último, la matriz H $m \times n$ en la medición relaciona el estado con la medición z_k. Empleando las ecuaciones anteriores, así como mediciones reales $\hat{z}_k$ en cada tiempo k, el filtro de Kalman se usa para estimar de forma recursiva la media $\hat{x}_k$ y el error de covarianza P_k. El filtro se aplica en dos pasos: actualización del tiempo:

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_{k-1}, \tag{4}$$

$$P_k = AP_{k-1}A^T + Q, \tag{5}$$

Y actualización de la medición:

$$K = p_k^- H^T \left(HP_k^- H^T + R\right)^{-1}, \tag{6}$$

$$\hat{x}_k = \hat{x}_k^- + K\left(\hat{x}_k - H\hat{x}_k^-\right). \tag{7}$$

Cabe mencionar que el filtro de Kalman es óptimo ya que la matriz de ganancia del filtro K minimiza el error de la covarianza en el seguimiento de la iteración siguiente.

3. Implementación de la detección de comportamientos

La propuesta de este proyecto se aborda mediante la Figura 1, contando con cuatro fases correspondientes a: 1) el sensor de entrada de datos, 2) la multimedia extraída, 3) el sistema de procesamiento de datos y 4) la salida del sistema. Con ello, se planteó la programación de las funciones necesarias para identificar a cada persona dentro de las escenas y establecer el seguimiento de las mismas, evitando el conflicto de detección de sujetos mencionado por [4] mediante la implementación del filtro de Kalman.

Lo anterior debido a su mayor afinidad con el problema abordado en comparación con los equivalentes filtros Bayesianos, ya que estos se enfocan principalmente en estimar una función de densidad probabilística de los estados observados en el tiempo,

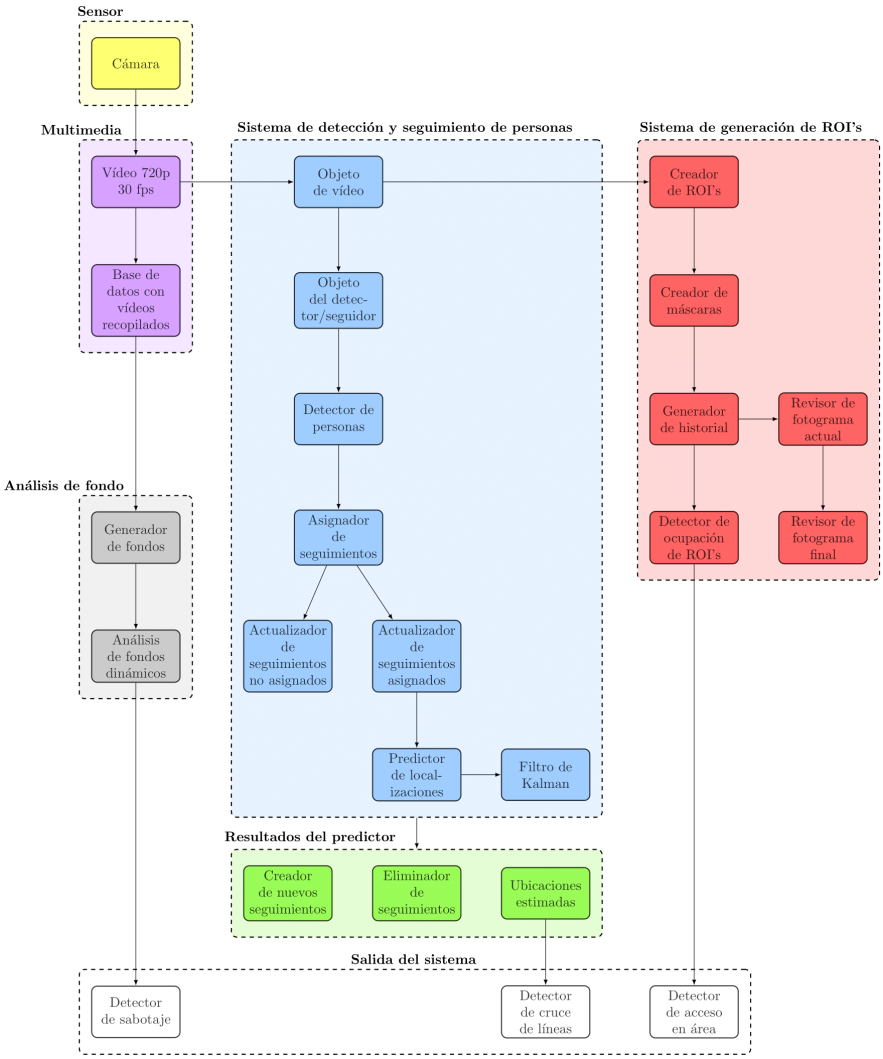


Fig. 1. Arquitectura general del sistema.

por ello, se encuentran generalmente dirigidos a estimar estados en el campo de la robótica, en donde se pretende determinar el estado actual del sistema dado un cierto historial de entradas y observaciones.

Por el contrario, el filtro de Kalman, como un caso especial de los filtros de Bayes, se utilizó para realizar una suposición en un sistema dinámico con información incierta, y establecer, a partir de ésta, el estado siguiente de dicho sistema, almacenando únicamente el estado previo.

Esto permitió realizar el seguimiento del movimiento de los sujetos basando este mismo en el punto central de cada persona identificada, utilizando el centroide de cada seguimiento para predecir su estado siguiente. Una vez hecho esto, se realizó la

Escena A1: pasillo, sin obstáculos, con puerta única ubicada a la izquierda de la escena.

Escena A2: pasillo y jardín, obstáculos por árboles y arbustos, con dos puertas.

Escena A3: pasillo, obstáculo de castillo cilíndrico a la izquierda, sin puertas.

Fig. 2. Vista y descripción de las escenas utilizadas en el proceso de validación.

actualización de cada caja tomando el largo y ancho actual para también establecer su tamaño futuro, permitiendo así corregir adecuadamente las localizaciones estimadas.

Por último, en este proceso se otorgó un número de identificación específico a cada caja para identificar su trayectoria de paso. A partir de ello, se buscó automatizar el proceso de seguimiento y predicción de los desplazamientos de cada sujeto en la escena, para no depender de un operador humano que, de acuerdo con [8, 15], genera una baja efectividad en la prevención de crímenes tras el monitoreo constante por largos periodos de tiempo.

Lo anterior con el fin de reconocer los siguientes tres comportamientos humanos: con cruce de líneas, acceso a área definida por el usuario y sabotaje de cámaras. Validando su eficacia en pruebas que utilizaron una base de datos de vídeos captados de manera propia, los cuales se observan en la Figura 2.

De acuerdo con lo establecido anteriormente, como primer paso, se buscó implementar un sistema dedicado al reconocimiento y seguimiento de los comportamientos humanos relacionados con actos delictivos, mencionados previamente, basado en el número de fotogramas por segundo y la secuencia de aparición de cada sujeto en escena. Posteriormente se formuló un modelo de detección de vulnerabilidades, registrando el cruce de líneas arbitrarias en la escena, ingreso a zonas determinadas en la misma y la obstrucción del campo de visión de la cámara.

En lo que respecta a estas dos primeras fases, se utilizaron grabaciones de videovigilancia obtenidas mediante datasets públicos de uso académico, y, posteriormente, fueron obtenidas las grabaciones de videovigilancia correspondientes

Tabla 1. Tabla de descripción de vídeos obtenidos.

Escena	Procedencia	Duración
A1	Nikon D3500	30 min
A2	Poco X3	20 min
A3	FOSCAM FI9804W	120 min
A4	rawi_mages_pedestrians[1]	10 min
A5	Virat_s_000102[2]	20 min

a instituciones públicas empleando los instrumentos referidos en la Tabla 1, donde se concentró el total de escenas utilizadas para esta investigación.

En lo referente al algoritmo de detección de cruce de líneas, este se realizó mediante el trazado de objetos llamados regiones de interés (del inglés Regions of Interest o ROI's), los cuales cumplieron el papel de interpretar límites en la escena dibujados por el usuario utilizando la función drawline.

Este método de trazado de líneas arbitrarias otorgó al programa la versatilidad para utilizar dichas líneas de acuerdo con las necesidades de monitorización de la escena, pues pueden ser empleadas tanto como delimitadores en la misma, detección de ingreso o salida de personas, o incluso como guías de trayectorias para el comportamiento deseado en los individuos que ocupen la zona.

El trazado de la región correspondiente al área restringida fue realizado utilizando la misma metodología empleada en el algoritmo de líneas previamente descrito, con la diferencia de que, para este caso, la ROI fue dibujada mediante la función drawpolygon, la cual permitió establecer un conjunto de puntos indefinido hasta que el usuario finalmente decidiera cerrar el polígono en cuestión.

De igual manera, la implementación de ambos algoritmos en el objeto de vídeo permitió establecer un conjunto de máscaras, con el fin de realizar un seguimiento de la región y líneas establecidas por el usuario durante el análisis del vídeo.

Una vez definidas las máscaras individuales de cada objeto, estas fueron conjuntadas en una única máscara general, con el tamaño de la escena estudiada, para poder ser comparadas con las bounding boxes pertenecientes a las personas, en cada uno de los fotogramas siguientes.

La interacción entre bounding boxes y ROI's se generó a través de la función inROI, con la cual, todos los píxeles pertenecientes a la máscara fueron comparados con las coordenadas delimitantes de cada bounding box, dando como resultado la interpretación de cuáles sujetos, de todos aquellos identificados en la escena, se encontraban interactuando con el área o lineas trazadas por el usuario. Una vez que esta interacción fue identificada por el programa, se realizó un primer registro en un vector

[1] E. Gebhardt and M. Wolf, "CAMEL Dataset for Visual and Thermal Infrared Multiple Object Detection and Tracking," IEEEInternational Conference on Advanced Video and Signal-based Surveillance (AVSS), 2018.

[2] "A Large-scale Benchmark Dataset for Event Recognition in Surveillance Video" by Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, J.K. Aggarwal, Hyungtae Lee, Larry Davis, Eran Swears, Xiaoyang Wang, Qiang Ji, Kishore Reddy, Mubarak Shah, Carl Vondrick, Hamed Pirsiavash, Deva Ramanan, Jenny Yuen, Antonio Torralba, Bi Song, Anesco Fong, Amit Roy-Chowdhury, and Mita Desai, in Proceedings of IEEE Comptuer Vision and Pattern Recognition (CVPR), 2011.

Tabla 2. Tabla de descripción de vídeos obtenidos.

Nivel	Descripción de alerta
1	Alerta de notificación de evento de bajo riesgo
2	Alerta preventiva a un evento de alto riesgo
3	Alerta de notificación de evento de alto riesgo

temporal llamado historial, que se encargó de almacenar el comportamiento detectado el cual puede ser cruce de línea 1, cruce de línea 2 o acceso al área restringida, además de que almacenaba información adicional como el fotograma de inicio y fin.

Con lo anterior se generó un listado de los eventos detectados en el orden en que transcurren en cada una de las grabaciones, los cuales fueron interpretados de forma semántica para comunicar al usuario cada uno de los comportamientos detectados. El algoritmo correspondiente a la detección del comportamiento de sabotaje se implementó por medio de la generación de fondos dinámicos, los cuales representaron el promedio de los fotogramas en intervalos de cinco segundos.

Cada vez que se obtuvo un nuevo fondo, este fue comparado con el promedio de los anteriores mediante una correlación de los histogramas de cada uno, de esta manera, cuando se mantuvo un coeficiente de correlación por encima del 80 % o 90 %, el nuevo fondo fue promediado junto a los demás.

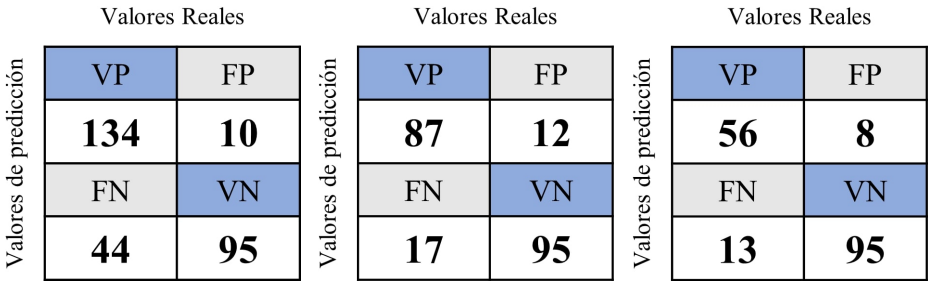
Por el contrario, cuando la correlación se encontró en un valor por debajo del 40 % o 30 % se optó por notificar al usuario acerca de sabotaje, ya que el algoritmo interpretó que un cambio radical en el último fondo obtenido pertenecía a una modificación de alto riesgo para la monitorización de la escena. Es importante mencionar que el umbral para evaluar el coeficiente de correlación fue ajustado a los valores que el usuario definió de acuerdo con el tipo y características de la escena estudiada.

Las salidas del algoritmo fueron validadas mediante un sistema de evaluación de umbrales, con el cual se establecieron opciones de respuesta de acuerdo a los parámetros de: tipo de comportamiento detectado, duración de comportamiento (para el caso de detección de acceso al área restringida), e incidencia de comportamiento (para el caso de la detección de cruce de líneas), con ello el sistema fue capaz de presentar una respuesta particular ante cada evento identificado, y de acuerdo con las características del mismo, obtuvo como salida la notificación al usuario y/o la activación de la alarma de seguridad, así como también la asignación de un nivel de alerta correspondiente a la situación encontrada con el fin de informar al usuario la gravedad del propio evento, los niveles de alerta son descritos en la Tabla 2.

Respecto al medio de notificación al usuario, se estableció un sistema de envío de correos electrónicos, para lo cual, se adaptó una plantilla con la estructura inicial de un email, considerando una sintaxis basada en el formato html para el envío de la información de interés del evento detectado empleando un socket de Outlook por medio de funciones propias de la plataforma.

4. Resultados

El análisis de cada una de las escenas descritas previamente se llevó a cabo por medio de matrices de confusión. La Figura 3 indica que la escena A1 cuenta con un



(a) Matriz de confusión para la escena A1.

(b) Matriz de confusión para la escena A2.

(c) Matriz de confusión para la escena A3.

Fig. 3. Resultados del reconocimiento de comportamientos en las escenas de validación.

Tabla 3. Tabla comparativa sobre las métricas de desempeño.

Fórmula	**A1**	**A2**	**A3**
$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}}$	0.7528	0.8365	0.8115
$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}}$	0.9305	0.8787	0.8750

total de 178 eventos, de los cuales 134 fueron identificados con éxito, 44 no pudieron ser identificados por el sistema y 10 se detectaron a pesar de no aparecer realmente en el vídeo. Por otra parte A2 registró un total de 104 eventos, con 87 detectados adecuadamente, 17 no reconocidos y 12 eventos detectados aunque inexistentes. Por último, A3 registró 69 eventos de los cuales detectó correctamente 56, pasando por alto únicamente 13 y reportando 8 eventos de más.

4.1. Desempeño del clasificador

Con los datos de las matrices de confusión mostrados en la figura previa se obtuvieron las métricas mostradas en la Tabla 3 de True Positive Rate y Positive Predictive Value. La información obtenida mediante las matrices de confusión y las métricas de desempeño, fue utilizada para generar un único gráfico de tabla ROC, con el cual comparar de manera visual la relación del TPR con el PPV.

La Figura 4 expone mediante la línea roja que la escena A1 posee un área bajo la curva correspondiente a la efectividad del 75.28 % en la detección de los eventos de interés (dado que la efectividad es mayor al 60 %, la escena revela un adecuado criterio de identificación de comportamientos), considerando igualmente que el número de eventos registrados es el mayor en comparación con los demás vídeos.

Por otro lado, la línea azul ilustra el hecho de que la escena A2 obtuvo el mayor porcentaje de efectividad encontrado en el sistema, con hasta un 83.65 % de éxito en el reconocimiento de los tres comportamientos. Finalmente, la escena A3 (línea verde), mostró un área bajo la curva relacionada con la efectividad del 81.15 %.

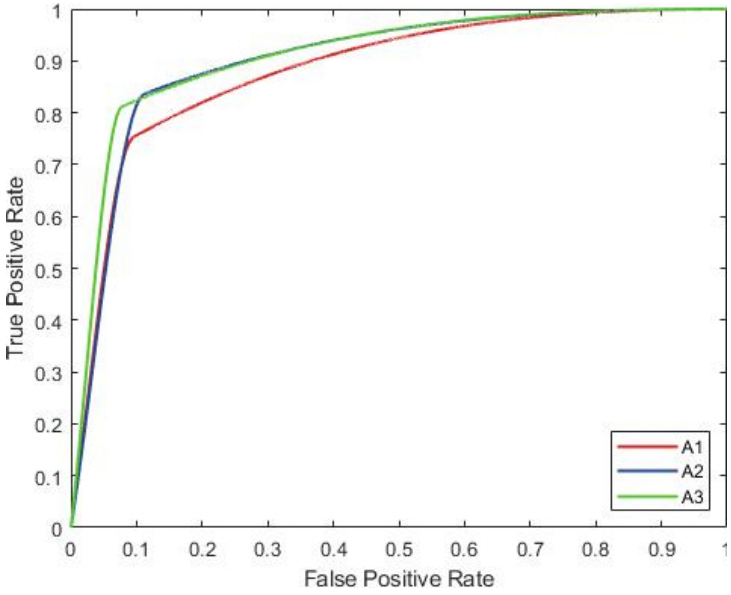


Fig. 4. Comparativa de las curvas ROC obtenidas en las escenas A1, A2 y A3.

5. Conclusiones

Una vez analizados los resultados obtenidos, se plantean las siguientes líneas de investigación que pueden ser abordadas a partir de este trabajo. En primer lugar, se propone la implementación de un algoritmo metaheurístico de optimización para la calibración autónoma del sistema, con el fin de encontrar los mejores parámetros del detector y seguidor de personas, así como umbral de correlación óptimo para la detección de sabotajes en cada escena.

En segundo lugar, se recomienda la búsqueda e implementación de nuevos comportamientos relacionados con actos delictivos, los cuales agreguen mayores opciones de identificación de eventos, tales como notificación por ausencia de sujetos, detección de dirección de flujo de personas y detección de merodeo.

Por otro lado, se propone realizar modificaciones al clasificador aumentando las características para generar un análisis de cada fotograma más allá de las interacciones de cajas de personas con ROI's y comparación de histogramas, mejorando la detección y el seguimiento de personas con la adición de datos biométricos almacenados en un sistema de perfiles, para clasificar a los sujetos de acuerdo con su potencial de amenaza en la escena.

En cuanto a las ROI's, se propone el desarrollo de un algoritmo de detección y generación de ROI's automático basado en el análisis de las regiones de mayor movimiento de la escena, con el fin de separarlas y delimitarlas mediante el algoritmo de segmentación de regiones Watershed, permitiendo establecer un estudio únicamente en las zonas de mayor actividad del vídeo.

Por último, se recomienda la adaptación de nuevos actuadores como parte de la respuesta física del prototipo, con el fin de permitirle tomar acciones específicas en

respuesta a los comportamientos detectados, como el encendido de luces inteligentes o el cierre de cerraduras automáticas, estableciendo un sistema adaptable en el área de la domótica.

Referencias

1. Bredemeier, K., Simons, D. J.: Working memory and inattentional blindness. Psychonomic Bulletin and Review, vol. 19, pp. 239–244 (2012) doi: 10.3758/s13423-011-0204-8
2. Chen, C., Surette, R., Shah, M.: Automated monitoring for security camera networks: Promise from computer vision labs. Security Journal, vol. 34, no. 3, pp. 389–409 (2020) doi: 10.1057/s41284-020-00230-w
3. Clarke, R. V.: Situational cime prevention: Successful case studies, Harrow and Heston, Publishers (1997)
4. Dan, X., Yan, Y., Ricci, E., Sebe, N.: Detecting anomalous events in videos by learning deep representations of appearance and motion. Comput Vis Image Underst, pp. 117–127 (2017) doi: 10.1016/j.cviu.2016.10.010
5. González, R. C., Woods, R. E.: Digital Image Proccessing, New York: Pearson (2018)
6. INEGI: Encuesta nacional de victimización y percepción sobre seguridad pública envipe 2021. (2021) https://www.inegi.org.mx/programas/envipe/2021/
7. Kong, L., Dai, R.: Object-detection-based video compression for wireless surveillance systems. IEEE MultiMedia, vol. 24, no. 2, pp. 76–85 (2017) doi: 10.1109/MMUL.2017.29
8. Meghdadi, A. H., Irani, P.: Interactive exploration of surveillance video through action shot summarization and trajectory visualization. In: IEEE Transactions on Visualization and Computer Graphics, vol. 19, pp. 2119–2128 (2013) doi: 10.1109/TVCG.2013.168
9. Sasse, M. A.: Not seeing the crime for the cameras? Communications of the ACM, vol. 53, no. 2, pp. 22–25 (2010) doi: 10.1145/1646353.1646363
10. Sorenson, H. W.: Least-squares estimation: from Gauss to Kalman. IEEE Spectrum, vol. 7, no. 7, pp. 63–68 (1970) doi: 10.1109/MSPEC.1970.5213471
11. Thanki, R. M., Kothari, A. M.: Digital Image Processing using SCILAB, Springer International Publishing (2019) doi: 10.1007/978-3-319-89533-8
12. Tyagi, V.: Understanding digital image processing, CRC Press (2018) doi: 10.1201/9781315123905
13. Welch, G., Bishop, G.: An introduction to the Kalman filter. (1995) perso.crans.org/club-krobot/doc/kalman.pdf
14. Welch, G. F.: Kalman filter. Computer Vision, Springer International Publishing, pp. 721–723 (2021) doi: 10.1007/978-3-030-63416-2_716
15. Wiliem, A., Madasu, V., Boles, W., Yarlagadda, P.: A suspicious behaviour detection using a context space model for smart surveillance systems. Computer Vision and Image Understanding, vol. 116, pp. 194–209 (2012) doi: 10.1016/j.cviu.2011.10.001

ISSN 1870-4069

Clasificación de ondas gravitacionales de supernovas usando redes neuronales convolucionales

Aldo A. Álvarez[1], Javier M. Antelis[2],
Claudia Moreno González[1]

[1] Universidad de Guadalajara,
Departamento de Física,
México

[2] Tecnologico de Monterrey,
Escuela de Ingeniería y Ciencias,
México

aldoa.alvarez@alumnos.udg.mx,
claudia.mgonzalez@academicos.udg.mx,
mauricio.antelis@tec.mx

Resumen. Las detecciones de Ondas Gravitacionales (OG) brindan información crucial para el estudio del comportamiento de diferentes fenómenos en el Universo. Hasta la fecha se han registrado más de cien detecciones de OG, pero todas ellas vienen de sistemas binarios de Agujeros Negros y Estrellas de Neutrones. Uno de los retos actuales en investigación de OG es encontrar señales provenientes de explosiones de colapso de núcleo de Supernovas (CCSNe). En este proyecto de investigación, se construye un clasificador de señales de OG usando Redes Neuronales Convolucionales. Se entrena el algoritmo con simulaciones de ondas generadas de forma computacional a partir de modelos 3D de eventos de CCSNe. El modelo final fue puesto a prueba para medir su nivel de precisión al clasificar nuevos datos aleatorios de señales de OG de CCSNe. Los resultados fueron satisfactorios para señales producidas por fuentes localizadas entre 0.10 y 0.58 Kpc de distancia del punto de detección.

Palabras clave: Redes neuronales convolucionales, deep learning, ondas gravitacionales.

Classification of Gravitational Wave Signals from Supernovae Using Convolutional Neural Networks

Abstract. Gravitational Wave (GW) detections provide crucial information for studying the behavior of different phenomena in the Universe. To this date, more than a hundred GW detections have been recorded, but all of them come from binary systems of Black Holes and Neutron Stars. One of the current challenges in GW research is to find signals coming from core-collapse Supernova explosions (CCSNe). In this research project, we built a GW signal classifier using Convolutional Neural Networks. We trained the algorithm with computationally

generated wave simulations from 3D models of CCSNe events.The final model was tested to measure its level of accuracy in classifying new random data of GW signals from CCSNe. The results were satisfactory for signals produced by sources located between 0.10 and 0.58 Kpc away from the detection point.

Keywords: Convolutional neural networks, deep learning, gravitational waves.

1. Introducción

De acuerdo con la teoría de Relatividad General (RG), las Ondas Gravitacionales (OG) son consideradas como fluctuaciones en el espacio-tiempo que pueden ser causadas por eventos cosmológicos extremadamente energéticos. Las OG inicialmente fueron predichas por Albert Einstein como una consecuencia de su teoría de RG en 1916 [1], pero fue hasta el año 2015 que la existencia de este fenómeno fue comprobada por medio de detecciones oficiales de una de estas señales cósmicas.

Desde entonces, muchas otras detecciones han sido registradas, lo cual ha permitido a los investigadores de esta rama de la Física seguir recopilando información sobre el comportamiento de varios cuerpos celestes que conforman el Universo. Las OG son perturbaciones débiles, y son tan pequeñas que solo eventos extremadamente energéticos pueden producir ondas que puedan ser medidas desde la Tierra con suficiente certeza.

Las OG detectables vienen de cuatro fuentes principales: sistemas binarios de estrellas, estrellas de neutrones giratorias, casos de colapso gravitacional y ondas producidas durante el Big Bang [2]. Este proyecto de investigación se limitó al análisis de OG de eventos de colapso gravitacional, específicamente de explosiones de Supernovas con Colapso de Núcleo (CCSNe).

Una de las mayores fuentes de datos de detección de OG son los interferómetros de LIGO (Laser Interferometry Gravitational-waves Observatory) [2]. Por esta razón, los métodos que se implementan en este proyecto de investigación fueron ajustados para trabajar con los estándares de procesamiento y análisis de señales propuestos por LIGO. La detección de OG ha sido un gran éxito para la ciencia, pero este campo aún está en sus primeras etapas.

Hasta la fecha, solo se han registrado detecciones confirmadas de OG que provienen de sistemas binarios de Agujeros Negros y Estrellas de Neutrones. La frontera de detección actualmente se encuentra en la búsqueda de OG generadas por CCSNe. En este proyecto se aborda este problema por medio de la implementación de herramientas de aprendizaje automático (Machine Learning). El objetivo principal de este trabajo es crear un clasificador de OG empleando técnicas de Machine Learning (ML).

Las OG de CCSNe son especialmente complicadas de detectar dado que se trata de eventos estocásticos que no se pueden predecir de manera exacta previo a la explosión. Por esta razón, existe un esfuerzo por hacer nuevos estudios sobre estos fenómenos para lograr comprender mejor su comportamiento, por ejemplo, por medio de simulaciones computacionales de este tipo de explosiones.

Las simulaciones son un elemento clave para el desarrollo de este proyecto, ya que son la fuente de datos que ayudarán a entrenar un algoritmo capaz de detectar un señal producida por una CCSNe dentro de una captura de datos de observación. Actualmente, se conoce la forma de las señales de OG de sistemas binarios, de modo que se tienen metodologías para encontrar este tipo de señales.

Sin embargo, estos procedimientos no pueden ser implementados para el caso de las señales de CCSNe, pues estas conllevan procesos estocásticos y la única manera de automatizar su detección es mediante búsquedas no-modeladas. La solución propuesta en este proyecto de investigación consiste en implementar algoritmos de ML que sean capaces de aprender patrones y rasgos generales de las simulaciones de OG de CCSNe para que en un futuro puedan utilizarse como una herramienta para encontrar señales dentro de datos reales de detección.

2. Ondas gravitacionales

Antes de alimentar un modelo de ML para hacer clasificaciones, es necesario entender la estructura y el significado de los datos que se desean analizar. Para este estudio, el clasificador trabajará con imágenes de señales de OG, de modo que el primer paso es entender la teoría básica que describe la generación de este tipo de ondas.

2.1. Relatividad general

La manera matemática de describir las OG es por medio de una solución linealizada de las ecuaciones de Einstein. En RG, las ecuaciones de Einstein relacionan la curvatura del espacio-tiempo con la distribución de masa contenida en este [1]. Estas ecuaciones se representan con el tensor de Einstein $G_{\mu\nu}$ (donde $\mu, \nu \in [0, 1, 2, 3] = [ct, x, y, z]$, y c es la velocidad de la luz en el vacío), el cual se define como:

$$G_{\mu\nu} = R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R \ , \tag{1}$$

donde $R_{\mu\nu}$ es el tensor de Riemann, R es el escalar de Ricci, y $g_{\mu\nu}$ es la métrica del espacio-tiempo [5]. Esta expresión matemática da origen a una serie de ecuaciones que, al ser resueltas para el espacio asintóticamente plano, se obtiene una solución con forma de onda que describe la propagación de las OG en el espacio-tiempo.

2.2. Explosiones de supernova por colapso de núcleo

Las estrellas tienen su ciclo de vida, nacen y mueren después de un tiempo. Aquellas estrellas masivas que contienen más de ocho masas solares mueren en un proceso violento en forma de una explosión de Supernova por colapso de núcleo (CCSNe). Las partículas contenidas en la estrella que está por morir se mueven de manera turbulenta debido a los efectos de la explosión.

Esto da como resultado un sistema de procesos estocásticos que son complicados de modelar. De modo que esta clase de fenómeno físico es de particular interés en diversas ramas de investigación dentro de la Cosmología y la Astronomía [6].

Las explosiones de Supernovas son eventos altamente energéticos que producen OG que pueden ser detectadas desde la Tierra con los observatorios actuales. Las OG de CCSNe se forman cuando el campo gravitacional de una estrella masiva hace que la materia contenida en ella colapse hacia su propio núcleo.

Como resultado de este proceso se liberan enormes cantidades de energía, lo cual genera perturbaciones en forma de ondas que se propagan en el espacio-tiempo [1]. Estas ondas pueden ser analizadas para hacer deducciones importantes sobre la dinámica que ocurre en la CCSNe. Se espera que las observaciones de OG de CCSNe contribuyan a responder muchas de las preguntas que aún se tienen sobre el funcionamiento de este tipo de fenómenos cósmicos.

3. Detección y procesamiento de señales de ondas gravitacionales

Las OG pueden ser representadas como oscilaciones en la curvatura del espacio-tiempo, las cuales acarrean información sobre los cambios en el campo gravitacional de un objeto en el Cosmos [7]. La información contenida en la OG puede ser analizada e interpretada con la ayuda de equipo especializado.

Durante los últimos años, diferentes grupos de investigación alrededor del mundo han hecho esfuerzos para crear nuevas técnicas que permitan detectar OG de manera más efectiva. Para este proyecto, se adaptó la metodología para trabajar con datos de detección de LIGO, que posee la mayor red de observatorios de OG hasta la fecha.

3.1. Interferómetros

La detección de OG de LIGO se hacen por medio de la implementación de un interferómetro Michelson modificado para funcionar a gran escala. Los interferómetros de LIGO se encuentran en las ciudades de Livingston, Louisiana (detector $L1$) y Hanford, Washington (detector $H1$), que además colaboran con el observatorio Virgo (detector $V1$) en Italia.

Esta red de interferómetros ayudan a que LIGO pueda corroborar con mayor certeza si una detección de OG es real o si se trata de una falsa alarma. De este modo, una OG detectada se puede confirmar si aparece en los tres observatorios en el tiempo correspondiente, además de que ayudan a seguir registrando datos aún cuando alguno de los interferómetros se encuentra fuera de servicio.

3.2. Procesamiento de señales

Una de las partes más importantes de la detección de OG es el pre-procesamiento de los datos de detección para que estos puedan ser interpretados. Los datos que se registran durante las corridas de detección están llenos de ruido que predomina sobre las señales de OG, de modo que no es posible observar directamente una señal de OG en una gráfica de datos sin pre-procesamiento. Los detectores de LIGO son capaces de registrar señales en un rango de frecuencias de 10 Hz - 7000 Hz [11].

Una señal de OG por lo general se encuentra en un orden de magnitud de 10^1 - 10^2 Hz, de modo que en cada detección existe un exceso de información que debe ser filtrada para poder observar la señal de OG.

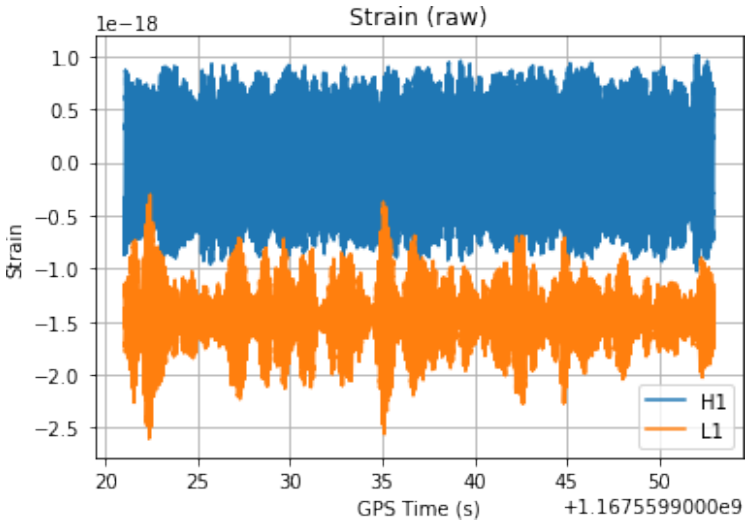


Fig. 1. Serie de tiempo de datos sin pre-procesar de una detección de LIGO hecha con los detectores H1 y L1 [10].

Además, LIGO genera datos en forma de series de tiempo que contienen registros de miles de segundos de observación. Esto añade una capa más de complejidad para las detecciones ya que una observación de OG tiene una duración de una fracción de segundo.

Por esta razón, existen grupos de investigación dedicados únicamente al desarrollo e implementación de algoritmos matemáticos que ayuden a filtrar las señales de OG que se esconden bajo todo el ruido de detección. Esta preparación de los datos de LIGO consiste en implementar técnicas como el análisis de densidad espectral de ruido (NSD), blanqueamiento de datos y transformaciones matemáticas, por nombrar algunas [12].

3.3. Datos de detección

Los datos de LIGO se registran en forma de series de tiempo, es decir, en sucesiones de puntos de medición tomados en un intervalo de tiempo y organizados de manera cronológica. Los datos de detección representan las variaciones en el patrón de interferencia del interferómetro a lo largo de un intervalo de tiempo específico.

Se muestra un ejemplo de los datos de observación de LIGO sin pre-procesar en la figura 1. Como se mencionó anteriormente, los datos de detección están repletos de ruido de diversas fuentes, por lo que es necesario pre-procesar las señales.

Generalmente, el ruido es demasiado denso comparado con las señales de OG, y normalmente es producto de fuentes como la actividad sísmica de la Tierra, vibraciones en el equipo de laboratorio, ruido térmico, perturbaciones eléctricas, entre muchas otras fuentes [10].

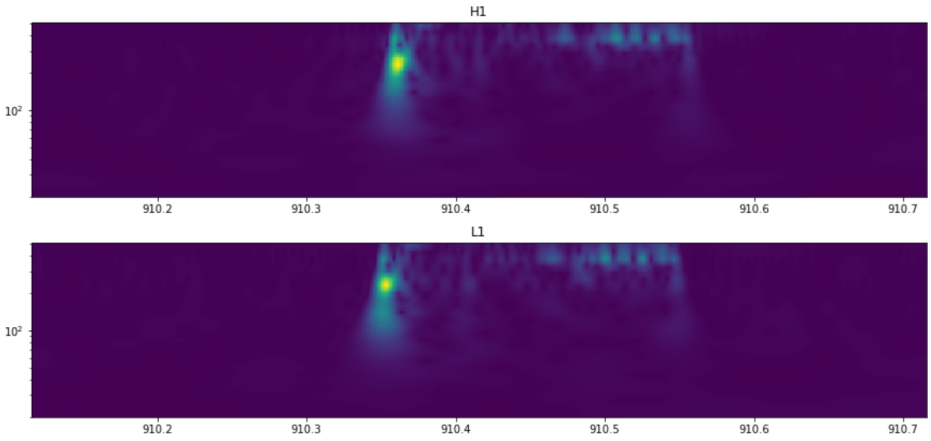


Fig. 2. Planos tiempo-frecuencia de una serie de tiempo de datos de detección de OG después de implementar la transformada Q [10].

3.4. Transformada Q

La transformada Q es una técnica matemática que tiene un uso muy importante en el análisis de datos de OG. Una manera eficaz de visualizar señales de OG es con una representación en el plano de tiempo-frecuencia que se obtiene a partir de la transformada Q. El resultado es similar a un espectrograma generado con transformadas de Fourier, con la ventaja adicional de que los intervalos de frecuencia se encuentran más espaciados para frecuencias mayores [10].

Esto da como resultado una mejor resolución para frecuencias altas, lo cual permite crear un espectrograma que muestra señales de OG de altas frecuencias con mayor claridad. En este sentido, la transformada Q proporciona una mejor representación para datos espectrales que la transformada de Fourier [17].

4. Metodología

Las OG producidas por CCSNe son un tema de estudio en la actualidad. Un ejemplo de estos esfuerzos por entender el comportamiento de dichos fenómenos estocásticos es el proyecto publicado por Scheideger [3], en el cual se generaron simulaciones computacionales tridimensionales de eventos de CCSNe. A partir de tales simulaciones se crearon plantillas que representan una señal de OG que cada explosión simulada hubiera producido en el espacio. Las plantillas de OG producidas por Scheideger son un recurso de libre acceso que aproximan el comportamiento de una CCSNe.

Estas simulaciones fueron el factor clave para el desarrollo de este proyecto de investigación, ya que se utilizaron para el proceso de entrenamiento del clasificador de OG. El proceso que se describe a continuación para la creación de un set de datos y su implementación para entrenar una Red Neuronal Convolucional (CNN) fueron basados en la metodología propuesta por Manuel Morales en su proyecto de clasificación de OG para sistemas binarios de agujeros negros [21].

Tabla 1. Distancias de simulación de las plantillas de OG seleccionadas para construir el set de datos de entrenamiento de la CNN.

	Plantillas de OG								
Distancia (Kpc)	10.00	5.62	3.16	1.78	1.00	0.58	0.32	0.18	0.10

4.1. Construcción del set de datos

El set de datos para entrenar la CNN necesita incluir imágenes de ejemplo de OG y ruido de detección correspondientes a las observaciones que produciría LIGO. Como se mencionó antes, hasta la fecha no se han logrado hacer detecciones de OG de CCSNe, por lo que no sería posible formar un set de datos con las detecciones disponibles en el repositorio de LIGO.

Por lo tanto, para crear el set de datos de entrenamiento se utilizaron las señales de OG de las simulaciones numéricas de Scheideger en conjunto con señales reales de ruido de detección de LIGO.

Los datos iniciales son series de tiempo que simulan detecciones de OG de LIGO producidas por eventos de CCSNe. Los datos se generan inyectando señales de CCSNe del repositorio de Scheideger dentro de detecciones de ruido de LIGO en formato de series de tiempo. Como parte de este procedimiento, se mantiene un registro de los intervalos de tiempo que contienen únicamente ruido o una combinación de ruido y señal de OG.

Los datos se dividen de acuerdo a las diferentes distancias de medición a las que fueron simulados, ya que la distancia es un factor que cambia la amplitud de la onda detectada. Los datos utilizados para este proyecto corresponden a las distancias que se muestran en el cuadro 1. Los conjunto de datos para cada distancia se dividen en cuatro sets. Cada set es una serie de tiempo que contiene 61 señales de OG distribuidas de manera uniforme sobre 1,200 segundos de datos de ruido de detección.

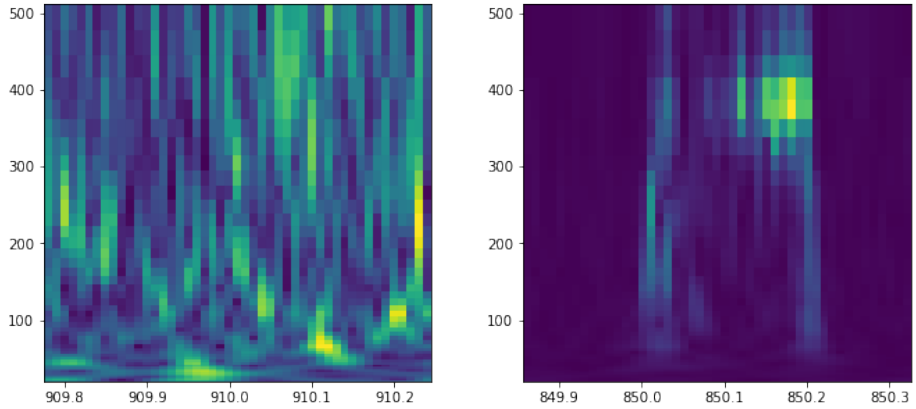
4.2. Pre-procesamiento de datos

Los datos iniciales están hechos para representar detecciones reales, de modo que deben ser pre-procesados según las recomendaciones de LIGO. Es necesario aplicar las técnicas de procesamiento de señales mencionadas en la sección 3.

La figura 2 muestran un ejemplo de cómo debería verse la señal después del pre-procesamiento. Esta figura muestra el espectrograma de la señal en el dominio de frecuencia, la cual se produce después de aplicar la transformada Q. En este estudio se utilizan las imágenes en el formato de espectrograma para entrenar la red neuronal.

Los espectrogramas son imágenes que muestran los datos a lo largo del tiempo, representados en términos de frecuencia y su densidad de frecuencia correspondiente. Se utilizaron datos de dos interferómetros de LIGO, H1 y L1, así que habrá una imagen del espectrograma por cada detector.

Como ajuste adicional, se redujo la resolución de las imágenes con el fin de disminuir los tiempos de procesamiento computacional. Cada imagen incluida en el set de datos es un array de 48×48 píxeles que representan 0.5 segundos de datos de detección. Un ejemplo de las imágenes que terminan guardándose en el set de datos se muestra en la figura 3.



(a) (b)

Fig. 3. Ejemplos de imágenes con baja resolución que se usaron en el set de datos. La imagen de la izquierda representa una señal de Ruido y la imagen de la derecha representa una señal de OG+Ruido.

Por cada inyección de una plantilla de OG se crearon tres imágenes de la señal: una imagen centrada en el medio de la señal, una imagen centrada en el extremo derecho de la señal y otra centrada en el extremo izquierdo de la señal.

Además, por cada grupo de tres imágenes de señal de OG+Ruido, se crearon tres imágenes de Ruido de detección con el fin de mantener un balance de las dos categorías dentro del set de datos de entrenamiento. Todas las imágenes van acompañadas de una etiqueta que indica la categoría correspondiente, ya sea Ruido u OG+Ruido.

4.3. Estructura del set de datos

El set de datos final incluye imágenes representadas como matrices de píxeles, y cada entrada está acompañada de su etiqueta categórica correspo 1, y cada set de datos se dividió en cuatro sub sets diferentes.

Esta subdivisión se hizo para tener mayor control sobre los sets de entrenamiento y evaluación. Cada set contiene 61 inyecciones de plantillas de OG, de manera que su conjunto correspondiente tiene 366 entradas de datos; 183 entradas de señal de OG y 183 entradas de Ruido. Cada factor de distancia contiene cuatro sets, lo cual da un total de 1,464 entradas de datos que se usan para entrenar y evaluar la CNN.

4.4. Estructura de la red neuronal convolucional

Se implementaron tres variaciones distintas en la arquitectura de la CNN, las cuales se usaron para procesar los mismos datos por separado y así obtener una comparación de su desempeño. La primera CNN se compone de una sola capa convolucional, la segunda tiene dos capas convolucionales y la tercera red contiene tres capas convolucionales. Las CNNs que se usaron en este proyecto de investigación mantienen una arquitectura simple.

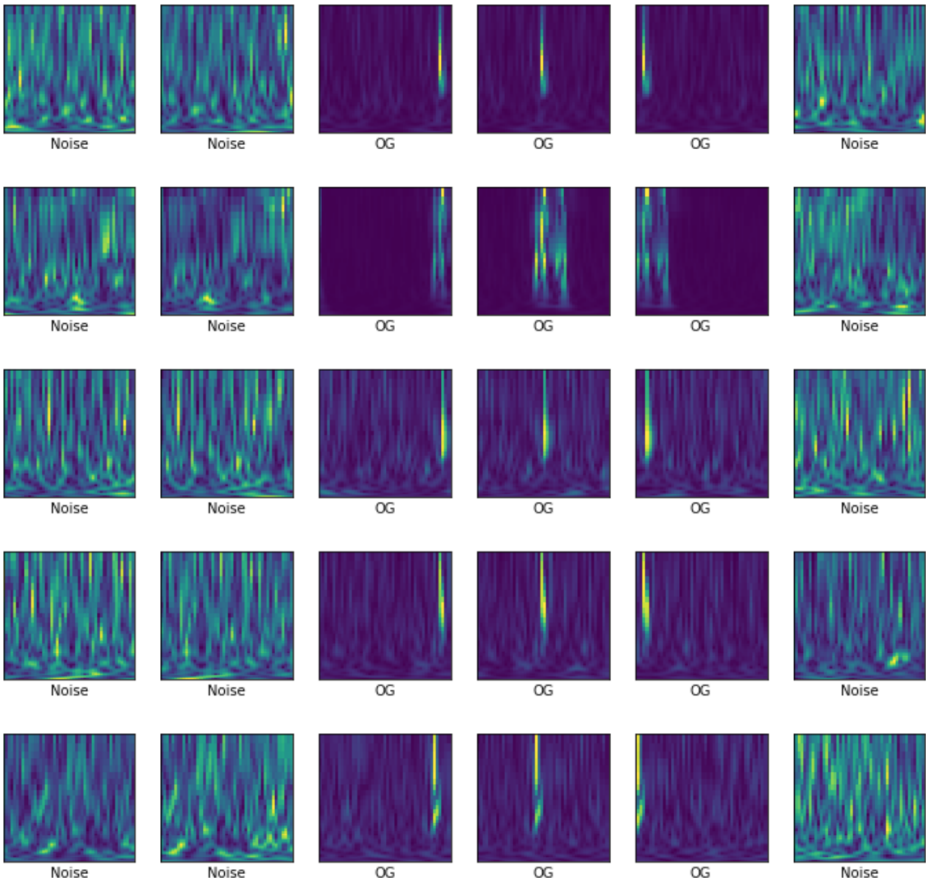


Fig. 4. Representación gráfica del set de datos final de imágenes de espectrogramas con su etiqueta categórica correspondiente.

Para el caso de la CNN de 3 capas, se construyó con dos bloques de capas convolucionales y de pooling, además de un tercer bloque con una capa convolucional y una de flattening para preparar los datos de salida. Al final de la estructura de la red se encuentran dos capas fully connected las cuales captan la salida de la red para asignarla a una de las dos categorías según corresponda la imagen que se analiza. En la figura 7 se muestra una descripción de la arquitectura del modelo. Para las otras dos variaciones de CNN se mantuvo la misma estructura, únicamente fueron ajustadas con una y dos capas convolucionales para satisfacer ambos casos.

4.5. Entrenamiento

Cada variación de CNN se entrenó por separado para cada uno de los factores de distancia del cuadro 1. Como se mencionó antes, a cada distancia le corresponde un grupo de datos divididos en cuatro sets, de modo que tres de estos sets se usaron para entrenar la CNN, y el último set se usó para evaluar el modelo.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 48, 48, 16)        304
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 24, 24, 16)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 22, 22, 32)        4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 11, 11, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 9, 9, 64)          18496
_________________________________________________________________
flatten (Flatten)            (None, 5184)              0
_________________________________________________________________
dense (Dense)                (None, 64)                331840
_________________________________________________________________
dense_1 (Dense)              (None, 2)                 130
=================================================================
Total params: 355,410
Trainable params: 355,410
Non-trainable params: 0
_________________________________________________________________
```

Fig. 5. Arquitectura de la CNN de tres capas utilizada en este proyecto de investigación.

Se hicieron todas las combinaciones posibles de los cuatro sets para dividirlos en entrenamiento y evaluación, de modo que cada CNN se entrenó cuatro veces para cada distancia. Al final se hizo un promedio de los resultados obtenidos de las cuatro iteraciones para calcular la exactitud general de los modelos según las distancias de detección. Además, en cada iteración de las CNNs se hicieron tres arreglos diferentes de los datos de entrada: el primero incluyendo sólo imágenes del detector L1, el segundo sólo con imágenes del detector H1, y el tercero con imágenes de ambos detectores L1+H1 simultáneamente.

5. Resultados

Los resultados de exactitud de clasificación de las CNNs fueron muy parecidos para el caso donde se entrenó con imágenes de una sola fuente, es decir, usando sólo imágenes de L1 o de H1. Para el caso donde se usaron juntas las matrices de píxeles de ambos detectores, la exactitud de clasificación aumentó entre $1\,\%$ y $4\,\%$ comparado con los casos previos. Este cambio de resultados no fue drástico, no obstante, se observó que la CNN fue más consistente en su porcentaje de exactitud durante los diferentes experimentos de entrenamiento usando imágenes de ambos detectores H1 y L1 juntos.

Este resultado era de esperarse, ya que cada entrada del set de entrenamiento contiene dos imágenes en lugar de sólo una, de modo que hay más información que la CNN puede usar para aprender patrones de clasificación. Después de entrenar la CNN, esta se puso a prueba con el set de datos de evaluación para cuantificar su exactitud.

Tabla 2. Desempeño promedio de la CNN para el conjunto de todas las distancias de observación de acuerdo a los datos de entrada y su número de capas convolucionales.

		Desempeño				
Capas	**Input**	**Acc**	**TN**	**FP**	**TP**	**FN**
3	**H1+L1**	86.38	81.58	8.79	91.21	18.42
	L1	84.54	80.68	11.57	88.43	19.32
	H1	84.17	80.68	12.29	87.71	19.32
2	**H1+L1**	85.68	81.27	9.89	90.11	18.73
	L1	83.75	80.52	12.99	87.01	19.48
	H1	84.34	80.59	11.85	88.15	19.41
1	**H1+L1**	84.61	80.79	11.51	88.49	19.21
	L1	83.74	78.47	12.06	87.94	21.53
	H1	83.14	80.36	14.09	85.91	19.64

Los resultados se dividen en cinco parámetros: Accuracy (Acc), True Positives (TP), False Positives (FP), True Negatives (TN), y False Negatives(FN). A continuación se enlista la descripción de lo que representa cada una de los parámetros:

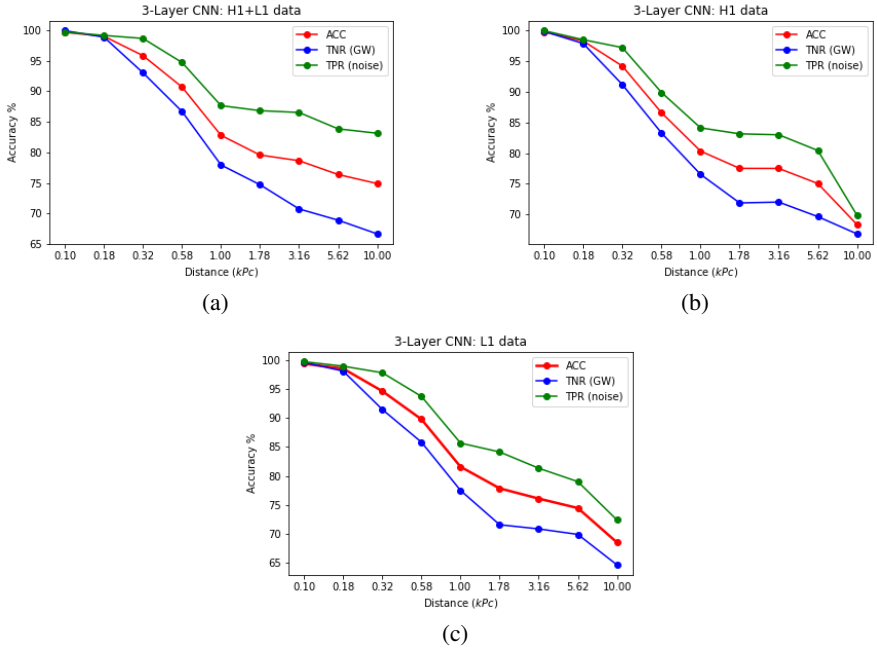
– Acc: porcentaje de aciertos de la CNN al clasificar los datos de evaluación.
– TN: porcentaje de OG clasificadas como OG.
– FP: porcentaje de OG clasificadas como Ruido.
– TP: porcentaje de Ruido clasificado como Ruido.
– FN: porcentaje de Ruido clasificado como OG.

El cuadro 2 muestra los resultados del desempeño en la clasificación por medio de las tres diferentes versiones de la CNN después de ser entrenadas con datos de los detectores H1, L1 y H1+L1. Cabe destacar que estos resultados corresponden al promedio del desempeño de las CNN usando los datos de las 9 distancias distintas que se consideraron para el estudio (cuadro 1).

La primera columna del cuadro 2 se muestra la cantidad de capas convolucionales de la CNN implementada, en la segunda columna se indica la fuente de datos de entrada, y las columnas restantes contienen los resultados de la evaluación de los modelos. También se puede observar el desempeño de la red con cada distancia individual.

La figura 6 muestra cómo cambia la exactitud de la CNN en promedio para cada distancia de observación de los datos. En estas gráficas se puede ver la tendencia del cambio en Acc, TNR y TPR de acuerdo a las diferentes distancias. Estas figuras son una representación de la evolución del aprendizaje de la CNN conforme se aumenta la distancia de observación de las simulaciones utilizadas, ilustrando por separado los casos donde se entrena con datos de H1, L1 y H1+L1.

En la figura 6 se muestran únicamente los resultados correspondientes al desempeño de la CNN de 3 capas convolucionales debido a que estos fueron considerados como los datos más representativos. En general, los casos de las CNNs con una y dos capas convolucionales no mostraron cambios significativos en la exactitud de clasificación, por lo que los resultados son bastantes similares. Se propuso que la CNN con 3 capas es el caso representativo porque, comparada con las otras dos variaciones, su disminución de exactitud es más consistente y menos pronunciada conforme aumenta la distancia.



(a) (b)

(c)

Fig. 6. Desempeño de la CNN de 3 capas de acuerdo a la distancia de detección de OGs. TPR representa el porcentaje de Ruido clasificado como Ruido, y TNR es el porcentaje de OG clasificadas como OG. Cada gráfica indica en el título si corresponde a la CNN entrenada con datos de H1, L1 o H1+L1.

6. Conclusiones

A pesar de que no se percibieron grandes diferencias en el desempeño de las CNN al variar su número de capas convolucionales, la CNN de tres capas fue la que mostró resultados de aprendizaje más consistentes comparada con las otras dos versiones. Por otro lado, los resultados indican que existe un notorio cambio en la exactitud del modelo de acuerdo a la distancia de observación de los datos.

Las clasificaciones para los datos a 10.00 Kpc de distancia tuvieron la exactitud general más baja del conjunto de datos con un Acc cerca de 70 %. El parámetro de Acc incrementa gradualmente para las distancias más cortas (0.18 y 0.10 Kpc), con un punto máximo que llega al 99 %.

Esto significa que las CNNs son considerablemente mejores para clasificar señales de OG en las distancias de detección más cortas (detecciones a 0.58 Kpc o menores). Estos resultados pueden deberse a que el ruido de detección tiende a incrementar su densidad con la distancia, haciendo que las señales de OG queden ocultas dentro de este.

Otro punto a destacar es que el porcentaje de TP es consistentemente mayor que el porcentaje de TN para todos los casos. Esto significa que las CNNs son mejores al clasificar correctamente el Ruido que las señales de OG. Este mismo resultado fue observado en el estudio de Morales (a partir del cual se basó la metodología de este trabajo) para el caso de OGs de sistemas binarios de agujeros negros [21].

Además, es bueno ver en los resultados que el porcentaje de FP en todos los casos es menor al porcentaje de FN, ya que esto indica que la CNN puede llegar a clasificar señales de Ruido como OG pero es menos propensa a confundir señales de OG con Ruido. Esto es relevante porque, si el algoritmo pre-entrenado clasifica señales de OG como Ruido, algunas señales de OG pasarían desapercibidas y no se podría confiar en la CNN como una herramienta de detección.

Los modelos de CNN proporcionaron resultados satisfactorios. Sin embargo, aún hay partes de la metodología implementada que pueden desarrollarse para obtener mejores resultados de clasificación. Se planea seguir trabajando en mejorar este clasificador de señales de OGs, pues se considera que podría llegar a convertirse en una poderosa herramienta de automatización para futuras detecciones de estos fenómenos físicos.

Referencias

1. McMahon D.: Relativity demystified. The McGraw-Hill Companies, Inc (2006) doi: 10.1036/0071455450
2. Schutz B.: A first course in general relativity. Cambridge University Press (2009) doi: 10.1017/9781108610865
3. Scheidegger, S., Käppeli, R., Whitehouse, S. C., Fischer, T., Liebendörfer, M.: The influence of model parameters on the prediction of gravitational wave signals from stellar core collapse. Astronomy and Astrophysics, vol. 514, pp. A51 (2010) doi: 10.1051/0004-6361/200913220
4. Fesik, L.: Polarization states of gravitational waves detected by LIGO-Virgo antennas (2017) doi: 10.48550/ARXIV.1706.09505
5. Szczepańczyk, M. J.: Multimessenger astronomy with gravitational waves from core-collapse supernovae. Ph. D. Thesis, Embry-Riddle Aeronautical University (2018)
6. Burrows, A., Vartanyan, D.: Core-collapse supernova explosion theory. Nature, vol. 589, no. 7840. Springer Science and Business Media LLC, pp. 29–39 (2021) doi: 10.1038/s41586-020-03059-w.
7. Moore, T. A.: A general relativity workbook. University science Books (2013)
8. Torres M. R.: Modos de polarización de ondas gravitacionales generadas por colapsos de núcleo de estrellas masivas. Tésis de Licenciatura, Universidad de Guadalajara, México (2021)
9. Villalvazo J. A.: Análisis del método de máxima verosimilitud en la detección de ondas gravitacionales emitidas por Supernovas tipo II. Tésis de Licenciatura, Universidad de Guadalajara, México (2020)
10. LIGO Virgo Collaboration. Gravitational-wave open data workshop #2. Physics Department, Paris Diderot University (2019)
11. Aasi, J., Abbott, B. P., Abbott, R., Abbott, T., Abernathy, M. R., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R. X., Adya, V., Affeldt, C., Aggarwal, N., Aguiar, O. D., Ain, A., Ajith, P., Alemic, A., Allen, B., Amariutei, D., Anderson, S. B.: Advanced LIGO. Classical and Quantum Gravity, vol. 32, no. 7, pp. 074001 (2015) doi: 10.1088/0264-9381/32/7/074001
12. Maggiore, M.: Gravitational waves. Oxford University PressOxford (2007) doi: 10.1093/acprof:oso/9780198570745.001.0001
13. Gravitational Wave Open Science Center: Event portal (2019) https://www.gw-openscience.org/eventapi/

14. Koivunen, A. C., Kostinski, A. B.: The feasibility of data whitening to improve performance of weather radar. Journal of Applied Meteorology and Climatology, vol. 38, no. 6, pp. 741–749 (1999) doi: 10.1175/1520-0450(1999)038⟨0741:TFODWT⟩2.0.CO;2
15. Huelsman, L. P.: Analog electrical filters. Encyclopedia of Physical Science and Technology, Elsevier, pp. 519–530 (2003) doi: 10.1016/b0-12-227410-5/00023-5
16. Abbott, B. P., Abbott, R., Abbott, T. D., Abraham, S., Acernese, F., Ackley, K., Adams, C., Adya, V. B., Affeldt, C., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O. D., Aiello, L., Ain, A., Ajith, P., Alford, T., Allen, G., Allocca, A., Aloy, M. A.: A guide to LIGO–Virgo detector noise and extraction of transient gravitational-wave signals. Classical and Quantum Gravity, vol. 37, no. 5, pp. 055002 (2020) doi: 10.1088/1361-6382/ab685e
17. Brown J. C.: Calculation of a constant Q spectral transform. Journal of the Acoustical Society of America, vol. 89, no. 1, pp. 425–434 (1991)
18. Géron A.: Hands-on machine learning with scikit-learn and tensorflow. O'Reilly Media (2017)
19. Keim, R.: All about circuits: How to train a basic perceptron neural network (2019)
20. García-Ordás, M. T., Benítez-Andrades, J. A., García-Rodríguez, I., Benavides, C., Alaiz-Moretón, H.: Detecting respiratory pathologies using convolutional neural networks and variational autoencoders for unbalancing data. Sensors, vol. 20, no. 4, pp. 1214 (2020) doi: 10.3390/s20041214
21. Morales, M. D., Antelis, J. M., Moreno, C., Nesterov, A. I.: Deep learning for gravitational-wave data analysis: A resampling white-box approach. Sensors, vol. 21, no. 9, pp. 3174 (2021) doi: 10.3390/s21093174.
22. Sormani, C.: A two-part feature: The mathematics of gravitational waves. Notices of the American Mathematical Society, vol. 64, no. 7, pp. 684–685 (2017) doi: 10.1090/noti1551
23. d'Inverno, R.: Introducing Einstein's relativity. EUA, Oxford University Press (1992)
24. Moore T. A.: A general relativity workbook. Gravitational wave polarization (2010)
25. LIGO Virgo Collaboration. Gravitational wave open science center: Tutorials (2017) www.gw-openscience.org/tutorials/

ISSN 1870-4069

Navegación libre y activa de un robot móvil controlado con BCI basada en SSVEP

Luis Fernando Román-Padilla, Luis Humberto Vaca,
Juan David Chailloux Peguero, Omar Mendoza-Montoya,
Javier M. Antelis

Tecnológico de Monterrey,
Escuela de Ingeniería y Ciencias,
México

{A01367067, A01638029,A00828218, omendoza83,
mauricio.antelis}@tec.mx

Resumen. El proyecto pretende dar soporte y apoyo a personas con enfermedades neurodegenerativas, como Esclerosis Lateral Amiotrófica (ELA) o lesión medular, mediante una interfaz cerebro-computadora basada en estímulos visuales SSVEP, para la teleoperación de un robot móvil (PuzzleBot); con el que se les ofrece un nuevo sistema de comunicación dentro de su entorno teniendo en cuenta sus limitaciones de movimiento. El robot dispone de una cámara en la que, mediante marcadores ArUco, se puede utilizar la realidad aumentada para generar señales o indicaciones que mejoren la orientación dentro del entorno. Actualmente, 8 sujetos sanos ya han sido capaces de utilizar el sistema para completar una ruta definida en un entorno real de oficina, lo que verifica la viabilidad de su uso; se espera realizar futuras pruebas con pacientes para que puedan navegar con el robot en hospitales y mejorar así su calidad de vida.

Palabras clave: BCI, navegación activa, entrenamiento, realidad aumentada.

Free and Active Navigation of a Mobile Robot Controlled with SSVEP-Based BCI

Abstract. Abstract The project aims to provide support and assistance to individuals with neurodegenerative diseases, such as Amyotrophic Lateral Sclerosis (ALS) or spinal cord injury, through a brain-computer interface based on visual SSVEP stimuli for teleoperation of a mobile robot (PuzzleBot). This offers them a new communication system within their environment, taking into account their limited mobility. The robot is equipped with a camera that utilizes ArUco markers to provide augmented reality signals or indications to improve orientation within the environment. Currently, 8 healthy subjects have been able to use the system to complete a defined route in a real office environment, verifying its feasibility for use. Future tests are expected to be conducted with patients so they can navigate with the robot in hospitals and improve their quality of life.

Keywords: BCI, active navigation, training, augmented reality.

1. Introducción

La ELA es una enfermedad neurológica progresiva que afecta a las neuronas que controlan los movimiento voluntarios e involuntarios. Su prevalencia puede darse de 3 a 6 personas por cada 100.000. [1] Normalmente, las células nerviosas del cerebro y de la médula espinal del paciente se atrofian con el tiempo, por lo que la enfermedad reduce la capacidad motora del paciente para realizar actividades de la vida diaria.

Una interfaz Cerebro-Computadora (BCI, en sus siglas en Inglés) es un sistema que permite comandar un dispositivo, como un robot móvil, que es el caso de nuestra investigación, u otros sistemas móviles o virtuales, utilizando únicamente señales electroencefalográficas (EEG), sin necesidad de realizar ningún movimiento. Una BCI clasifica constantemente la actividad cerebral en curso y, por tanto, está siempre disponible para su monitorización [3].

El Potencial Evocado Visual de Estado Estacionario (SSVEP, en sus siglas en inglés) es un fenómeno en el que una fuente de luz emite una estimulación que oscila una frecuencia establecida. Esta estimulación provoca una respuesta en el cerebro que imita la frecuencia de la fuente luminosa cuando se observa detenidamente. Este fenómeno ocurre principalmente en la corteza visual del cerebro humano [4].

Es necesario ofrecer a estas personas una nueva alternativa de comunicación, la telepresencia se vuelve una herramienta importante en el día a día de los pacientes porque les permite mantener su calidad de vida al momento de perder capacidades motrices. Y la BCI puede utilizarse precisamente para este fin, pero esto conlleva una serie de retos. El principal es la generación de un sistema de navegación integrado en un BCI con el que las personas puedan tener acceso a una forma de navegación remota.

Diferentes autores han propuesto controlar un dispositivo robótico mediante BCI similar a nuestro proyecto, para mejorar la calidad de vida de estos pacientes, como es el caso aplicado a la automatización de una silla de ruedas utilizando otro fenómeno conocido como P300 donde se activaron una serie de botones para generar las acciones de una silla de ruedas previamente automatizada con una serie de personas sanas y posteriormente utilizada con pacientes con enfermedades neurodegenerativas, consiguiendo un sistema con un 86 % de efectividad en la selección de direcciones [6].

También existe un caso relacionado enfocado a mejorar la calidad de vida de pacientes neurológicos realizando un juego de laberintos para su entretenimiento, esto se consiguió mediante un sistema BCI basado en SSVEP para el movimiento de un pequeño robot móvil el cual recorría este mismo laberinto [7].

Otro trabajo relacionado con la navegación se centra en la comparación de Realidad Virtual(RV) y Realidad Aumentada (RA) con un sistema BCI, este caso de investigación se centra en el movimiento y control de un dron con RV versus RA a través de P300, donde se buscaba ver la comparación de navegación entre un entorno virtual o un entorno aumentado, donde resultó que no había diferencia en temas de navegación por el sistema y perspectiva de los participantes [8].

Otro proyecto similar fue el movimiento de un robot móvil alrededor de una pequeña ruta para las mediciones de tiempo de navegación [9]. Por último, tenemos una investigación que mezcla un sistema BCI para tareas de navegación en un juego de Laberinto junto con indicaciones a través de RA, esto mismo con un robot humanoide con varios sensores y cámaras, enfocado a probar el posible uso de un BCI en

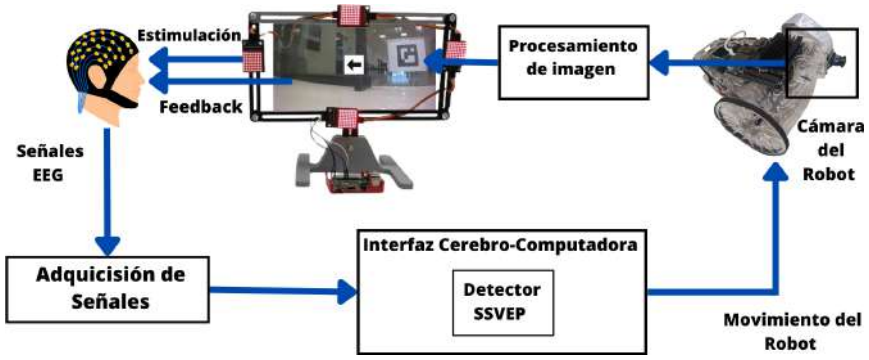


Fig. 1. Diagrama de sistema de navegación teleoperado por señales SSVEP.

medios de navegación y robótica [10]. El propósito de esta investigación es poder crear un sistema integrado para la el control, navegación y validación de un robot teleoperado con una BCI mediante el fenómeno SSVEP por personas sanas donde posteriormente se busca ayudar a pacientes con Esclerosis Lateral Amiotrófica (ELA), lesiones medulares y Atrofia Muscular Espinal (AME) para que puedan interactuar con su entorno teniendo en cuenta sus limitaciones de movimiento, su evolución dentro de sus mismas enfermedades y diferentes entornos.

El sistema propuesto y el trabajo de investigación constan de tres partes: Adquisición de datos EEG del participante, SSVEP-BCI y control del robot móvil. La parte de planificación de la ruta tiene como objetivo realizar una ruta de navegación óptima desde el punto de partida hasta el punto de destino. Esta ruta se indicaba mediante códigos con flechas flotantes mediante Realidad Aumentada (RA) sobre la misma ruta. Dentro de este mismo proceso, se obtuvieron las señales encefalográficas (EEG) del participante para su análisis.

Para generar la señal de control direccional se utilizó un sistema SSVEP-BCI, este mismo sistema se utilizó gracias a su facilidad de modificación y control de las frecuencias en contrario a otros paradigmas como lo podría ser P300. Esto es por medio de una serie de módulos de Diodos Emisores de Luz (LED, por sus siglas en Inglés), a ciertas frecuencias que describiremos más adelante, estas señales pasan por un pequeño procesamiento para que la señal sea enviada al robot móvil para controlar su movimiento. Donde a su vez tenía retroalimentación para el usuario a través de un sistema de visión con un módulo de cámara PIS-1685 con Raspberry Pi Camera Board V2.

El presente artículo está organizado de la siguiente manera: Métodos y materiales presenta en detalle los materiales y sistemas de medición utilizados dentro de la investigación. Se divide en 3 subsecciones (Interfaz BCI, Descripción del Robot y Sistema de Visión). Posteriormente, pasaremos a la sección de Validación del Sistema donde describiremos cómo se midió, validó y analizó el experimento; esta sección se compone de 5 subsecciones (Procedimiento, Calibración del Sistema, Tarea de navegación, Participantes y Validacion en línea), seguida de la sección de Resultados con 2 subsecciones (Resultados de la validación Online y Resultados de la tarea de navegación). Terminamos con la sección de Conclusión y referencias.

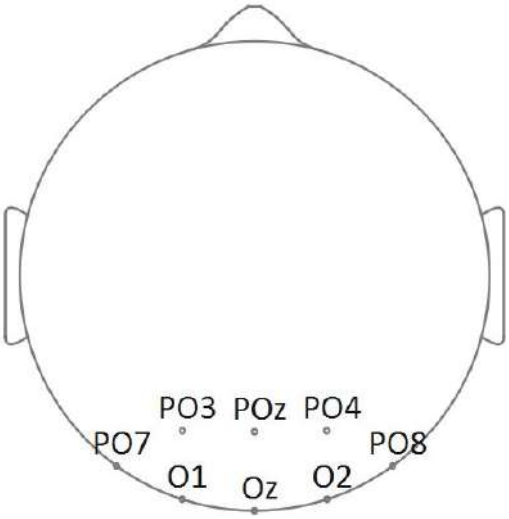


Fig. 2. Sistema de posicionamiento de electrodos utilizado para la captación del paradigma SSVEP.

2. Métodos y materiales

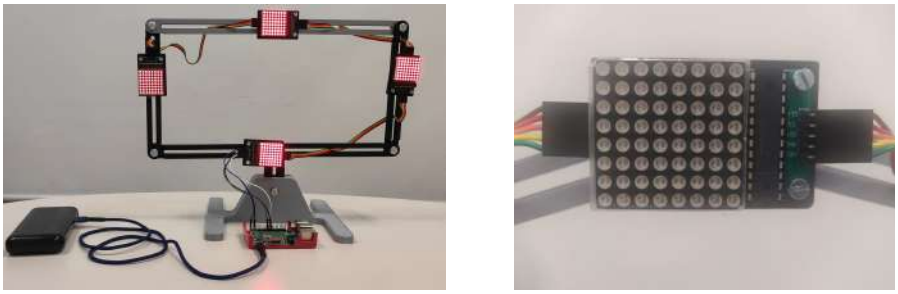
El sistema consta de varias partes enfocadas a la generación de un sistema de navegación, a continuación mostraremos como se dividen y componen cada una de ellas. El sistema consta de 4 partes esenciales (ver Fig.1).

- La BCI.
- Sistema de estimulación.
- Robot móvil.
- Sistema de visión del robot.

En la Fig.1 anterior se ilustra el flujo de trabajo del sistema de navegación. En la parte superior del esquema, encontramos el monitor que muestra la visualización proporcionada por la cámara de PuzzleBot. En la parte inferior, el participante genera una señal EEG en respuesta a los estímulos proporcionados por la matriz LED, la cual es adquirida por el sistema BCI. A continuación, la señal es procesada por el BCI y se traduce en un movimiento del robot.

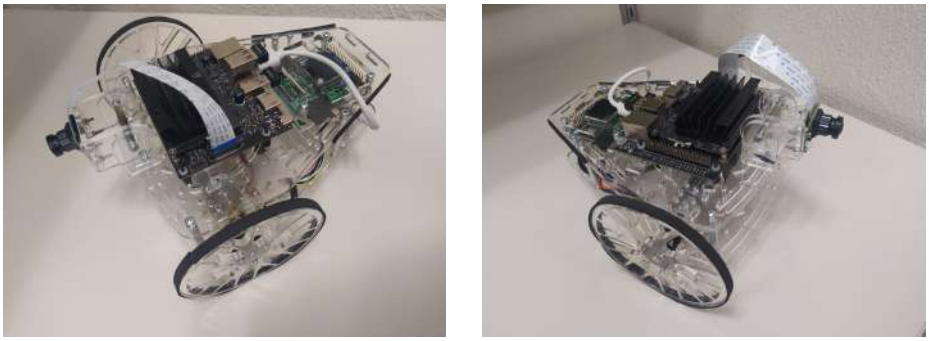
2.1. Interfaz BCI

2.1.1. Posicionamiento de electrodos para adquisición de señales EEG. En el experimento propuesto, se utilizó una BCI para controlar remotamente un robot móvil, que navega desde una posición dada en un entorno real de oficina a través de una ruta específica que fue previamente mostrada a los participantes. Se reclutaron 8 voluntarios, que previamente aceptaron participar en el experimento.



(a) (b)

Fig. 3. Módulo LED utilizado para la experimentación y tipo de LED utilizado.



(a) (b)

Fig. 4. Puzzlebot e integración del sistema de visión.

Se adquirieron señales EEG durante todo el experimento con 8 canales sobre la corteza visual en las posiciones PO7, PO3, POZ, PO4, O1, Oz, y O2 referenciadas en el lóbulo de la oreja derecha y conectadas a tierra en el lóbulo de la oreja izquierda (ver Fig.2). Las señales del EEG fueron amplificadas con un amplificador de bioseñal de alta precisión de 8 canales,filtradas por un filtro paso banda tipo FIR de 0,5-60HZ y con un filtro Notch a 60 Hz suprime la frecuencia de la línea de generación.

2.1.2. Estimulación de matriz LED. Se ha desarrollado un sistema de matrices LED parpadeantes que se encienden a diferentes frecuencias que probocan una respuesta cerebral, la cual imita a la frecuencia de estimulacion visual por cada módulo LED. Cada frecuencia se controla con precisión mediante una Raspberry Pi 3, y las frecuencias de parpadeo se pueden modificar mediante una sencilla reprogramación.

Los participantes se sentaron frente a un monitor con cuatro paneles estimulantes a cada lado, cada uno de ellos compuesto por LEDs de alta precisión con unas dimensiones de 3,5 cm $\times$ 3,5 cm y módulos de 8 x 8 (ver Fig.3). La elección de utilizar LED en lugar de un monitor LCD se basó en la obtención de una mejor respuesta en comparación con otros experimentos similares. Esta elección también se ha encontrado en otros trabajos previos, lo que respalda su eficacia en la generación de estímulos visuales [11].

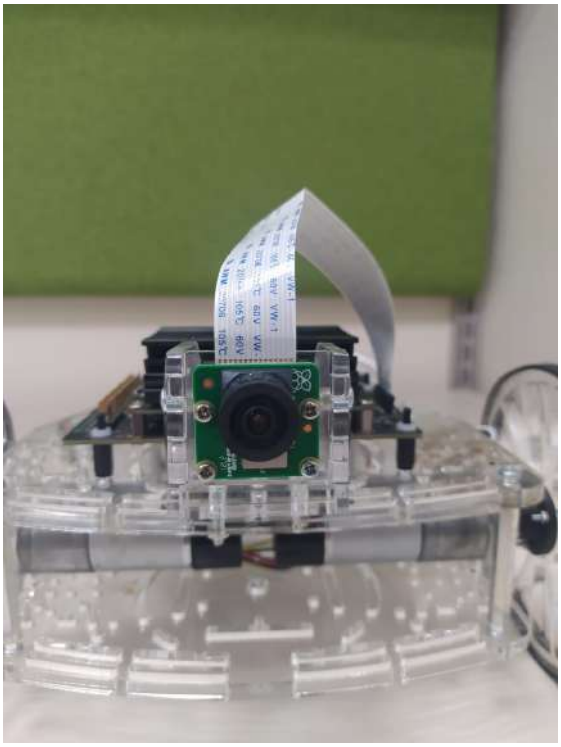


Fig. 5. Cámara utilizada para el sistema de visión en la tarea de navegación.

2.2. Descripción del robot

En los experimentos, se utilizó el robot móvil Puzzlebot, desarrollado por Manchester Robotics [14], que mide (20 cm de largo $times$ 25 cm de ancho $times$ 15 cm de alto) con módulo de cámara PIS-1685 para Raspberry Pi (Placa de cámara V2) montada en el mismo robot.4 Los voluntarios observaron en la pantalla las imágenes de la trayectoria por la que circulaba el robot (ver Fig.4). La comunicación entre el robot y el ordenador era inalámbrica (estándar Wi-Fi). El robot se movía a una velocidad lineal constante de 0,2 mm/s y angular de 0,05m/s. Esta velocidad estaba restringida por motivos de seguridad debido a la potencia de los motores.

2.3. Sistema de visión

Cuando el usuario quería dar una orden al robot, fijaba la vista en un estímulo específico y, la BCI detectaba la intención del usuario cuando este mismo fijaba la vista en un módulo LED en específico. De esta manera, la BCI podía producir cuatro órdenes para el robot: .ªdelante", "girar a la derecha", "girar a la izquierdaz "parar"(esta última estaba programada por defecto), cada una de las cuales correspondía a los estímulos visuales ubicados en la parte superior, derecha, izquierda e inferior, respectivamente.

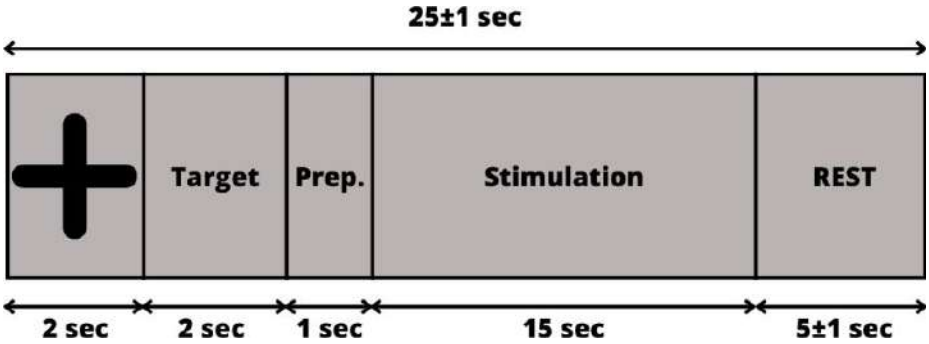


Fig. 6. Serie de tiempos en etapa de validación del sistema.

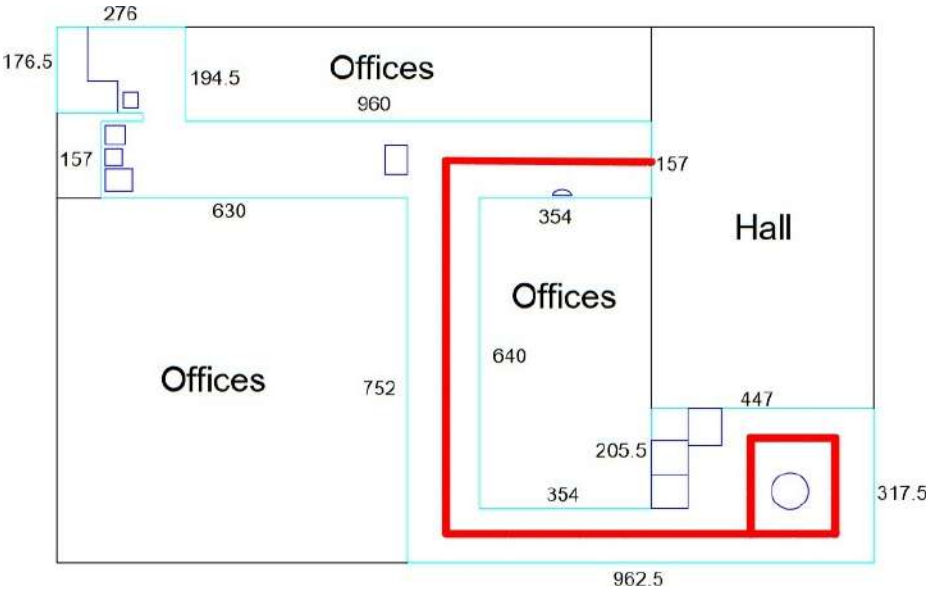


Fig. 7. Ruta mostrada a los participantes para la tarea de navegación.

Posteriormente, el robot se movía en la dirección deseada y, debido a que el entorno cambiaba alrededor del robot (debido a las personas que se encontraban dentro de las instalaciones y que participaban en el mismo experimento), la cámara capturaba la nueva situación. Finalmente, el usuario podía ver estos cambios en la pantalla y decidir enviar un nuevo comando (ver Figura 5).

3. Validación del sistema

En esta sección, describimos el diseño y las condiciones experimentales, así como la configuración del software, la ruta del robot y las tareas necesarias para reproducir el experimento.

(a) (b) (c)

Fig. 8. Marcadores ArUco para indicación de tareas de navegación con AR.

(a) (b)

Fig. 9. Participante en el momento del experimento.

3.1. Procedimiento

En primer lugar, se sentó al voluntario frente al monitor de medición utilizado para mostrar la visión del robot (54 cm de largo x 30 cm de alto) junto con la base LED, a continuación se colocaron los electrodos de electroencefalografía (EEG) secos y dos pegatinas en la parte posterior de las orejas, este proceso fue para preparar al participante para el proceso de adquisición de la señal.

Asimismo, se le explicó la serie de tareas a realizar dentro del experimento, así como los objetivos específicos de su participación en el proyecto. Una vez colocados los electrodos, se llevó a cabo una sesión de adquisición de datos la cual consiste en 1 minuto de entrenamiento del sistema. En la misma, el sujeto participante se encontraba en un estado de reposo total y movimiento limitado.

El participante enfocó sus ojos en uno de los cuatro cuadrados el cual estaba encendido de manera fija (sin ninguna frecuencia de parpadeo). Posteriormente, se pasó a una fase de pre-validación se contaba con 5 posibles frecuencias de estimulación: 8 Hz, 12 Hz, 15 Hz, 20 Hz y 27 Hz. De las cuales se seleccionaban 3 en dependencia de la respuesta del usuario, en la que cada panel de LEDs apuntaba en una de estas 3 direcciones: derecha, izquierda y adelante.

Tabla 1. Estudio preliminar participantes.

	Género		Mano Dominante		E.P		Tiempo Descanso	Lentes		
Clasificación	**Femenino**	**Masculino**	**Derecha**	**Izquierda**	**Si**	**No**	**P**	**Si**	**No**	**NL**
NP	1	7	7	1	4	4	6.8	3	2	3

Estas frecuencias no fueron múltiplos unas de otras para poder generar una respuesta del fenómeno SSVEP. Se le decía al sujeto que prestara atención a un cuadrado en específico para evaluar el rendimiento de la BCI. Posteriormente, se encendía la cámara del robot para que el sujeto tuviera una preámbulo de la siguiente tarea.

A continuación se pasaba a una segunda fase de validación, esta consistía en una serie de indicaciones dentro del sistema, un total de 21 estímulos visuales que se indicaban en la pantalla del sujeto, donde el participante se centraba en los LEDs hasta que se indicara lo contrario, pasando a un tiempo de receso de 5 segundos y continuando con el siguiente estímulo (ver Fig 6).

Finalmente, el participante se centraba en observar uno de los tres cuadrados parpadeantes de la base de LEDs para seguir una ruta específica establecida a través de la visión del robot móvil. Esta ruta se indicaba mediante códigos con flechas flotantes a través de realidad aumentada sobre la misma ruta.

Dentro de este mismo proceso, se obtuvieron las señales encefalográficas del participante para su posterior análisis. Una vez finalizada la ruta establecida, se realizó una encuesta al voluntario sobre el uso y percepción obtenida de la navegación.

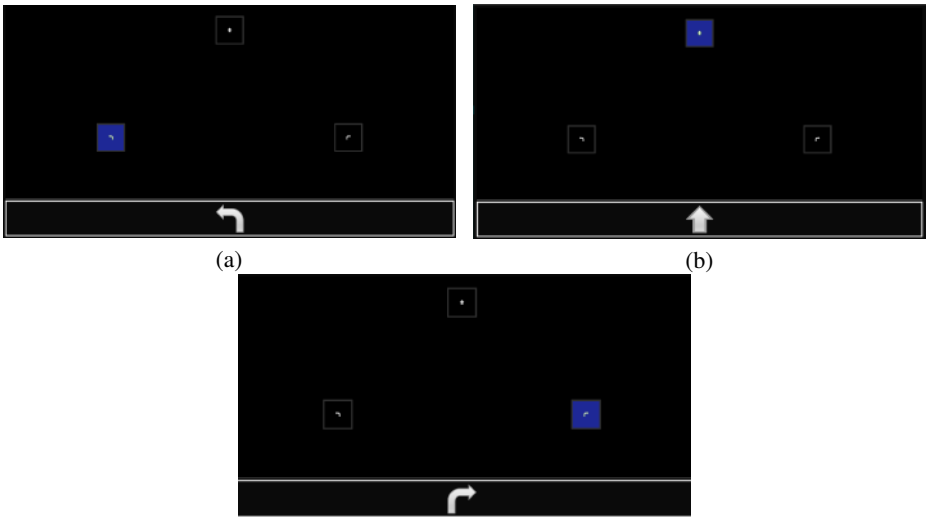
3.2. Calibración del sistema

Dentro del sistema se realiza un procedimiento de calibración previa a llegar a la segunda etapa, la cual es la validación online y posteriormente a la navegación. Tenemos un proceso de entrenamiento del sistema en el que adquirimos las señales EEG en estado neutro (sin ningún estímulo visual activo), durante el entrenamiento se le pide al participante que esté lo más quieto posible, este mismo entrenamiento tiene una duración de un minuto.Posteriormente visualizamos la calidad y velocidad de la respuesta SSVEP del participante utilizando los siguientes parámetros del sistema:

– Control de la tasa de errores: Proporción 30 % a 50 %.
– Ventaneos: 5 a 10.

El Control de la Tasa de Error es la medida porcentual clasificada de cada frecuencia por módulo LED. El número de ventaneos son el número de muestras que se generan por estímulo. La BCI selecciona tiene que estar en clasificación continua para poder generar la cantidad de datos suficientes para poder generar un movimiento dentro de la navegación.

Estos parámetros se modificaban a consideración de los experimentadores, que indicaban al participante qué led tenía que ver (arriba, abajo, derecha, izquierda) dentro de la matriz de LEDs. Dependiendo de si la selección era correcta y tenía una velocidad de respuesta aceptable, se procedía con el experimento.



(a) (b)

(c)

Fig. 10. Indicaciones para el participante en la validación en línea.

3.3. Tarea de navegación

El recorrido del experimento constaba de 8 giros (5 a la izquierda, 3 a la derecha, cada uno de aproximadamente 90°) y 9 comandos de avance que movían el Puzzlebot. El recorrido total era de 44 metros(ver Fig. 7). Durante la tarea de navegación, se daban indicaciones como ”girar a la derecha”, ”girar a la izquierdaz ”terminar”superponiendo imágenes en la cámara.

Para ello, se utilizaron marcadores Aruco [13], estos códigos son marcadores binarios cuadrados que permiten la realidad aumentada (AR, en sus siglas en Inglés) y pueden ser utilizados para la estimación de la pose; con esto, calculamos la distancia entre el robot y el marcador para poner flechas en la cámara cuando la persona necesitaba girar para que pudiera continuar con la ruta evitando colisiones (ver Fig. 8).

Durante el experimento, cuando se clasificaba un comando incorrecto y el usuario chocaba, se le movía manualmente para corregir su trayectoria y continuar la navegación.

3.4. Participantes

Nuestros participantes experimentales sólo tenían una restricción de seguridad, que era no tener antecedentes de epilepsia. Se midió a un total de 8 participantes, de 19-22 años de edad, donde había un total de 5 hombres y 3 mujeres todos con visión normal o corregida a normal y adultos sanos (estudiantes del Tecnológico de Monterrey) (ver Fig. 9). Se obtuvo el consentimiento por escrito de todos los participantes.

Se utilizó gafas cuando fue necesario para garantizar una correcta visualización. La grabación se llevó a cabo en una sala de laboratorio convencional con niveles estándar de iluminación. Cabe destacar que, debido a las limitaciones del equipo de registro (electrodos secos), se excluyó a las personas con cabello muy largo o grueso.

Tabla 2. Validación Online.

	Izquierda	**Adelante**	**Derecha**
1	71 %	100 %	100 %
2	85 %	100 %	100 %
3	85 %	100 %	85 %
4	100 %	100 %	100 %
5	100 %	100 %	100 %
6	100 %	100 %	100 %
7	85 %	100 %	100 %
8	85 %	85 %	100 %
Promedio	88.87 %	98.12 %	98.12 %
Desviación Estandar	10.35	5.3	5.3

Después del experimento, se hicieron varias preguntas preliminares a los participantes para posteriomente validar nuestros datos y realizar análisis posteriores de estos mismos. Podemos visualizar que 4 de los participantes necesitan llevar lentes. En la tabla tenemos diferentes abreviaturas que significan: (NP)-Número de participantes, (E.P)-Experiencia previa con BCI, (P)-Promedio tiempo de sueño previo al experimento, (NL)-Necesito lentes pero no los uso (ver Tab.1).

3.5. Validación en línea

Para comprobar que nuestro sistema era viable para la navegación, realizamos 2 mediciones esenciales. Se realizó una fase de validación online y un cuestionario relacionado con la navegación a los participantes. Dentro de la Validación Online, contamos con una serie de instrucciones dadas por el BCI.

Consta de 3 instrucciones esenciales, empezando por una çruz de fijación"que indica al participante que preste atención cuando reciba la siguiente indicación. Le sigue un LED direccional al que debe prestar atención hasta la tercera indicación con un mensaje RESTçon una duración de 5 segundos (ver Fig. 10).

4. Resultados

En esta sección, mostraremos los resultados de la fase de experimentación, la validación Online, los datos de navegación del robot y las respuestas al cuestionario de los participantes tras completar la tarea de navegación.

4.1. Resultados de la validación online

Tras la calibración, los sujetos pasaron a realizar una validación online previamente explicada, la siguiente tabla muestra los resultados de los participantes. En la tabla (D.E.)- Desviación Estándar (ver Tab. 2). Dentro de los resultados, podemos visualizar que los sujetos en su conjunto tuvieron promedios de respuesta altos en la prueba de validación en línea pero, hubo una respuesta más baja en el lado izquierdo con una

Tabla 3. Resultados de Navegación.

N. Participantes	Tiempo (s)	Colisiones	Izquierda	Adelante	Derecha	Total de movimientos
1	497	0	193	851	149	1193
2	560	0	192	875	152	1219
3	566	0	236	835	185	1256
4	741	0	242	814	205	1261
5	512	0	211	844	161	1216
6	370	0	188	888	140	1216
7	724	0	230	795	179	1204
8	979	1	199	893	140	1232
Promedio	618.6	12 %	211	849	164	1072
Desviación Estandar	189.03	0.35	21.73	34.84	23.53	23.80

frecuencia de 15Hz, en ese mismo estímulo. Esta desviación en el resultado es causada por el sujetos con peor respuesta, que fue el participante 1 donde mayoritariamente su respuesta fue baja en contraste a los estimulos de los modulos de la derecha y adelante que presentaron un 100 % de certeza.

4.2. Resultados de la tarea de navegación

Al llegar a la tarea de navegación se obtuvieron datos como el tiempo total del recorrido, el número de colisiones, el total de comandos recibidos por la BCI para las direcciones (Adelante, izquierda y derecha), y el número de movimientos totales(ver Tab.3). Dentro de la navegación, podemos visualizar varios puntos específicos relacionados con todas las variables posibles dentro del experimento.

– El sujeto con menor tiempo en la tarea de navegación fue número 6 con un total de 370 segundos dentro del recorrido, pero no es el sujeto con menor número de comandos.

5. Discusión

Así que podemos suponer que independientemente de la cantidad de datos que enviemos todo depende del tipo de respuesta que obtengamos del BCI. Anteriormente algunos trabajos han mencionado que este tipo de sistema es viable para el tema de navegación, y comparándolo con nuestros resultados de la utilización del mismo paradigma SSVEP mediante módulos LED pero con la diferencia que en nuestra investigación realizabamos trayectos de largas distancias con alrededor 42 metros de distancia en comparación de alrededor de 10 metros descritos en la parte de obstaculos dentro de los trabajos relacionados.

[5], otra diferenciación al nuestro es la utilización de realidad aumentada dentro de las indicaciones del recorrido, en comparación con los trabajos relacionados, fueron establecido figuras con idicaciones de Realidad Virtual pero con un recorrido totalmente controlado, en comparación al realizado donde el sujeto tomaba la decisión de la navegación dentro del recorrido establecido [12].

Sin embargo, el nivel de fatiga visual provocado por los estímulos puede afectar al rendimiento y a los resultados de los participantes durante las pruebas, estos se ven afectados porque al alcanzar un mayor nivel de fatiga persivida la respuesta del BCI se ve afectada. Enfocandonos en la respuesta de la BCI-Robot consideramos que es posible tener un buen control y orientación de navegación, pero, se necesita una buena referencia espacial para la navegación de cualquier ruta, como muestran los resultados de esta investigación.

Una Validación Online previa es una gran referencia cuando se quiere saber cómo va a ser la respuesta en la navegación del participante ya que existe una gran relación entre los resultados de la Validación Online y la respuesta percibida en la ruta de Navegación. Es importante mencionar que se encontró que no hay relación entre la cantidad de tiempo de la carrera y el número de movimientos realizados.

Esto se debe a que la velocidad de navegación depende específicamente de la respuesta del sujeto, y también la respuesta de los participantes se ve afectada por los movimientos del robot cámara en términos de atención, y respuesta a las señales. Una solución a este problema podría ser rediseñar la ruta para que fuera un poco más corta o con un menor número de estímulos para reducir la fatiga visual de los participantes.

Para mejorar el sistema, se podrían utilizar electrodos húmedos para evitar cualquier ruido en el entorno, ya que los electrodos secos tienden a ser más sensibles a cualquier perturbación ambiental. El experimento presentó algunos problemas, como la dificultad de mantener a los participantes quietos y la influencia del grosor y longitud del cabello en las mediciones, lo que dificultó la toma de medidas en distintas personas, especialmente en mujeres. También se podrían utilizar frecuencias más altas para mejorar la respuesta de los participantes, evitar la fatiga visual y mejorar la respuesta dentro de un sistema de navegación.

Otra forma de mejorar este sistema es la implementación de un Sistema de Control para el robot, enfocandonos en el control de los actuadores y navegación del robot, donde necesitamos hacer una relación entre el número de señales o estímulos recibidos de la BCI y el movimiento enviado y procesado del robot.

Referencias

1. Fernández-Lerones, M. J., de la Fuente-Rodríguez, A.: Esclerosis lateral amiotrófica: Un diagnóstico incierto. SEMERGEN - Medicina de Familia, vol. 36, no. 8, pp. 466–470 (2010) doi: 10.1016/j.semerg.2010.03.006
2. González, R. C., Woods, R. E.: Digital image processing. Pearson, 4ta. ed. (2018)
3. Zhan, X., Zhou, T.: Application of two-dimensional gel electrophoresis in combination with mass spectrometry in the study of hormone proteoforms. Mass Spectrometry - Future Perceptions and Applications, IntechOpen (2019) doi: 10.5772/intechopen.82524
4. Liu, Y., Li, X., Yang, C.: Design of a control platform for mobile robot with SSVEP-BCI system. In: IEEE International Conference on Cybernetics and Intelligent Systems and IEEE Conference on Robotics, Automation and Mechatronics, pp. 198–203 (2019) doi: 10.1109/cis-ram47153.2019.9095824
5. Diez, P. F., Mut, V. A., Laciar, E., Perona, E. M. A.: Mobile robot navigation with a self-paced brain–computer interface based on high-frequency SSVEP. Robotica, vol. 32, no. 5, pp. 695–709 (2013) doi: 10.1017/s0263574713001021

6. He, S., Zhang, R., Wang, Q., Chen, Y., Yang, T., Feng, Z., Zhang, Y., Shao, M., Li, Y.: A P300-Based threshold-free brain switch and its application in wheelchair control. In: IEEE Transactions on Neural Systems and Rehabilitation Engineering, vol. 25, no. 6, pp. 715–725 (2017) doi: 10.1109/tnsre.2016.2591012
7. Wu, C. M., Chen, Y. J., Zaeni, I. A. E., Chen, S. C.: A new SSVEP based BCI application on the mobile robot in a maze game. In: International Conference on Advanced Materials for Science and Engineering, pp. 550–552 (2016) doi: 10.1109/icamse.2016.7840197
8. Quiles, E., Dadone, J., Chio, N., García, E.: Cross-platform implementation of an SSVEP-Based BCI for the control of a 6-DOF robotic arm. Sensors, vol. 22, no. 13, pp. 5000 (2022) doi: 10.3390/s22135000
9. Kapeller, C., Hintermuller, C., Abu-Alqumsan, M., Pruckl, R., Peer, A., Guger, C.: A BCI using VEP for continuous control of a mobile robot. In: 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 5254–5257 (2013) doi: 10.1109/embc.2013.6610734
10. Wang, Y., Zhang, X., Li, K., Wang, J., Chen, X.: Humanoid robot control system based on AR-SSVEP. In: Proceedings of the 6th International Conference on Computing and Artificial Intelligence, pp.529–533 (2020) doi: 10.1145/3404555.3404625
11. Mu, J., Grayden, D. B., Tan, Y., Oetomo, D.: Comparison of steady-state visual evoked potential (SSVEP) with LCD vs. LED stimulation. In: 42nd Annual International Conference of the IEEE Engineering in Medicine and Biology Society (2020) doi: 10.1109/embc44109.2020.9175838
12. Quiles, E., Dadone, J., Chio, N., García, E.: Cross-platform implementation of an SSVEP-Based BCI for the control of a 6-DOF robotic arm. Sensors, vol. 22, no. 13, pp. 5000 (2022) doi: 10.3390/s22135000
13. OpenCV. Detection of ArUco markers (2023) docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html
14. Manchester Robotics. Puzzlebot: A revolution in robotics (2023) manchester-robotics.com/

ISSN 1870-4069

Transferencia de aprendizaje de una arquitectura de aprendizaje profundo para la separación de frecuencias musicales

Esteban Uriel Ildefonso-Orozco[1], Alberto Jorge Rosales-Silva[1], Armando Adrián Miranda-González[2], Jena Marie Vianney Kinani[3], Dante Mújica Vargas[4], Ponciano Jorge Escamilla Ambrosio[5]

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

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

[3] Instituto Politécnico Nacional, Unidad Profesional Interdisciplinaria de Ingeniería Campus Hidalgo, México

[4] Tecnológico Nacional de México, Departamento de Ciencias de la Computación, México

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

{eildefonsoo1500, amirandag1100}@alumno.ipn.mx, {arosaless, jkinani}@ipn.mx, dantemv@cenidet.edu.mx, pescamilla@cic.ipn.mx

Resumen. Este artículo propone un enfoque de aprendizaje por transferencia para abordar la separación de frecuencias musicales en pistas de audio no disponibles públicamente. La arquitectura combina una red convolucional (U-NET) y una red recurrente (LSTM) para segmentar y organizar la información a lo largo de una línea de tiempo utilizando coeficientes cepstrales de frecuencia Mel (MFCC). Esta arquitectura permite la extracción de pistas de audio individuales correspondientes a diferentes instrumentos (bajo, batería, voz y melodía), comúnmente conocidas como STEMS. Además de abordar la escasez de recursos en la separación de fuentes y satisfacer la creciente demanda de habilidades de producción musical, también facilita el aprendizaje y la práctica musical, fomentando la creatividad y la exploración de nuevas ideas musicales. Se destaca que los beneficios de esta arquitectura se enfocan exclusivamente a fines educativos y de obtención de pistas de canciones inaccesibles. Esta arquitectura de aprendizaje profundo propuesta representa una alternativa automatizada para la obtención de STEMS.

Palabras clave: LSTM, MFCC, neural networks, U-Net.

Transfer Learning of a Deep Learning Architecture for Musical Frequency Separation

Abstract. This article proposes a transfer learning approach to address the separation of musical frequencies in publicly unavailable audio tracks. The architecture combines a convolutional network (U-NET) and a recurrent network (LSTM) to segment and organize information along a timeline using Mel frequency cepstral coefficients (MFCC). This architecture enables the extraction of individual audio tracks corresponding to different instruments (bass, drums, vocals, and melody), commonly known as STEMS. In addition to addressing the scarcity of resources in source separation and meeting the growing demand for musical production skills, it also facilitates music learning and practice, fostering creativity and the exploration of new musical ideas. It is emphasized that the benefits of this architecture are exclusively focused on educational purposes and obtaining tracks from inaccessible songs. This proposed deep learning architecture represents an automated alternative for obtaining STEMS.

Keywords: LSTM, MFCC, neural networks, U-Net.

1. Introducción

Este artículo presenta un enfoque innovador para la separación de frecuencias musicales en pistas de audio utilizando aprendizaje profundo. A diferencia de otros métodos que requieren acceso a pistas de audio de un estudio de grabación, este enfoque propone el uso de redes neuronales para separar automáticamente las pistas de bajo, batería, voz y melodía a partir de audios disponibles en línea.

Esto hace que sea más accesible para su uso en diferentes aplicaciones. En este trabajo se exploran diferentes criterios de evaluación para validar la efectividad del enfoque propuesto. Las pistas de audio digital se dividen en STEMS, que son componentes musicales independientes que incluyen el bajo, la batería, la voz y la melodía.

Cada uno de estos componentes está definido por su contenido de frecuencias específicas y se pueden manipular individualmente para crear mezclas personalizadas [13]. En este trabajo se propone una arquitectura que combina una red neuronal convolucional (U-NET) y una red neuronal recurrente (LSTM) para abordar el problema conocido como el 'Efecto de fiesta de cóctel' [2].

Este fenómeno se refiere a la habilidad de enfocar la atención auditiva en un estímulo particular mientras se filtra un conjunto más amplio de estímulos, de manera coloquial sería como una persona puede concentrarse en una sola conversación en medio de una sala ruidosa durante una fiesta.

La arquitectura propuesta busca solucionar este problema mediante la identificación y separación de cada instrumento en una mezcla de audio, lo que resulta en pistas de audio separadas para cada uno de ellos. El entrenamiento se lleva a cabo utilizando la base de datos denominada sigsep musdb18, que consta de un total de 150 canciones completas de diferentes estilos.

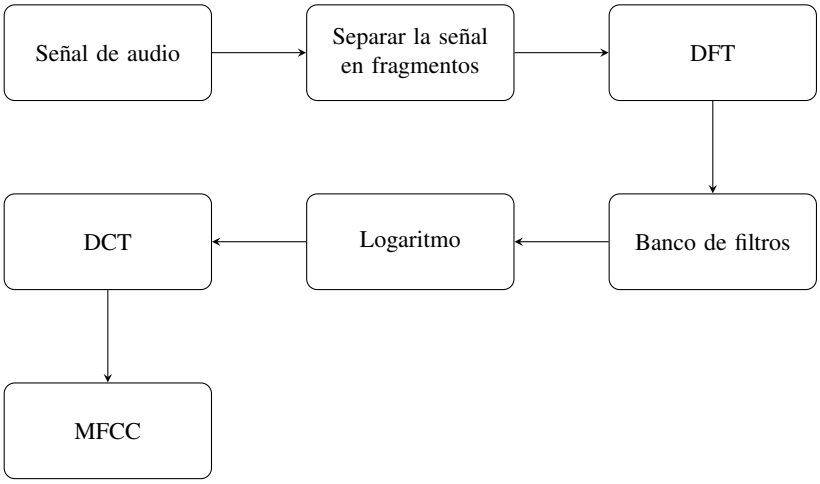


Fig. 1. Diagrama de bloques para obtener los MFCC.

Esta base de datos incluye tanto las mezclas estéreo como las fuentes originales, que son las pistas individuales conocidas como STEMS. Las grabaciones tienen una frecuencia de muestreo de 44.1 KHz y están divididas en conjuntos de entrenamiento y prueba.

Esta base de datos ha sido diseñada y evaluada como referencia para algoritmos de separación de fuentes, y es proporcionada por la Campaña de Evaluación de Separación de Señales de 2018 (SiSEC 2018) [14]. Se propone una arquitectura que combina una estructura U-Net y LSTM para separar pistas de audio.

La red U-Net es comúnmente utilizada para la segmentación semántica de imágenes, pero aquí se utiliza para separar pistas de audio. La red LSTM se utiliza para solucionar la línea del tiempo de las pistas de audio, lo que es crucial para la correcta separación de las pistas.

1.1. Red U-NET

Una Red U-Net es un tipo especial de red neuronal convolucional utilizada para la segmentación semántica de imágenes. Durante su proceso, aplica convoluciones para reducir dimensionalmente la información y, posteriormente, la recupera con el uso de deconvoluciones [12]. Además de las convoluciones y deconvoluciones, la red U-Net también contiene funciones de activación que se encargan de devolver una salida a partir de un valor de entrada.

Estas funciones pueden ser ReLU (Unidad Lineal Rectificada), sigmoide, tangente hiperbólica, entre otras [15]. La arquitectura de la Red U-Net permite que la red aprenda tanto características locales como globales de la imagen, lo que la hace especialmente útil en tareas de segmentación semántica.

La arquitectura propuesta utiliza la segmentación semántica de los MFCC para separar cada pista de los instrumentos.Aplicando la red U-Net logrando separar cada componente musical en pistas separadas.

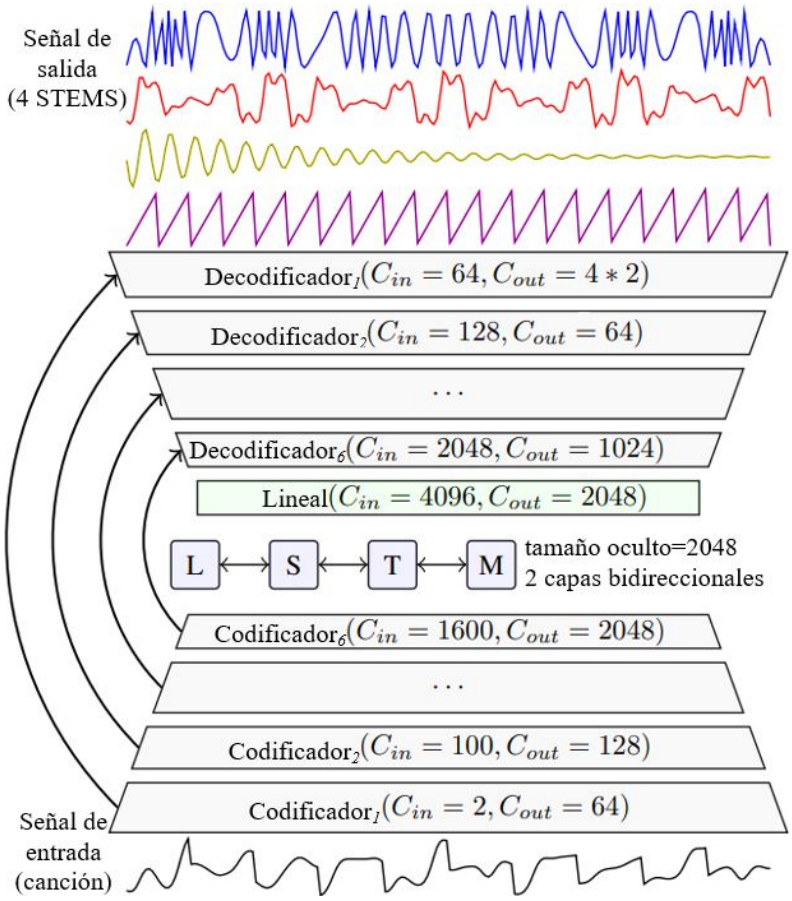


Fig. 2. Arquitectura de separación de frecuencias musicales.

1.2. Funciones de activación

La función de activación ReLU (Unidad Lineal Rectificada) se define como:

$$f(x) = \text{máx}(0, x) = \begin{Bmatrix} 0 \text{ para } X < 0 \\ 0 \text{ para } X \geq 0 \end{Bmatrix}, \tag{1}$$

donde $f(x)$ representa la salida de la función de activación, donde y x es el valor de entrada. La función ReLU es comúnmente utilizada en redes neuronales debido a su simplicidad y eficiencia computacional. Además, su comportamiento no lineal permite a la red aprender relaciones no lineales entre los datos de entrada y salida.

En resumen, la función ReLU se encarga de anular los valores negativos de la entrada y mantener los valores positivos para ser utilizados como entrada en la siguiente capa de la red. La función GLU (Gated Linear Unit) es una función de activación que se utiliza en redes neuronales. Esta función reduce a la mitad el número de canales de la capa anterior de la red [6]. La ec. 2 que define la función GLU es la siguiente:

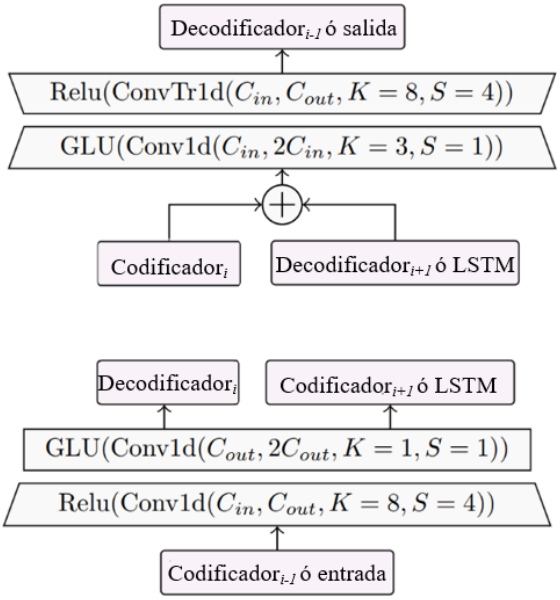


Fig. 3. Bloque i: Codificador y decodificador.

$$h(X) = (X \cdot W + b) \bigotimes \sigma\left(X \cdot V + c\right), \tag{2}$$

donde $h(X)$ representa la salida de la función de activación GLU, σ es la función de activación sigmoide, $\bigotimes$ representa la operación de multiplicación puntual entre matrices, W y V son matrices de pesos que se aprenden durante el entrenamiento de la red, b y c son parámetros de bias que también se aprenden durante el entrenamiento de la red, y X es la entrada a la capa de la red que utiliza la función GLU como función de activación [3].

Se utilizan las funciones de activación ReLU y GLU en la arquitectura propuesta debido a sus propiedades de no linealidad y eficiencia computacional. La función ReLU es simple y eficiente, lo que la hace adecuada para su uso en redes neuronales profundas. Además, ayuda a evitar el problema del desvanecimiento del gradiente, que puede afectar negativamente el rendimiento del modelo. Por otro lado, la función de activación GLU se utiliza para resaltar las características más importantes de los datos de entrada y ayudar a la red a aprender patrones más complejos en los datos.

1.3. Red LSTM

Una red LSTM es una arquitectura de red neuronal recurrente que se enfoca en mantener y recordar información a largo plazo. La principal ventaja de una red LSTM sobre otras arquitecturas recurrentes es su capacidad para evitar el problema del desvanecimiento del gradiente, que puede ocurrir cuando se propagan los errores a través de muchas capas de la red.

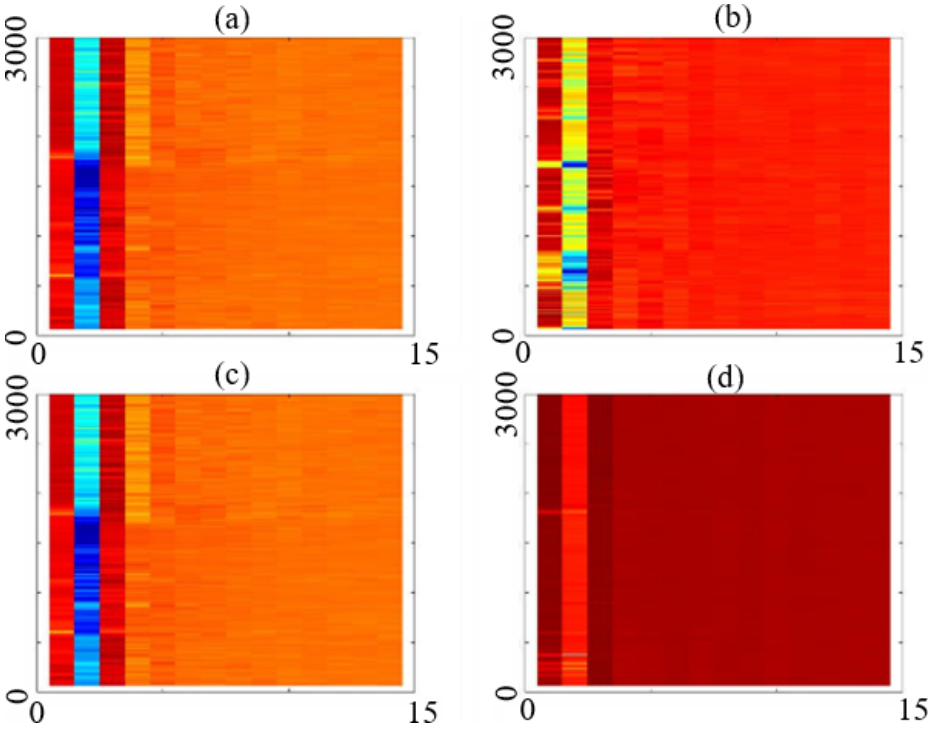


Fig. 4. Comparativa de STEMS: (a) Voz en estudio vs (b) Voz con la arquitectura. (c) Bajo en estudio vs (d) Bajo con la arquitectura.

Los LSTM utilizan una estructura de celdas de memoria con compuertas que regulan el flujo de información en la celda, lo que les permite recordar y olvidar información según sea necesario [11]. En el contexto de la arquitectura propuesta, la red LSTM se utiliza para organizar la salida de la red en una línea de tiempo, lo que permite que los STEMS de la canción tengan la misma duración de tiempo que la entrada y mantengan la coherencia temporal.

1.4. Coeficientes cepstrales en las frecuencias de Mel

A la entrada de la red U-NET se le proporciona un archivo de audio el cual es procesado para obtener los coeficientes Cesptrales en las frecuencias de mel (MFCC) para reducir la dimensionalidad y extraer características relevantes de los espectros de frecuencia.

Los MFCC son similares al espectro de frecuencia, pero en lugar de representar la amplitud en cada banda de frecuencia, se utilizan los coeficientes cepstrales de Mel para representar la energía en diferentes bandas de frecuencia. Esto permite una mejor identificación de los diferentes STEMS y su posterior segmentación en la U-NET [5]. La Figura 1. muestra el diagrama de flujo del proceso para obtener los MFCC:

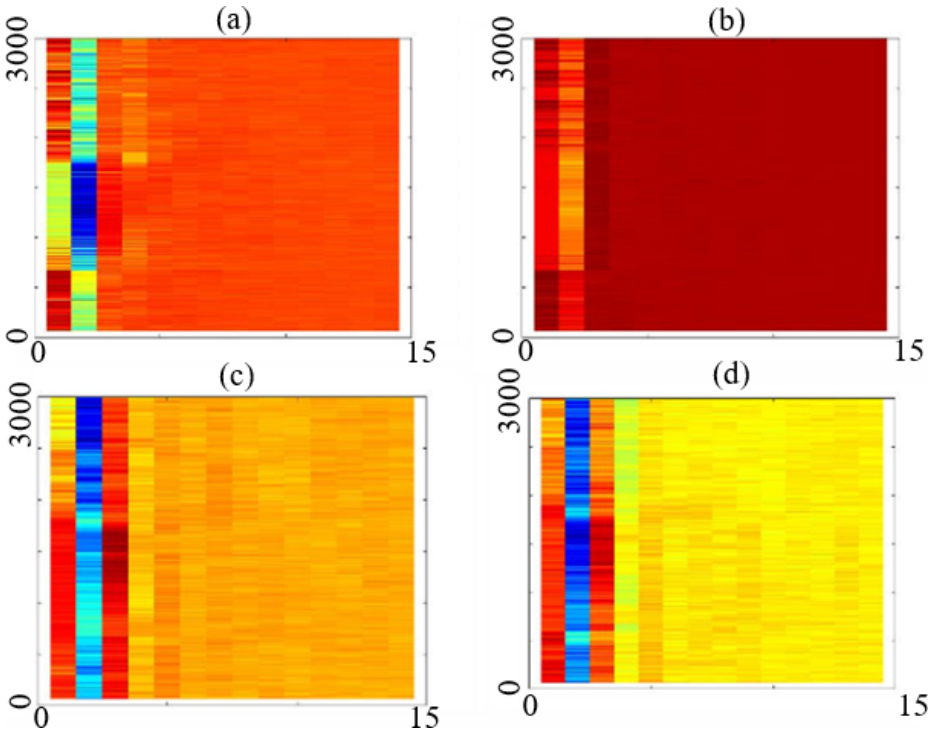


Fig. 5. Comparativa de STEMS: (a) Batería en estudio vs (b) Batería con la arquitectura. (c) Otros en estudio vs (d) Otros con la arquitectura.

1. Señal de audio: Esta entrada contiene la canción a la cual se le quieran extraer los MFCC.
2. Separar la señal en fragmentos cortos y de tiempos iguales: las frecuencias en una señal cambian con el tiempo, es por eso que se analiza en pequeños tramos de tiempo (comúnmente se realiza un corte de entre 20 y 40 ms).
3. Transformada discreta de Fourier (DFT): Se calcula la transformada, obteniendo así la representación en el dominio de la frecuencia (periodograma).
4. Banco de filtros: Lo que hace este filtro es tomar grupos de periodogramas y sumarlos, para tener una idea de cuanta energía existe en varias regiones de frecuencia.
5. Logaritmo: Los resultados son sometidos al logaritmo que representan una aproximación al comportamiento del odio humano al escuchar música.
6. Transformada de coseno discreta (DCT): Se aplica para obtener una correlación de las energías de potencia en dB de los MFCC.
7. Al finalizar el proceso se obtienen los MFCC.

Tabla 1. Análisis de SDR en dB.

	SDR Vocales	SDR Bajo	SDR batería	SDR Otros
STEMS obtenidos de estudio musical	8.91	5.94	11.05	9.80
STEMS obtenidos con red neuronal	7.99	4.65	8.99	9.11

2. Metodología / desarrollo

La arquitectura propuesta está diseñada para tomar una mezcla estéreo como entrada (denominada $C = 2$) y generar una salida estéreo para cada fuente, se puede considerar que la U-Net es la arquitectura principal, seguido de un codificador convolucional, un LSTM bidireccional y un decodificador convolucional.

El codificador se encarga de reducir la dimensionalidad de la entrada mediante la aplicación de múltiples capas convolucionales, mientras que el decodificador se encarga de reconstruir la salida a su tamaño original mediante la aplicación de múltiples capas deconvolucionales.

El LSTM bidireccional se utiliza para capturar las relaciones temporales a lo largo del tiempo en la entrada. Además, la arquitectura utiliza conexiones U-Net para vincular el codificador y el decodificador, lo que permite que la información de la entrada se transmita directamente a la salida. Esto permite que la red pueda realizar una segmentación de los STEMS de la canción en la salida, la arquitectura general se representa en la Figura 2.

El codificador consta de bloques convolucionales ($L = 6$) apilados y numerados del 1 al L. Cada bloque (i) está compuesto por una convolución con tamaño de núcleo $K = 8$, zancadas $S = 4$, canales de entrada $Ci - 1$, canales de salida Ci y una función de activación ReLU.

Luego, se realiza una convolución con tamaño de núcleo $K = 1$, canales de salida $2Ci$ y se aplica la función de activación GLU. Como las GLU reducen a la mitad el número de canales, la salida final del bloque i tiene canales de salida Ci. En la Fig. 3 se describe este bloque, Debido a que nuestros canales de entrada son una mezcla estéreo, es decir, izquierda y derecha, corresponde $Ci = 2$.

El decodificador se compone de L bloques, numerados de forma inversa de L a 1. Cada bloque i comienza con una convolución de tamaño de núcleo 3, zancada 1, canales de entrada/salida Ci y activación ReLU. Luego, se aplica una convolución transpuesta con un ancho de núcleo de 8 y zancada de 4, canales de salida $Ci-1$ y activación ReLU.

En la capa final, se sintetizan las fuentes (S) después de todos los bloques del decodificador. La capa final es lineal con $S \cdot Ci$ canales de salida, uno para cada fuente (4 canales estéreo en nuestro caso), sin ninguna función de activación. Cada uno de estos canales genera directamente una representación de la forma de onda correspondiente a una pista de audio [4].

3. Resultados y discusión

3.1. Comparativa de SDR y MFCC

La relación de fuente a distorsión (SDR) es uno de los métodos más comúnmente utilizados para evaluar la calidad de la salida de un sistema de separación de fuentes de música.

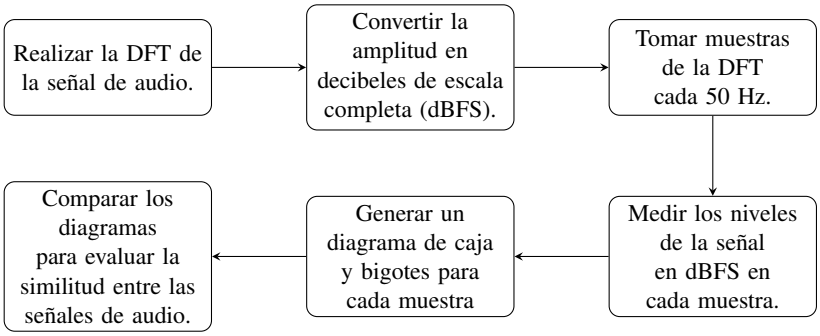


Fig. 6. Diagrama de bloques para obtener la comparativa en dBFS.

El SDR se utiliza para medir la relación entre las señales de las fuentes originales y las señales estimadas por el sistema de separación de fuentes [16]. Una mayor SDR indica una mejor separación de las fuentes y, por lo tanto, una mejor calidad de la pista de audio resultante, con ayuda de la ec. 3, donde:

$$\mathrm{SDR} = 10\log_{10}\left(\frac{\|S_{\text{target}}\|^2}{\|e_{\text{interf}} + e_{\text{noise}} + e_{\text{artif}}\|^2}\right), \tag{3}$$

donde S_{target} se refiere a la fuente verdadera, e_{interf} al error de interferencia, e_{noise} al error de ruido y e_{artif} al error de artefactos. Basándonos en los criterios de la sexta campaña comunitaria de evaluación de separación de señales de música (SiSEC-Mus) [14], se presentarán los resultados del SDR para evaluar la calidad de la salida del sistema de separación de fuentes de música. Además, se analizará la forma de onda del resultado para obtener una mejor comprensión de la calidad de la separación [10]. Para obtener los resultados presentados, se siguieron los siguientes pasos:

1. Se obtuvieron 30 segundos de STEMS de una canción producidos en un estudio de grabación de música, y se obtuvieron STEMS de la misma canción producidos con la red neuronal.
2. Se realizaron comparaciones entre los STEMS obtenidos con la arquitectura propuesta en el paso 1 y los obtenidos en el estudio de grabación mediante la extracción de MFCC para analizar las características de frecuencia de los coeficientes. Se detectó una diferencia en potencia en dB entre ambos, y se identificó que los coeficientes de color rojo indican una saturación en el sonido, lo que puede ser causado por la presencia de ruido no deseado debido a la filtración de instrumentos o partes de audio no deseadas. Cabe destacar que la presencia de ruido indica que la arquitectura no proporciona una separación de pistas de audio completamente fiel al original.
3. Se obtuvieron los resultados de SDR de los STEMS obtenidos en el paso 1, en donde Un valor más alto de SDR indica que la señal es más clara y está menos distorsionada, lo que se traduce en una mayor calidad de audio. En otras palabras, cuanto mayor es el valor de SDR, menor es la cantidad de ruido y distorsión en la señal de audio, lo que indica una mayor calidad de los STEMS.

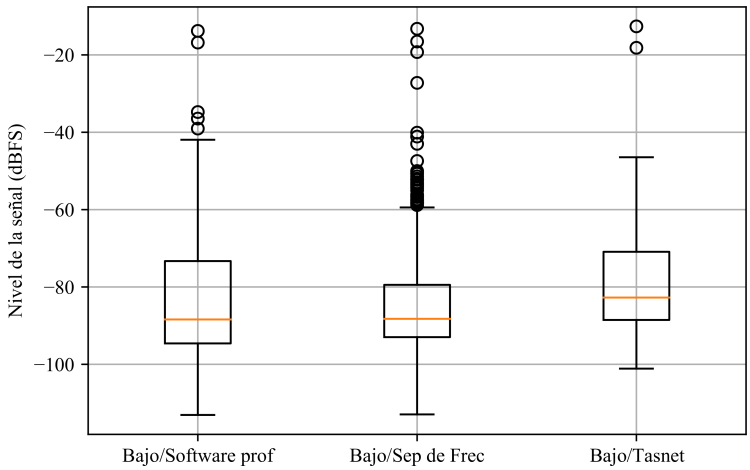


Fig. 7. Diagrama de caja del STEM de bajo.

Los resultados de SDR muestran que la arquitectura de separación de frecuencias de música proporciona resultados similares a los STEMS originales obtenidos en el estudio de grabación. Dichos resultados se pueden observar en la tabla 1.

3.2. Comparativa de nivel de señal

Una forma de analizar el nivel de una señal de audio es con la transformada discreta de Fourier (DFT) ec. 4, que es una técnica utilizada para analizar señales en el dominio de la frecuencia. La DFT nos proporciona un espectro que muestra la distribución de las diferentes componentes de frecuencia que componen una señal de audio con respecto a una magnitud de coeficientes adimensionales dados por la DFT [7]:

$$\text{DFT} = x(k) = \sum_{n=0}^{N-1} x(n) e^{-\mathrm{j}\,2\pi kn/2}, \tag{4}$$

donde, $x(k)$ es la k-ésima muestra en el dominio de la frecuencia, $x(n)$ es la n-ésima muestra en el dominio del tiempo, N es el número de muestras en la señal, k es el índice de la frecuencia en el dominio de la frecuencia y j es la unidad imaginaria.

Para obtener una representación más inteligible de dichos coeficientes adimensionales, se pueden transformar a un nivel en dBFS (decibeles en relación a la escala completa) con el uso de la ec. 5.

El dBFS es una medida utilizada comúnmente para conocer los niveles de audio digital, y permite expresar la amplitud de la señal en una escala logarítmica relativa al máximo nivel posible en el formato digital utilizado.

De esta manera, los valores en dBFS proporcionan una mejor comprensión del nivel de señal presente en la DFT, especialmente cuando se trabaja con señales de audio digitales y se requiere una medición más precisa y detallada de los niveles de señal:

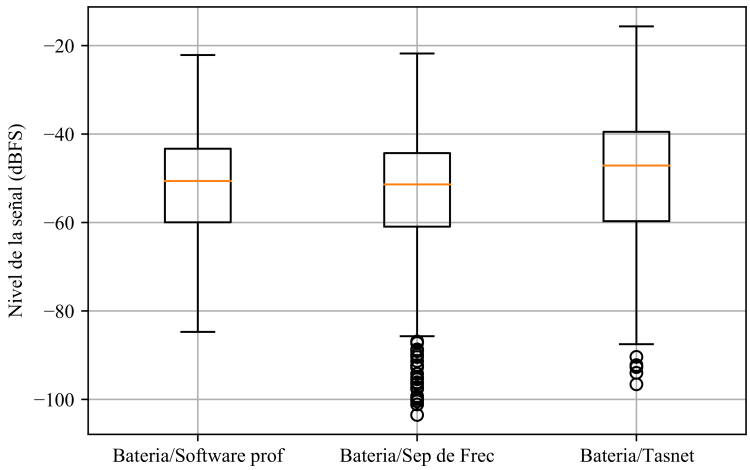


Fig. 8. Diagrama de caja del STEM de Batería.

$$\mathrm{dBFS}[k] = 20 \cdot \log_{10}(PR[k]), \tag{5}$$

donde, $\mathrm{dBFS}[k]$ representa el nivel de amplitud de la señal en una escala de decibeles a escala completa y $PR[k]$ es la potencia relativa de la señal en la muestra de la DFT. Posteriormente, se realizaron mediciones de los niveles en dBFS para cada intervalo de 50 Hz en nuestra DFT.

El análisis se llevó a cabo con la hipótesis de que la DFT obtenida a través de la arquitectura de aprendizaje profundo debería ser similar al STEM obtenido mediante un programa profesional. Con el fin de evaluar la calidad de la DFT obtenida por el modelo de aprendizaje profundo, se midieron los niveles de la señal en dBFS en intervalos regulares de frecuencia (Fig. 6).

Tomando como nota que en el caso de una frecuencia de muestreo de 44100 Hz, el teorema de Nyquist-Shannon establece que la frecuencia máxima que se puede representar es de 22050 Hz (la mitad de la frecuencia de muestreo). Esto implica que, para realizar mediciones de niveles en dBFS cada 100 Hz en una señal digital con frecuencia de muestreo de 44100 Hz, el rango de frecuencias que se puede analizar es de 0 a 22050 Hz [1].

Además, también se realizó la comparativa con Conv-TasNet [9], que es un algoritmo de separación de fuentes de audio que utiliza capas convolucionales unidimensionales (1D) en su red neuronal. TasNet, que significa Red de separación de audio en el dominio del tiempo, es la arquitectura general utilizada en Conv-TasNet.

Esta arquitectura consiste en capas convolucionales 1D seguidas de capas de activación y agrupamiento. El objetivo de Conv-TasNet es separar señales de audio en fuentes separadas, como la separación de voces en una grabación. Las capas convolucionales 1D aplican filtros a lo largo del eje temporal de la señal de entrada, capturando características locales y patrones en datos secuenciales. La salida de una capa convolucional 1D puede tener múltiples canales, lo que significa que se aplican varios filtros para capturar diferentes características.

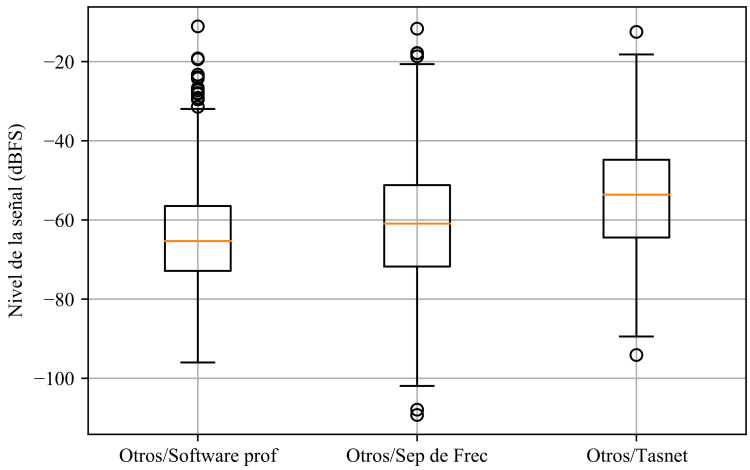


Fig. 9. Diagrama de caja del STEM de Otros.

Después de la convolución, se puede aplicar una función de activación, como ReLU, para introducir no linealidad en la red. Estas capas convolucionales 1D son especialmente adecuadas para el procesamiento de señales de audio debido a la estructura temporal de las formas de onda de audio, lo que las convierte en una herramienta útil en la separación de fuentes de audio, incluyendo la separación de frecuencias musicales. Datos por resaltar:

– La mediana en los audios obtenidos con la arquitectura de frecuencias musicales es similar a la mediana de la fuente original esto sugiere que esta arquitectura ha logrado preservar la ubicación central de la información en la señal original. La similitud en las medianas indica que la arquitectura de frecuencias musicales ha sido efectiva para capturar y mantener la tendencia central de la señal de audio, mientras que por otra parte en Conv-Tasnet se observa un desfase de esta mediana.
– En el diagrama del bajo, se puede observar que el valor máximo y mínimo es más amplio en la arquitectura de separación de frecuencias en comparación con Conv-TasNet por lo que nos habla de que tiene un mayor rango de nivel de señal con respecto a la potencia. Además, se pueden identificar varios valores atípicos en la arquitectura de separación de frecuencias, los cuales casi completan el parecido al rango del bajo de la fuente original. Esto podría sugerir que la arquitectura de separación de frecuencias captura una mayor variabilidad o componentes inusuales en comparación con la fuente original y Conv-TasNet.
– Es importante tener en cuenta que la presencia de valores atípicos puede tener diferentes causas, como artefactos de separación de fuentes o la presencia de componentes de bajo similares en otras fuentes. Por lo tanto, aunque los valores atípicos en la arquitectura de separación de frecuencias se asemejen al rango del bajo original, es necesario analizar con mayor detalle la naturaleza de estos valores atípicos y su relación con la fuente original.

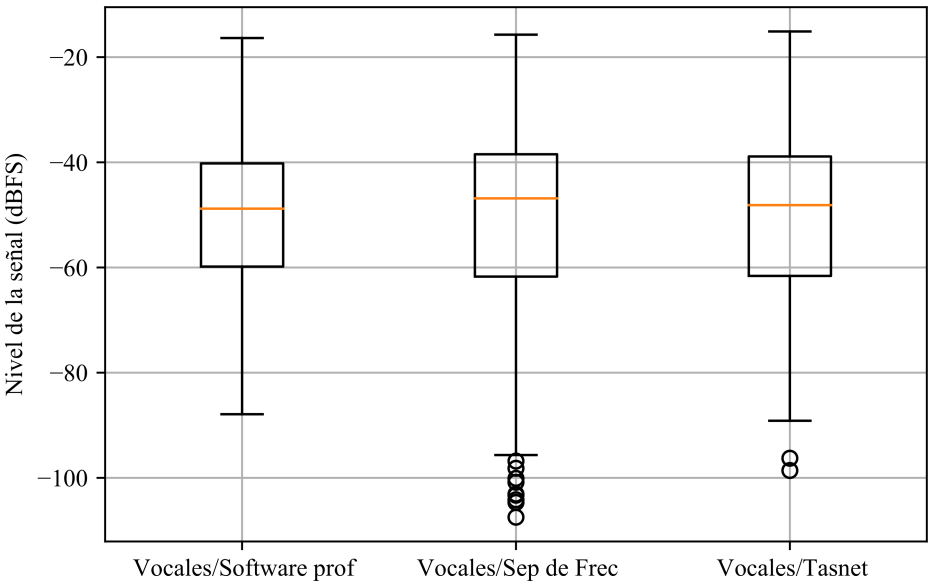


Fig. 10. Diagrama de caja del STEM de Vocales.

– El fenómeno de valores atípicos que se menciona anteriormente se observa con frecuencia en las arquitecturas de aprendizaje profundo utilizadas para el procesamiento de frecuencias musicales.
– El tamaño de la caja en el diagrama de las arquitecturas Conv-Tasnet y de separación de frecuencias musicales es similar al de la fuente original. Esta similitud en el tamaño de la caja sugiere que no existe una diferencia significativa en la dispersión de los datos entre ambas fuentes.

4. Conclusiones

Se ha logrado una transferencia de aprendizaje a una arquitectura que utiliza dos modelos de red neuronal para mejorar la calidad de la separación de las pistas de audio de los instrumentos principales que componen una canción.

Se ha analizado individualmente las dos arquitecturas que componen la red neuronal, comprendido su funcionamiento en el procesamiento de audio, y se ha entendido la información relevante para implementarla en la arquitectura de aprendizaje profundo.

Además, se ha ejecutado y aplicado la arquitectura propuesta para la separación de frecuencias de música la separación de frecuencias musicales. Estos logros permitirán avanzar en la mejora de la calidad de la separación de pistas de audio y en la aplicación de esta tecnología en la separación de fuentes de música.

Agradecimientos. Agradecimientos al Instituto Politécnico Nacional por el apoyo en la realización de este trabajo de investigación.

Referencias

1. Alibeigi, M., Hashemi, S., Hamzeh, A.: DBFS: An effective density based feature selection scheme for small sample size and high dimensional imbalanced data sets. Data and Knowledge Engineering, vol. 81-82, pp. 67–103 (2012) doi: 10.1016/j.datak.2012.08.001
2. Cherry, E. C.: Some experiments on the recognition of speech, with one and with two ears. The Journal of the Acoustical Society of America, vol. 25, no. 5, pp. 975–979 (1953) doi: 10.1121/1.1907229
3. Dauphin, Y. N., Fan, A., Auli, M., Grangier, D.: Language modeling with gated convolutional networks. In: Proceedings of the 34th International Conference on Machine Learning, vo. 70, pp. 933-941 (2017) doi: 10.48550/ARXIV.1612.08083
4. Défossez, A., Usunier, N., Bottou, L., Bach, F.: Demucs: Deep extractor for music sources with extra unlabeled data remixed (2019) doi: 10.48550/ARXIV.1909.01174
5. Huizen, R. R., Kurniati, F. T.: Feature extraction with mel scale separation method on noise audio recordings (2021) doi: 10.48550/ARXIV.2112.14930
6. Lederer, J.: Activation functions in artificial neural networks: A systematic overview (2021) doi: 10.48550/ARXIV.2101.09957
7. Lenssen, N., Needell, D.: An introduction to fourier analysis with applications to music. Journal of Humanistic Mathematics, vol. 4, no. 1, pp. 72–91 (2014) doi: 10.5642/jhummath.201401.05
8. Lostanlen, V., Cella, C. E.: Deep convolutional networks on the pitch spiral for musical instrument recognition (2016) doi: 10.48550/ARXIV.1605.06644
9. Luo, Y., Mesgarani, N.: Conv-TasNet: Surpassing ideal time–frequency magnitude masking for speech separation. In: IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 27, no. 8, pp. 1256–1266 (2019) doi: 10.1109/taslp.2019.2915167
10. Nakajima, H., Takahashi, Y., Kondo, K., Hisaminato, Y.: Monaural source enhancement maximizing source-to-distortion ratio via automatic differentiation (2018) doi: 10.48550/ARXIV.1806.05791
11. Staudemeyer, R. C., Morris, E. R.: Understanding LSTM – a tutorial into long short-term memory recurrent neural networks (2019) doi: 10.48550/ARXIV.1909.09586
12. Stoller, D., Ewert, S., Dixon, S.: Wave-U-Net: A multi-scale neural network for end-to-end audio source separation (2018) doi: 10.48550/ARXIV.1806.03185
13. Stöter, F. R., Uhlich, S., Liutkus, A., Mitsufuji, Y.: Open-Unmix - A reference implementation for music source separation. Journal of Open Source Software, vol. 4, no. 41, pp. 1667 (2019) doi: 10.21105/joss.01667
14. Stöter, F. R., Liutkus, A., Ito, N.: The 2018 signal separation evaluation campaign. In: Latent Variable Analysis and Signal Separation, Lecture Notes in Computer Science, vol 10891, pp. 293-–305 (2018) doi: 10.1007/978-3-319-93764-9_28
15. Takahashi, N., Goswami, N., Mitsufuji, Y.: MMDenseLSTM: An efficient combination of convolutional and recurrent neural networks for audio source separation. In: 16th International Workshop on Acoustic Signal Enhancement (IWAENC), pp. 106-110 (2018) doi: 10.1109/IWAENC.2018.8521383
16. Vincent, E., Gribonval, R., Fevotte, C.: Performance measurement in blind audio source separation. In: IEEE Transactions on Audio, Speech and Language Processing, vol. 14, no. 4, pp. 1462–1469 (2006) doi: 10.1109/tsa.2005.858005

ISSN 1870-4069

Aplicaciones de aprendizaje automático para estimar la evaporación en regiones áridas: caso de estudio Calera, Zacatecas

Luis Fernando Castillo Martínez, Julián González Trinidad, Hugo Enrique Júnez Ferreira, Carlos Francisco Bautista Capetillo, Cruz Octavio Robles Rovelo, José Armando Rodríguez Carrillo

Universidad Autónoma de Zacatecas,
Campus UAZ Siglo XXI,
Unidad Académica de Ingeniería Eléctrica,
México

{fercast, jgonza, hugo.junez, baucap, octavio.robles, jarmando.rc}@uaz.edu.mx

Resumen. La evaporación es un proceso fundamental dentro del ciclo hidrológico, el cual consiste en la pérdida de agua en forma de vapor desde la superficie terrestre hacia la atmósfera. Debido a su complejidad, se han implementado diferentes técnicas de Aprendizaje Automático (ML, por sus siglas en inglés), para comprender mejor este proceso. En esta investigación se realizó una comparación de tres modelos de ML, regresión lineal múltiple (MLR), bosques aleatorios (RF) y k-vecinos más cercanos (KNN), para estimar la evaporación en la región Calera, Zacatecas, México. Para evaluar el rendimiento de los modelos, se utilizaron las métricas coeficiente de correlación de Pearson (R), coeficiente de eficiencia Nash-Sutcliffe (NSE), raíz del error cuadrático medio (RMSE) y el error medio absoluto (MAE). El modelo regresión lineal múltiple (MLR) fue el que presentó mejor desempeño, con un coeficiente de correlación de Pearson (R) para la estación Calera de 0.97 y para Fresnillo de 0.94, de igual manera, se obtuvo un NSE de 0.93 y 0.87, un RMSE de 15.97 y 20.53 mm, y un MAE de 12.56 y 14.66 mm, respectivamente.

Palabras clave: Evaporación, aprendizaje automático, regresión lineal múltiple, bosques aleatorios, k-vecinos más cercanos.

Machine Learning Applications for Evaporation Estimation in Arid Regions: A Case Study in Calera, Zacatecas

Abstract. Evaporation is a key process in the hydrological cycle, consisting of the loss of water in vapor form from the land surface to the atmosphere. Due to its complexity, various Machine Learning (ML) techniques have been developed to better understand this phenomenon. In this research, it was compared the performance of three ML models, multiple linear regression (MLR), random

forest (RF), and k-nearest neighbors (KNN), for estimating evaporation in the region of Calera, Zacatecas, Mexico. To evaluate model performance, it was used the Pearson correlation coefficient (R), Nash-Sutcliffe efficiency coefficient (NSE), root mean square error (RMSE), and mean absolute error (MAE). The results showed that multiple linear regression performed best in the study area, with a Pearson correlation coefficient (R) of 0.97 for the Calera climatological station and 0.94 for Fresnillo. The NSE values were 0.93 and 0.87, the RMSE values were 15.97 and 20.53 mm, and the MAE values were 12.56 and 14.66 mm, respectively.

Keywords: Evaporation, machine learning, multiple linear regression, random forest, k-nearest neighbors.

1. Introducción

La evaporación es uno de los componentes más importantes del ciclo hidrológico, en el cual, el agua en su fase líquida parte de la superficie de la tierra hacia la atmósfera en forma de vapor de agua [2], es probablemente el parámetro más complicado y difícil de estimar de entre todos los elementos que integran el ciclo hidrológico debido a las interacciones complejas de los componentes hidrológicos como la superficie del agua, el suelo, el proceso atmosférico y la vegetación.

Por lo tanto, le estimación de la evaporación es un tema relevante en el manejo de los recursos hídricos y la agricultura, particularmente en regiones áridas y semi-áridas. Debido a la diferencia de temperatura, este fenómeno hidrológico es un proceso no lineal que ocurre en la naturaleza [13].

Este fenómeno es influenciado por el suministro de energía calórica y el gradiente de vapor de presión, los cuales están relacionados con datos meteorológicos tales como la temperatura del aire, radiación solar, humedad relativa, velocidad del viento y la presión atmosférica, a su vez estos aspectos están estrechamente relacionados con otros factores como la ubicación geográfica, la hora del día, la temporada del año y el tipo de clima [1].

La estimación de la evaporación en áreas áridas y semi-áridas es de suma importancia debido a la poca disponibilidad de agua en las fuentes de abastecimiento, así como, para los requerimientos de agua de la vegetación en los ecosistemas [16].

La pérdida de agua por evaporación se ha incrementado significativamente durante las últimas décadas, particularmente en regiones áridas y semi-áridas a lo largo del mundo. Por lo tanto, la estimación precisa de las tasas de evaporación es vital para diferentes contextos, como el presupuesto y manejo del agua para irrigación, hidrología, agronomía y manejo de los recursos hídricos.

Generalmente, la evaporación del agua superficial es medida utilizando dos métodos, el primero a través de una medición directa mediante tanques evaporímetros, y la segunda utilizando ecuaciones semi-empíricas basadas en variables climáticas generando una medición indirecta [10]. Existe una gran variedad de tanques evaporímetros que tienen diferente forma y tamaño, sin embargo, el tanque evaporímetro estándar clase A es uno de los tanques utilizados más comunes y con una aceptación a nivel global [8].

Tabla 1. Coordenadas de las estaciones climatológicas.

Estación	Latitud (Norte)	Longitud (Oeste)	Municipio
El Pardillo 3	$22°54'31''$	$102°39'34''$	Fresnillo
CEZAC	$22°10'49''$	$102°43'1''$	Calera
Fresnillo	$23°10'40''$	$102°53'20''$	Fresnillo
Calera	$22°54'00''$	$102°39'00''$	Calera

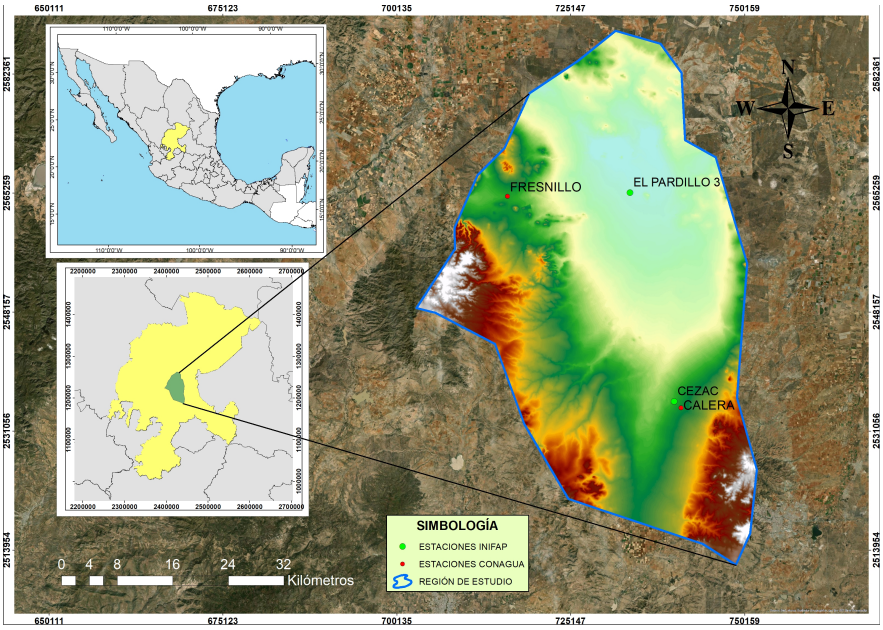


Fig. 1. Estaciones climatológicas de la región Calera.

El desarrollo de métodos de estimación indirecta basados en el uso de diferentes variables meteorológicas tales como las horas de luz solar, velocidad del viento, humedad relativa, precipitación, temperatura máxima, mínima y media a menudo son sugeridos para estimar la evaporación, especialmente cuando se trabaja con modelos empíricos y semi-empíricos.

Sin embargo, una de las limitantes para estimar la evaporación es la naturaleza dinámica de las variables meteorológicas aplicadas, debido a que es no estacionario y presenta características estocásticas. Por tanto, se requiere el desarrollo de modelos inteligentes, robustos y confiables para estimar la evaporación, el desarrollo de tales modelos se ha incrementado dentro del campo de la ingeniería y administración de recursos hídricos [18].

En los últimos años, investigadores han tratado de modelar el fenómeno de la evaporación a través de técnicas de Aprendizaje Automático (ML, por sus siglas en inglés) debido a los diferentes inconvenientes que se encuentran al momento de estimar la evaporación de manera directa, utilizando otros parámetros climatológicos que presentan relación con este proceso.

Tabla 2. Estadísticos de las variables.

Variable	Máximo		Mínimo		Media		Desviación estandar	
	C	F	C	F	C	F	C	F
T_1	30.40	31.70	17.10	18.20	23.98	25.35	2.97	3.06
T_2	13.80	13.60	-0.70	-4.50	7.84	6.56	4.10	4.98
T_3	22.00	22.20	8.70	8.00	15.93	15.99	3.34	3.79
P	207.80	273.70	0.00	0.00	38.84	36.87	45.69	47.85
HR_1	100.00	99.60	39.90	48.00	80.20	83.44	15.86	13.02
HR_2	60.00	59.30	8.00	7.70	27.43	24.30	12.41	11.01
HR_3	85.80	86.20	20.90	21.30	53.31	53.19	16.86	15.74
RS	1.02e6	9.47e5	5.02e5	4.55e5	7.43e5	7.12e5	1.28e5	1.30e5
V_1	26.50	30.10	13.10	10.40	20.06	20.15	3.18	4.26
V_2	12.90	14.20	4.40	3.90	8.64	7.91	1.87	2.01
EP	350.10	286.10	100.70	54.20	195.15	154.52	63.58	55.97

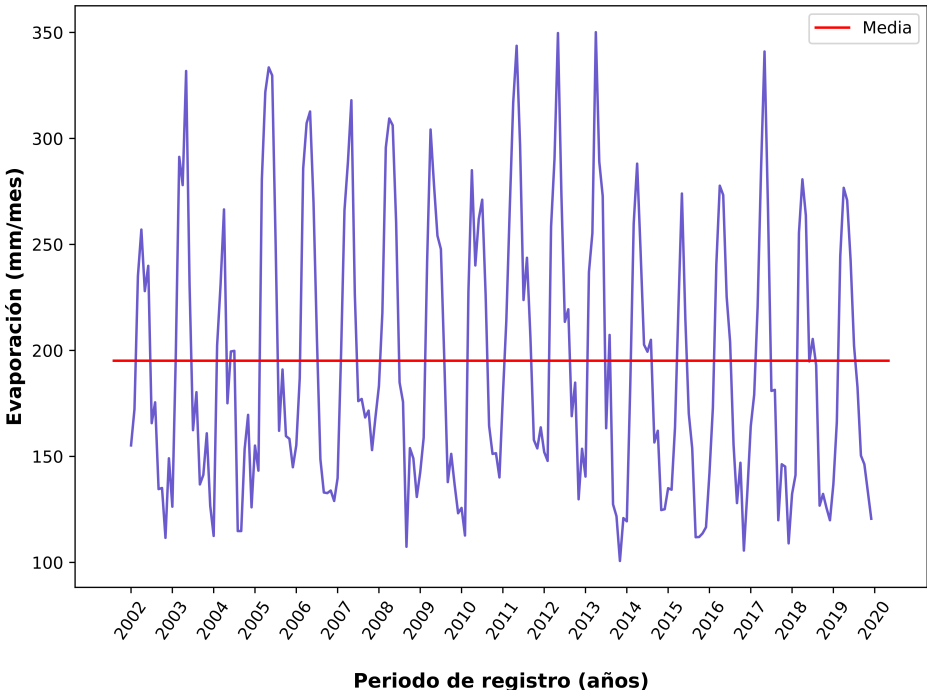


Fig. 2. Comportamiento de la evaporación en la estación: Calera.

Al-Mukhtar [4], realizó una comparación de seis modelos de ML en tres diferentes regiones Bagdad, Basora y Mosul, de Irak, los cuales son regresión condicional de bosques aleatorios (Cforest), regresión spline adaptativa multivariada (MARS), regresión bagged splines multivariada adaptativa (BaggedMARS), modelo de árboles de decisión (M5), k-Vecinos más cercanos (KNN) y k-vecinos más cercanos ponderado (KKNN).

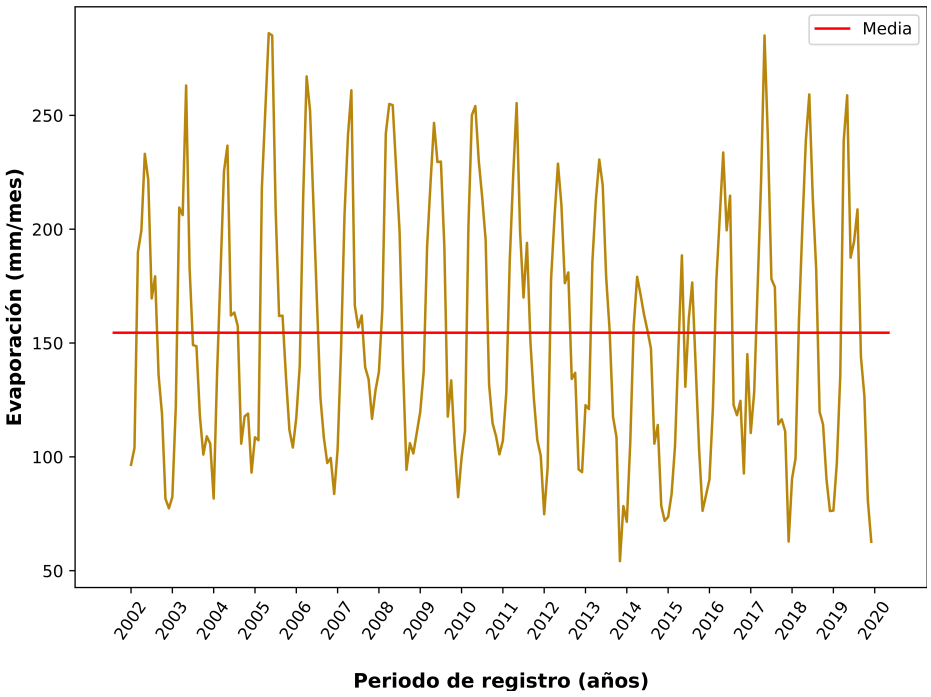


Fig. 3. Comportamiento de la evaporación en la estación: Fresnillo.

Los resultados del análisis dieron a conocer que los modelos KNN y M5 fueron los mejores en términos de capacidad predictiva para modelar las tasas de evaporación, comprobando la buena eficiencia que tienen los modelos de ML.

Shabani [15], implementó 4 diferentes modelos de ML para estimar la evaporación en la Provincia de Golestan, al sureste del mar de Caspian, tales algoritmos son el proceso de regresión Gaussiano (GPR), regresión de máquinas de soporte vectorial (SVR), KNN y bosques aleatorios (RF). Los resultados del estudio indican que bajo las condiciones del análisis el mejor modelo fue el GPR, teniendo ligeramente mejor desempeño que los demás modelos.

El objetivo de esta investigación es realizar una comparación del comportamiento de la evaporación a través de los modelos de ML regresión lineal múltiple (MLR), bosques aleatorios (RF) y k-vecinos más cercanos (KNN), evaluando su desempeño bajo las métricas coeficiente de correlación de Pearson (R), coeficiente de eficiencia Nash-Sutcliffe (NSE), raíz del error cuadrático medio (RMSE) y el error medio absoluto (MAE).

2. Zona de estudio

La región Calera se localiza en la porción central del estado de Zacatecas; entre los paralelos $22°41'$ y $23°24'$ de latitud norte y entre los meridianos $102°33'$ y $103°01'$ de longitud oeste, cubriendo una superficie aproximada de 2,226 km^2.

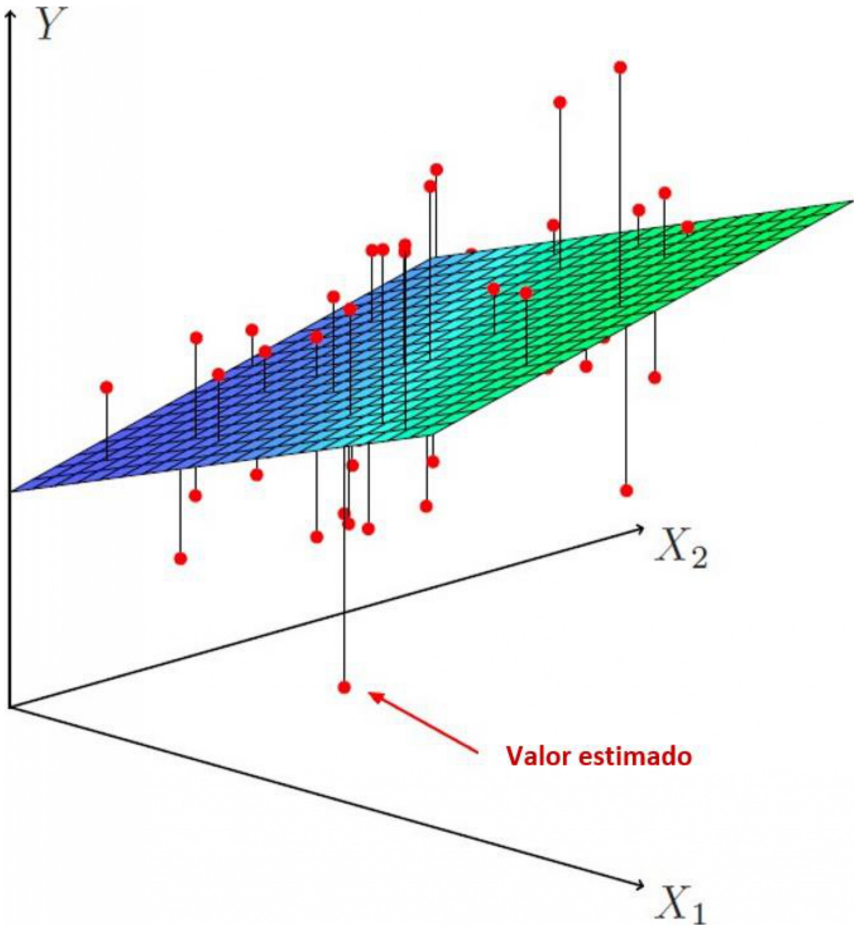


Fig. 4. Regresión lineal múltiple.

De acuerdo con la clasificación de Köppen, modificada por Enriqueta García en 1964 para las condiciones de la República Mexicana [7], en la mayor superficie de la región prevalece el clima semi-seco templado BS1kw, clima seco estepario (BS), que corresponde con el más seco de este tipo de climas, subtipo semi-seco (tipo 1).

Se caracteriza por presentar una temperatura media anual que varía entre $18°C$ y $22°C$, la temperatura media del mes más frío es menor de $18°C$, con invierno fresco y régimen de lluvias en verano. Con los registros obtenidos para el periodo 1980-2009, utilizando el método de isoyetas e isotermas, se determinaron valores de precipitación, temperatura y evaporación potencial media anual de 425 mm, $16.3°C$ y 2,263 mm, respectivamente [5].

Las estaciones climátológicas automáticas analizadas son CEZAC y El Pardillo 3, ubicadas en el municipio de Calera y Fresnillo, así como, las estaciones convencionales ubicadas en estos municipios. En la Tabla 1 se muestran las coordenadas de las estaciones climatológicas.

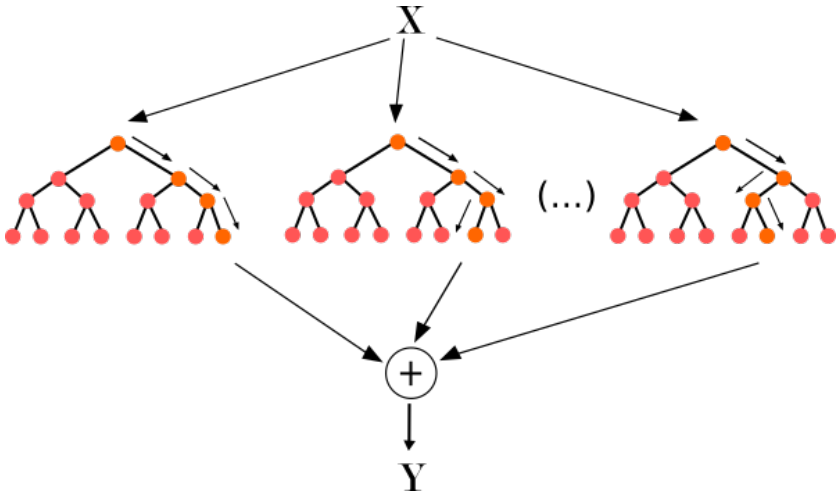


Fig. 5. Bosques aleatorios.

La evaporación se estimó a través del método del tanque evaporímetro Clase A, el cual consiste en un tanque de 120.7 cm de diámetro y 25 cm de profundidad, que debe ser colocado a 5 cm de la superficie del suelo, montado sobre una base de madera que permita la circulación del aire, el material del tanque debe ser de acero resistente a la corrosión, la medición se realiza mediante un micrómetro que indica cuanta cantidad de agua ha evaporado en un intervalo de tiempo y se registra en mm [14]. En la Figura 1 se muestra la ubicación de las estaciones climatológicas.

3. Materiales y métodos

La información climatológica fue proporcionada por el Instituto Nacional de Investigaciones Forestales, Agrícolas y Pecuarias (INIFAP) [9, 14], mediante la Red de Monitoreo Agroclimático del Estado de Zacatecas, las cuales tienen un periodo de registro de 18 años, de enero de 2002 a diciembre de 2019, comprenden los parámetros meteorológicos temperatura media máxima ($T_1, ^\circ C$), temperatura media mínima ($T_2, ^\circ C$), temperatura media ($T_3, ^\circ C$), precipitación (P, mm), humedad relativa media máxima ($HR_1, \%$), humedad relativa media mínima ($HR_2, \%$), humedad relativa media ($HR_3, \%$), radiación solar ($RS, W/\mathrm{m}^2$), velocidad del viento media máxima ($V_1, \mathrm{m}/\mathrm{s}$), y velocidad del viento media ($V_2, \mathrm{m}/\mathrm{s}$).

Por otro lado, mediante la Comisión Nacional del Agua (CONAGUA), a través del Sistema de Información Hidrológica (SIH) se obtuvieron los registros de evaporación (EP, mm) de manera mensual, con el mismo periodo de registro que las estaciones automáticas [6, 14].

En la Tabla 2 se presenta el análisis estadístico de cada variable, para la estación Calera (C) y Fresnillo (F), donde se muestra el valor mínimo, máximo, medio y desviación estandar, en las Figuras 2 y 3 se muestra el comportamiento de la evaporación (EP, mm) a través de los años.

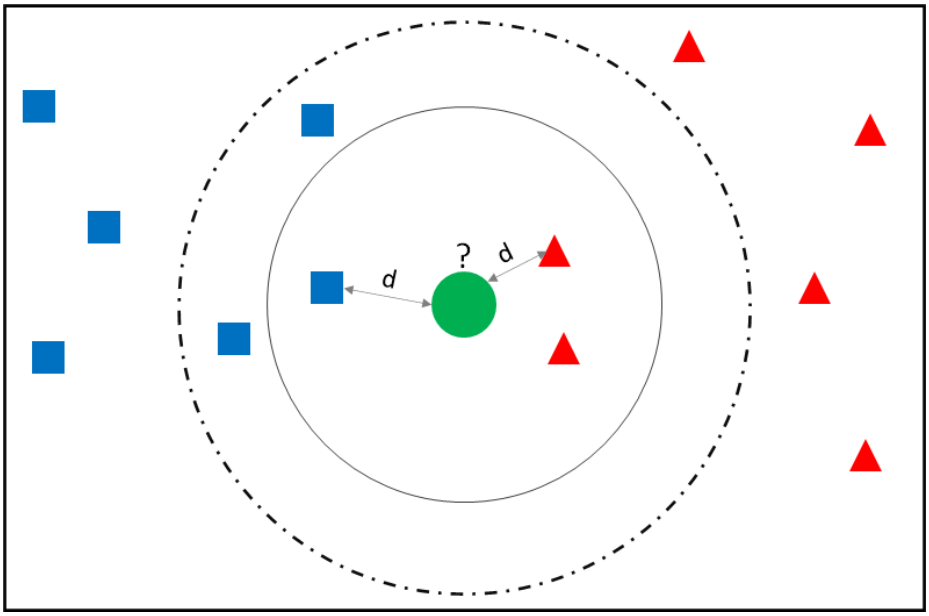


Fig. 6. K-vecinos más cercanos.

3.1. Regresión lineal múltiple (MLR)

La regresión lineal múltiple (MLR, por sus siglas en inglés), es un modelo que ha sido ampliamente utilizado como una herramienta fundamental para modelar relaciones entre variables de manera lineal, y ha sido utilizada en diferentes estudios [3]. MLR establece una relación cuantitativa entre las variables dependientes e independientes, puede ser definido de la siguiente manera:

$$\hat{Y} = b_o + \sum_{i=1}^{n} b_i x_i, \tag{1}$$

donde $\hat{Y}$ es el valor predicho o esperado de la variable dependiente, b_o es el valor de $\hat{Y}$ cuando todos los valores de las variables independientes son iguales a cero, $x_i (i = 1, ..., n)$ son los valores de cada uno de los valores muestra, y $b_i (i = 1, ..., n)$ son los coeficientes (pesos) estimados por la regresión, como se muestra en la Ecuación 1, en la Figura 4 se muestra gráficamente la representación de un modelo de regresión lineal múltiple en tres dimensiones.

3.2. Bosques aleatorios (RF)

Bosques aleatorios (RF, por sus siglas en inglés), es un algoritmo de aprendizaje prominente para problemas de clasificación y regresión. Desde un conjunto de datos único, un número diferente de árboles son construidos de forma aleatoria para el proceso inicial de construcción de árboles total.

Tabla 3. Correlación de las variables con la EP.

Variable	Correlación	
	C	**F**
T_1	0.78	0.84
T_2	0.33	0.49
T_3	0.64	0.76
P	**-0.24**	**-0.06**
HR_1	-0.66	-0.52
HR_2	-0.61	-0.42
HR_3	-0.66	-0.50
RS	**0.85**	**0.89**
V_1	0.66	0.52
V_2	0.55	0.52

Tabla 4. Variables seleccionadas.

Variable	MLR		RF		KNN	
	C	**F**	**C**	**F**	**C**	**F**
T_1	✓				✓	✓
T_2	✓	✓	✓		✓	✓
T_3	✓		✓	✓	✓	✓
P	✓			✓		
HR_1		✓	✓	✓		
HR_2		✓				
HR_3	✓		✓	✓	✓	✓
RS	✓	✓	✓	✓		
V_1	✓		✓	✓		✓
V_2					✓	

La predicción de un bosque aleatorio es entonces, el cálculo del promedio general de las predicciones de todos los árboles. En este algoritmo cada árbol se construye utilizando una mestra de tamaño a_n del conjunto de datos, estos datos solo son utilizados para construir la partición de árboles y posteriormente realizar las predicciones.

Una vez que la observación es seleccionada, el algoritmo forma un control de entrenamiento utilizando cuadrículas o un método aleatorio para la búsqueda. En cada celda, un número de variables de prueba es seleccionada, después son escogidos el número de árboles y los nodos máximos. Matemáticamente el modelo RF se puede representar de la siguiente manera:

$$m_{M,n}(x, \theta_1, ..., \theta_M, D_n) = \frac{1}{M} \sum_{m=1}^{M} m_n(x, \theta_M, D_n), \tag{2}$$

donde $m_n(x, \theta_M, D_n)$ es el valor predicho en el punto x dado por el j-ésimo árbol. $\theta_1, ..., \theta_M$ son las variables aleatorias independientes, distribuidas como una variable aleatoria θ, independiente de la muestra D_n. En la Figura 5 se muestra un esquema de cómo se genera la decisión de cada árbol para realizar una decisión final.

Bosques aleatorios no solo otorga confianza al momento de predecir el modelado, también ayuda en la importancia de la medición, la cual puede ser utilizada para reducir las variables sin perder ninguna información importante.

En el método de bosques aleatorios de Breiman, la variable importante es calculada de la siguiente manera: en primer lugar, se calcula la tasa de error original o el error cuadrático medio de cada árbol formado y también se realiza el mismo experimento con los datos originales con una variable permutada.

Después, se toma la diferencia entre estas dos tasas de error, la nueva medida es la diferencia media de los datos generales dividida por el error estándar de estas diferencias. Esta importante medición de la variable podría usarse para seleccionar un subconjunto de las características más importantes que tienen un gran impacto en la predicción de la variable objetivo [12].

Tabla 5. Resultado de métricas de evaluación.

Modelo	Métricas			
	RMSE(mm)	**MAE(mm)**	**NSE**	R
MLRC	**15.97**	12.56	0.93	0.97
MLRF	**20.53**	14.66	0.87	0.94
RFC	19.46	15.23	0.89	0.95
RFF	21.84	15.78	0.85	0.93
KNNC	18.38	13.36	0.90	0.95
KNNF	24.403	18.269	0.82	0.90

3.3. K-vecinos más cercanos (KNN)

El modelo de los k-vecinos más cercanos (KNN, por sus siglas en inglés), es una técnica no paramétrica que ha sido ampliamente utilizada en el campo de la regresión y clasificación de aprendizaje supervisado.

El fundamento del concepto de regresión KNN es estimar las densidades de probabilidad y funciones de regresión a través de los promedios locales ponderados de la función dependiente. Eso se debe lograr junto con la estimación de la probabilidad condicional basada en los k-vecinos más cercanos de la probabilidad condicional del vector x. La función de densidad utilizada por el modelo KNN se estima de la siguiente manera:

$$f_{NN(x)} = \frac{k/n}{V_{k(x)}} = \frac{k/n}{C_d r_k^d(x)}, \tag{3}$$

donde k es el número de vecinos más cercanos, d son las dimensiones del espacio del vector, C_d es el volumen unitario de la esféra en d dimensiones, $r_{k(x)}$ es la distancia Euclidiana hacia el k-ésimo valor del punto más cercano, y $V_{k(x)}$ es el volumen de la esfera d-dimensional con radio $r_{k(x)}$.

La elección de los k patrones en las observaciones son determinadas con base en la probabilidad del vector condicional utilizando la distancia Euclidiana. En el modelo KNN en todas las variables predictoras se asume tener la misma importancia al momento de estimar la probabilidad condicional [4]:

$$\xi_{t,i} = \sqrt{\sum_{1}^{m} \{S_j x_{j,i} - x_{j,t}\}^2}. \tag{4}$$

Para la estimación de la distancia Euclidiana se tiene la Ecuación 4, donde x es un vector con m predictores $x_{j,i}$ y S_j es el factor ponderado de escala para el j-ésimo predictor. Una vez que se estima la distancia Euclidiana para cada vector característico proyectado, se ordena de manera ascendente.

Así, un conjunto de K-NN casos es seleccionado para que un elemento del conjunto de registro en un tiempo t, se asocie el estado histórico más cercano con el vector actual [4]. En la Figura 6 se observa cómo dependiendo de los vecinos más cercanos, el nuevo valor se asocia a determinado conjunto, esto de acuerdo con la distancia Euclidiana y el número de elementos que tienen características similares.

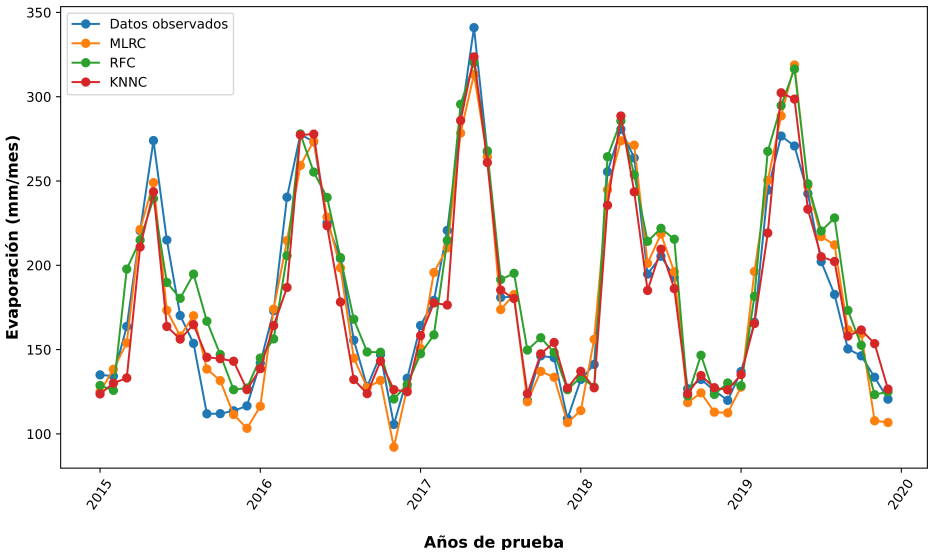


Fig. 7. Modelos estación: Calera.

3.4. Métricas de evaluación

Para evaluar la precisión de los modelos, fueron utilizadas cuatro funciones, coeficiente de correlación de Pearson (R), coeficiente de eficiencia Nash-Sutcliffe (NSE), la raíz del error cuadrático medio (RMSE) y el error medio absoluto (MAE). Las funciones mencionadas están definidas de la siguiente manera:

$$\mathrm{R} = \frac{\sum_{i=1}^{n}(o_i - \bar{o})(p_i - \bar{p})}{\sqrt{\sum_{i=1}^{n}(o_i - \bar{o})^2 \sum_{i=1}^{n}(p_i - \bar{p})^2}}, \tag{5}$$

$$\mathrm{NSE} = 1 - \frac{\sum_{i=1}^{n}(o_i - p_i)^2}{\sum_{i=1}^{n}(o_i - \bar{o})^2}, \tag{6}$$

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n}(p_i - o_i)^2}{n}}, \tag{7}$$

$$\mathrm{MAE} = \frac{\sum_{i=1}^{n}|p_i - o_i|}{n}, \tag{8}$$

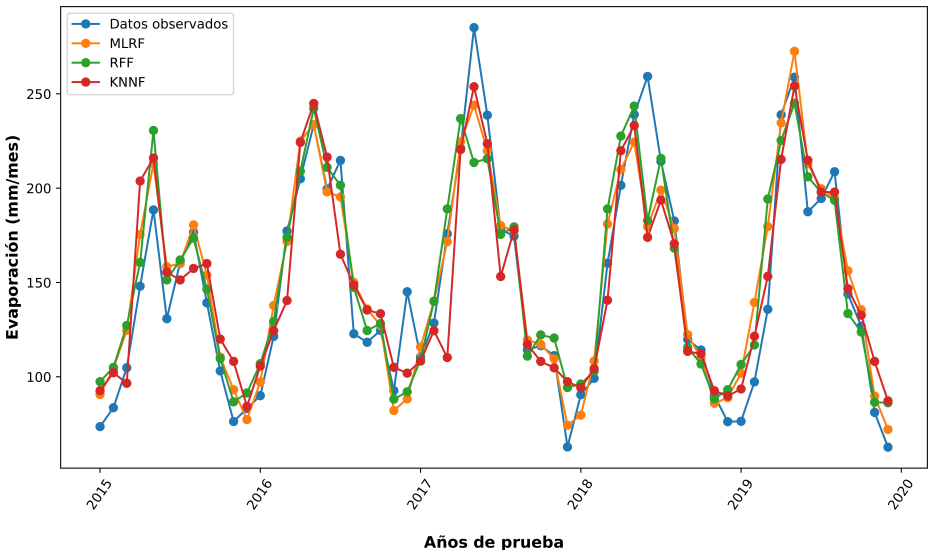


Fig. 8. Modelos estación: Fresnillo.

donde p_i es el i-ésimo valor predicho por los modelos, o_i es el i-ésismo valor observado, y n es un número entero que representa el número total de los datos de la muestra, $\bar{o}$ y $\bar{p}$ son el promedio de los valores observados y predichos, respectivamente. Se utilizó el lenguaje de programación Python en su versión 3.6, en el cual se realizaron los análisis estadísticos, la implementación de los modelos, cálculo de las métricas de evaluación y la elaboración de gráficos.

4. Resultados

Se realizó la comparación de los modelos MLR, RF y KNN, para predecir la EP en dos diferentes estaciones climatológicas, Calera (C) y Fresnillo (F), Zacatecas. A partir de los registros mensuales del periodo 2002-2019, se tomaron los primeros 13 años para entrenamiento y los últimos 5 años para la fase de prueba, con base en aplicar el muestreo aleatorio simple [17].

Se evaluó cada modelo mediante una selección de características exhaustiva, la cual consiste en realizar todas las combinaciones posibles dentro de un conjunto de datos, siempre y cuando la cantidad de variables y muestras lo permitan [11].

Dentro del análisis estadístico, la variable que presentó mayor correlación para ambas estaciones fue la radiación solar ($RS, W/\,\mathrm{m}^2$), con un valor de 0.85 para la estación Calera y 0.89 para Fresnillo, seguido de la temperatura media máxima ($T_1, ^\circ C$) con un valor de 0.78 y 0.84, respectivamente, por otro lado, el parámetro que presentó menor correlación fue la precipitación (PE, mm), con un valor de -0.24 para Calera y -0.06 para Fresnillo, como se muestra en la Tabla 3.

La selección de variables de entrada se realizó tomando como referencia la metrica RMSE, generando los modelos que se muestran en la Tabla 4, los cuales son MLRC, RFC y KNNC, para la estación de Calera, y MLRF, RFF y KNNF, para Fresnillo.

La comparación de los diferentes modelos con respecto a los valores observados, indica una variación espacio-temporal de la evaporación, teniendo un comportamiento diferenciado en primavera-verano y otoño-invierno, los valores con menor ajuste se tienen en la segunda estación (Figuras 7 y 8).

En la Tabla 5 se muestra el resultado de las métricas de evaluación, el mejor rendimiento lo obtuvo el modelo MLR, con R de 0.97 y 0.94, NSE de 0.93 y 0.87, RMSE de 15.97 y 20.53 mm, y MAE de 12.56 y 14.66 mm utilizando como variables climatológicas $T_1, T_2, T_3, P, HR_3, RS$ y V_1; T_2, HR_1, HR_2, y RS, para las estaciones Calera y Fresnillo, respectivamente.

5. Conclusiones

En la presente investigación se realizó una comparación del comportamiento de la evaporación a través de tres modelos de ML, los cuales fueron MLR, RF y KNN, para la región Calera perteneciente al estado de Zacatecas, México. Las técnicas utilizadas bajo las métricas R, NSE, RMSE y MAE, para las estaciones Calera y Fresnillo mostraron que el mejor modelo fue regresión lineal múltiple (MLR), con R de 0.97 y 0.94, NSE de 0.93 y 0.87, RMSE de 15.97 y 20.53 mm, y MAE de 12.56 y 14.66 mm, respectivamente.

La diferencia entre la evaporación estimada y la medida en el tanque se considera significativamente alta, sin embargo, se recomienda para futuros estudios realizar un análisis detallado de la información del tanque e implementación de otras técnicas de ML bajo diversos escenarios, para un mejor comportamiento del proceso de evaporación.

Agradecimientos. Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por la beca otorgada mediante la convocatoria “Becas Nacional (Tradicional) 2022 - 1” para la realización de la Maestría en Ciencias del Procesamiento de la Información (MCPI), así como al Instituto Nacional de Investigaciones Forestales, Agrícolas y Pecuarias (INIFAP) y a la Comisión Nacional del Agua (CONAGUA), por la información proporcionada.

Referencias

1. Abed, M., Imteaz, M. A., Ahmed, A. N., Huang, Y. F.: Application of long short-term memory neural network technique for predicting monthly pan evaporation. Scientific Reports, vol. 11, no. 1 (2021) doi: 10.1038/s41598-021-99999-y
2. Aghelpour, P., Bagheri-Khalili, Z., Varshavian, V., Mohammadi, B.: Evaluating three supervised machine learning algorithms (LM, BR, and SCG) for daily pan evaporation estimation in a semi-arid region. Water, vol. 14, no. 21, pp. 3435 (2022) doi: 10.3390/w14213435

3. Al-Ghobari, H. M., El-Marazky, M. S., Dewidar, A. Z., Mattar, M. A.: Prediction of wind drift and evaporation losses from sprinkler irrigation using neural network and multiple regression techniques. Agricultural Water Management, vol. 195, pp. 211–221 (2018) doi: 10.1016/j.agwat.2017.10.005
4. Al-Mukhtar, M.: Modeling of pan evaporation based on the development of machine learning methods. Theoretical and Applied Climatology, vol. 146, no. 3-4, pp. 961–979 (2021) doi: 10.1007/s00704-021-03760-4
5. CONAGUA: Actualización de la disponibilidad media anual de agua en el acuífero Calera (3225) estado de Zacatecas (2020)
6. CONAGUA: Comisión nacional del agua. sih.conagua.gob.mx/ (2020)
7. García, E.: Modificaciones al sistema de clasificación climática de Köppen (2004)
8. Ghazvinian, H., Karami, H., Jun, C., Francis, O., Bateni, S. M., DadrasAjirlou, Y., Band, S.: Laboratory comparison of evaporation rate between Colorado sanken evaporation pan and class a evaporation pan (case study: Semnan, Iran). (2023) doi: 10.20944/preprints202302.0165.v1
9. INIFAP: Instituto nacional de investigaciones forestales, agrícolas y pecuarias. zacatecas.inifap.gob.mx/ (2022)
10. Malik, A., Rai, P., Heddam, S., Kisi, O., Sharafati, A., Salih, S. Q., Al-Ansari, N., Yaseen, Z. M.: Pan evaporation estimation in Uttarakhand and Uttar Pradesh states, India: Validity of an integrative data intelligence model. Atmosphere, vol. 11, no. 6, pp. 553 (2020) doi: 10.3390/atmos11060553
11. Mosre, J., Suárez, F.: Actual evapotranspiration estimates in arid cold regions using machine learning algorithms with in situ and remote sensing data. Water, vol. 13, no. 6, pp. 870 (2021) doi: 10.3390/w13060870
12. Rakhee, Singh, A., Mittal, M., Kumar, A.: Predictive modeling of pan evaporation using random forest algorithm along with features selection. In: 10th International Conference on Cloud Computing, Data Science and Engineering (Confluence), pp. 380–384 (2020) doi: 10.1109/confluence47617.2020.9057856
13. Rezaie-Balf, M., Kisi, O., Chua, L. H. C.: Application of ensemble empirical mode decomposition based on machine learning methodologies in forecasting monthly pan evaporation. Hydrology Research, vol. 50, no. 2, pp. 498–516 (2018) doi: 10.2166/nh.2018.050
14. Secretaría de Economía: Proyecto de norma méxicana. Estaciones meteorológicas, climatológicas e hidrológicas, Parte 3: Condiciones de operación y mantenimiento (2021)
15. Shabani, S., Samadianfard, S., Sattari, M. T., Mosavi, A., Shamshirband, S., Kmet, T., Várkonyi-Kóczy, A. R.: Modeling pan evaporation using Gaussian process regression k-nearest neighbors random forest and support vector machines; comparative analysis. Atmosphere, vol. 11, no. 1, pp. 66 (2020) doi: 10.3390/atmos11010066
16. Sudani, Z. A. A., Salem, G. S. A.: Evaporation rate prediction using advanced machine learning models: A comparative study. Advances in Meteorology, vol. 2022, pp. 1–13 (2022) doi: 10.1155/2022/1433835
17. Verma, S. P.: Estadística básica para el manejo de datos experimentales: Aplicación en la geoquímica: (Geoquimiometría). Universidad Nacional Autónoma de México (2005)
18. Yaseen, Z. M., Al-Juboori, A. M., Beyaztas, U., Al-Ansari, N., Chau, K. W., Qi, C., Ali, M., Salih, S. Q., Shahid, S.: Prediction of evaporation in arid and semi-arid regions: A comparative study using different machine learning models. Engineering Applications of Computational Fluid Mechanics, vol. 14, no. 1, pp. 70–89 (2019) doi: 10.1080/19942060.2019.1680576

ISSN 1870-4069

Un estudio empírico de los fotomosaicos

Héctor Benítez Pérez[1], Manuel López Michelone[2]

[1] Universidad Nacional Autónoma de México,
México

[2] Universidad Nacional Autónoma de México,
Instituto de Investigación en Matemáticas Aplicadas y Sistemas,
México

hector.benitez@iimas.unam.mx, morsa@la-morsa.com

Resumen. Un fotomosaico es una imagen compuesta por muchas otras pequeñas imágenes dispuestas de tal manera que, a cierta distancia, forman una imagen más grande y reconocible. Cada pequeña fotografía es seleccionada cuidadosamente para que tenga un color y una textura similares a la parte de la imagen original que representa. Más allá de su valor en los campos de entretenimiento y arte, la creación de los fotomosaicos involucra una metodología para crearlos, así como requisitos mínimos necesarios para un resultado visual aceptable. En este artículo proponemos describir cómo se crea un fotomosaico así como plantear algunos criterios como repetición de imágenes, entropía y blending (que serán representados por índices numéricos), para intentar valorar cuando un fotomosaico es mejor que otro. Los resultados obtenidos pueden considerarse una primera aproximación a estos criterios.

Palabras clave: Fotomosaicos, entropía, algoritmos, métricas.

An Empirical Study of Photomosaics

Abstract. A photomosaic is an image composed of many small images arranged in such a way that, at a certain distance, they form a larger, more recognizable image. Each small photograph is carefully selected so that it has a similar color and texture to the part of the original image it represents. Beyond their value in the fields of entertainment and art, the creation of photomosaics involves a methodology to create them, as well as minimum requirements necessary for a result visually acceptable. In this article we propose to describe how a photomosaic is created as well as to propose some criteria such as repetition of images, entropy and blending (which will be represented by numerical indices), to try to assess when a photomosaic is better than another. The results obtained can be considered a first approach to these criteria.

Keywords: Photomosaics, algorithms, metrics, entropy.

1. Introducción

Un fotomosaico es una imagen compuesta por muchas fotografías pequeñas que se combinan para crear una imagen más grande. La técnica de crear un fotomosaico implica dividir la imagen original en pequeñas secciones y luego reemplazar cada sección con una fotografía diferente que tenga colores y tonos similares. El resultado final es una imagen digital compuesta por cientos o incluso miles de fotografías individuales que se combinan para crear una imagen única y coherente [9].

El crédito por la idea de los fotomosaicos probablemente sea para Robert Silvers, quien desarrolló esta técnica cuando era estudiante en el MIT [4]. Silvers finalmente patentó su proceso y fundó Runaway, una empresa dedicada a generar fotomosaicos [7]. Hay algunos elementos que nos permiten mostrar cuando un fotomosaico está bien hecho:

- Las imágenes que se utilizan para crear el fotomosaico deben ser de alta calidad y tener una resolución adecuada para que se vean bien cuando se reduzcan de tamaño.
- La imagen resultante debe ser clara y nítida, y no debería haber distorsiones ni errores visibles (artefactos), en las imágenes que la componen.
- El fotomosaico debe ser coherente y tener un aspecto uniforme, lo que significa que las imágenes que lo componen deben tener un tamaño y una orientación similares.
- El fotomosaico debe ser reconocible como la imagen original a la que se ha aplicado el efecto. Es decir, la imagen resultante debe tener una forma y un aspecto general similares al de la imagen original.
- La elección de las imágenes que componen el fotomosaico debe ser coherente con el tema o la temática de la imagen original.

La calidad visual de un fotomosaico es algo que no se ha analizado formalmente. Y aunque la idea ya tiene algunos años, parece evidente que la calidad de un fotomosaico depende en parte de las propiedades del ojo humano para ver colores e imágenes.

Aun así, hay muchos puntos de interés sobre los criterios que se deben utilizar para seleccionar las imágenes específicas como parte del proceso de construcción del fotomosaico. Algunas de las preguntas más relevantes serían:

- ¿Cómo se puede definir algo similar a una distancia de color entre una parte de la imagen original y la colección de imágenes que se utilizarán en el fotomosaico?
- ¿Cuántas fotografías debe tener una biblioteca para construir un fotomosaico?
- ¿Cuántas repeticiones de imágenes puedes tener antes de que el fotomosaico parezca monótono?

La clave para comprender estos problemas se analiza en el artículo El reconocimiento de rostros, de Leon Harmon [3] La primera imagen del artículo mencionado es el rostro de Abraham Lincoln, que se construyó a partir de una colección de mosaicos grises sólidos.

Harmon, un investigador de Bell Labs en la década de 1970, que se consideraba un ciberartista, escribió el programa Magna Dott, que probablemente fue la primera aplicación de un filtro mosaico a una foto digital. Lo que importa aquí, sin embargo, es el estudio de la redundancia de la imagen, que toma la esencia de la imagen con un filtro de mosaico simple.

De hecho, Salvador Dalí tomó la imagen de Lincoln y la usó en su pintura Gala mirando al mar. Si se observa la imagen, digamos desde 30 metros de distancia, se puede ver el rostro de Lincoln, pero al acercarse a la pintura,se observa la imagen de la esposa de Dalí. La imagen se conoce popularmente como Lincoln en Dalivisión.

2. Construyendo un fotomosaicos

Estas dos imágenes –la de Harmon y la de Dalí– son la clave del secreto de los fotomosaicos. En lugar de escribir un filtro de mosaico que cambie un área de la imagen original con un color sólido, es posible escribir un filtro de mosaico mejorado que cambie cada área de la imagen original con alguna imagen que contenga un color similar. Así, los elementos básicos de un programa que construye un fotomosaico son:

1. Tómese una foto escaneada (imagen fuente) para procesar.
2. Defina una malla de cuadrícula en la imagen fuente. Cada región rectangular en dicha cuadrícula define el tamaño de una de las fotos que se colocará en esa posición de la imagen.
3. Calcúlese el color promedio de cada cuadrado de la malla definida.
4. Encuéntrese la imagen más cercana, con el color promedio más similar al cuadro de la cuadrícula, y sustitúyala en esa celda.
5. Repítase el proceso para toda la cuadrícula.

Un sistema para construir fotomosaicos no es un único programa. Se requieren diferentes programas para realizar una variedad de tareas. Un primer programa calcula el color promedio de cada imagen de la biblioteca a usarse en el fotomosaico que va a ser creado. Una vez teniendo esta información, se procede a crear el fotomosaico con un programa que sigue los pasos ya mencionados. El resultado es un archivo que describe qué foto va en cada región del fotomosaico. Un tercer programa ensambla finalmente la imagen final.

3. Colección de imágenes y color promedio

Aunque el enfoque elegido es muy sencillo, está claro que se requieren decenas de miles de imágenes digitales de alta calidad (fotografías) para construir el fotomosaico. (Silvers indica que tiene una biblioteca con más de 100 mil imágenes escaneadas para este fin [6].) Nuestro programa original apenas usaba 6 mil imágenes fotográficas de alta calidad (color de 24 bits), aunque hoy contamos con más de 120 mil fotografías.

Para entender cómo encontrar el color promedio de una imagen, debemos decir que cada fotografía digital está formada por pixeles que contienen tres componentes de color –que van de 0 a 255 en valor– (Rojo, Verde y Azul), que hacen el esquema RGB. La combinación de valores en cada componente RGB nos da diferentes colores específicos. Para encontrar el color promedio de la foto, tomamos la suma de cada pixel de la foto y dividimos el resultado por la cantidad de pixeles utilizados:

$$\text{Color Promedio} = \sum_{1}^{n}\sum_{1}^{m}(R_{nm}G_{nm}B_{nm})/nm, \tag{1}$$

donde n y m es el ancho y alto respectivamente, de cada foto procesada. El programa que procesa todas las imágenes genera un archivo que contiene el color promedio para cada una de las imágenes de la biblioteca, separada en sus componentes RGB.

4. Métricas de color

En este punto tenemos el color promedio de cada una de las casi 6 mil imágenes de la colección. Ahora lo que necesitamos es un criterio que establezca la distancia mínima entre el color promedio de cada celda de la cuadrícula en la imagen de origen y las imágenes de la colección. La idea más simple fue usar la distancia euclidiana (de la geometría analítica), en términos del color promedio RGB[3] [8, 10]:

$$\Delta = \sqrt{(R_2 - R_1)^2 + (G_2 - G_1)^2 + (B_2 - B_1)^2}, \tag{2}$$

donde R_1, G_1 y B_1 es el color promedio de los componentes RGB de la celda en la cuadrícula de la imagen de origen y R_2, G_2, y B_2 es el color medio de una de las imágenes de la colección. La imagen mejor seleccionada será la que tenga la menor distancia después de aplicar esta última ecuación a todas las imágenes de la colección contra el color de la celda de la cuadrícula que se está analizando. A esto lo llamamos métrica de color.

Esta métrica en particular –a la que llamamos métrica euclidiana– parece suficiente para construir un fotomosaico de forma adecuada. Sin embargo, hay otra métrica interesante, basada en un estudio de Thiadmer Riemersma [4], quien analiza cómo el ojo humano percibe los colores. Implementando las ideas de Riemersma (a la que llamamos métrica Riemersma), hallamos que la distancia entre dos colores puede definirse de la siguiente manera:

$$\Delta = \sqrt{(2 + \bar{r}/256)(\Delta R)^2 + 4(\Delta G)^2 + (2 + (255 - \bar{r})/256)(\Delta B)^2}, \tag{3}$$

donde:

$$\bar{r} = (R_1 + R_2)/2, \tag{4}$$

$$\Delta R = R_1 - R_2, \Delta G = G_1 - G_2, \Delta R = B_1 - B_2. \tag{5}$$

[3] La distancia entre dos colores se refiere a la diferencia numérica entre los valores RGB (rojo, verde, azul) de dos colores. Existen varias formas de calcular este valor, como la distancia euclidiana o la distancia de Manhattan, que pueden dar resultados diferentes según los pesos dados a cada canal de color. El valor de la distancia se puede utilizar para comparar qué tan similares o diferentes son dos colores entre sí.

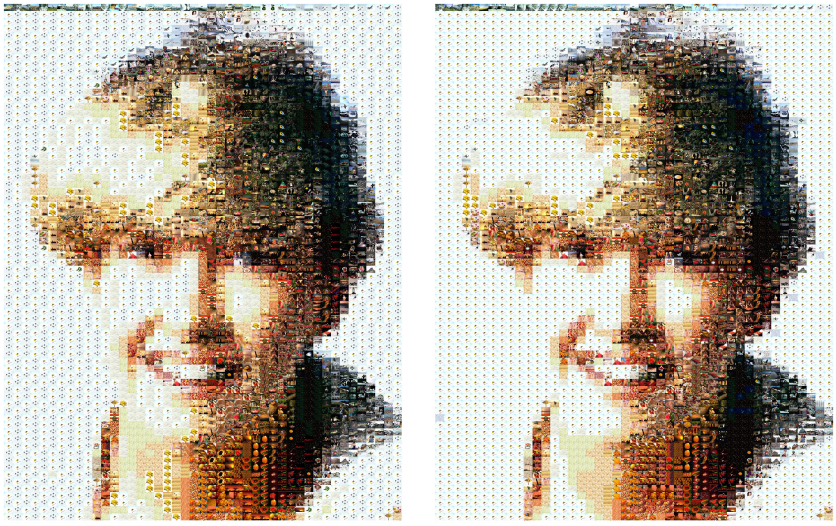


Fig. 1. Métrica euclidiana (izquierda) contra métrica Riemersma (derecha).

Riemersma tiene en cuenta la forma en que el ojo humano percibe el rojo, el verde y el azul. Con todos estos elementos, definimos es el algoritmo fundamental para producir un fotomosaico:

1. Se obtiene cada celda de la cuadrícula de la imagen de origen.
2. Se calcula el color promedio de esa celda en tiempo real.
3. Se encuentra la distancia mínima entre este número en comparación con el color promedio de cada imagen en la colección (usando alguna de las dos métricas definidas, métrica Euclidiana o Riemersma).
4. Se reemplaza la celda con la imagen elegida.
5. Se repite el proceso para cada celda de la cuadrícula de la imagen de origen.

5. Resultados

Escribimos todos los programas de computadora en Delphi 7 (Embarcadero), usando una computadora HP EliteDesk 705 (procesador AMD Ryzen 5 PRO de 6 núcleos), con 16 GB de memoria, con Windows 10, los cuales procesaron el color promedio de imágenes, la construcción del fotomosaico, el software que hace blending y el ensamble de la imagen final del mosaico en formato JPG. Esta es la descripción de los programas escritos:

– **Programa de pre–procesamiento de la colección de imágenes**. Este programa encuentra el color promedio de cada imagen. La salida de este software es un archivo de texto, que llamamos archivo indexado, que contiene en cada línea el color

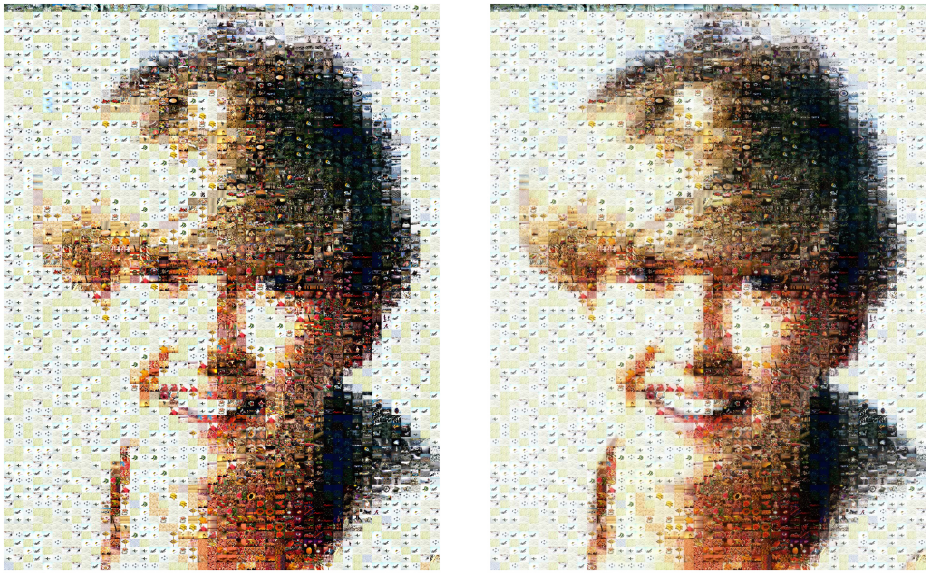


Fig. 2. Fotomosaico sin repetición con un 25 % de fusión (derecha).

promedio de cada imagen de la colección. Este conjunto de imágenes se procesa solo una vez, ya que el software de construcción de fotomosaicos lee este archivo para encontrar la mejor imagen para colocar en cada celda de la cuadrícula de imágenes (de acuerdo a algún criterio en particular).

– **Generador de fotomosaicos**. Una vez que se generado el índice de las fotografías a usar en un archivo, el programa que construye el fotomosaico. Se requiere la imagen fuente, la que se va a procesar y el nombre de un archivo que generará la descripción del fotomosaico a construir y el archivo de índice que contiene la descripción del color promedio de todas las imágenes de la biblioteca.
– **Programa para ensamblar el fotomosaico final**. El sistema genera un archivo de texto que contiene la descripción en qué región se coloca cada pequeña imagen. Normalmente, el software de creación de fotomosaicos necesita en promedio de dos a cuatro minutos para crear la descripción del fotomosaico que se va a construir.
– **Generador de imágenes**. Toma el archivo de texto de salida creado por el generador de fotomosaicos y ensambla la imagen final, que se puede guardar en formato JPEG. El archivo de salida es un conjunto de líneas que indican qué imagen colocar en cada posición de la cuadrícula. Esta se genera línea por línea.
– **Programa que analiza los índices de belleza del fotomosaico**. Este programa cuenta las imágenes usadas y permite fusionar la imagen original con el fotomosaico generado para ver si mejora visualmente.

El éxito de los resultados depende de la métrica ya que es en definitiva el criterio de elección de cada imagen que formará el fotomosaico. Las siguientes imágenes muestran los resultados con las dos métricas aplicadas, que no parecen ser radicalmente diferentes.

Fig. 3. Comparación en un fragmento de un fotomosaico contra la imagen original (blending de la imagen original al 35 %).

Un problema evidente es la repetición de las mismas imágenes, lo que hace que el resultado final sea monótono y poco deseable. Sin embargo, en este proceso observamos puntos importantes en la tecnología de creación de fotomosaicos:

– **Calidad de la imagen de origen**. No todas las imágenes son lo suficientemente buenas para ser procesadas por un software de fotomosaico. Las imágenes con más contraste son generalmente mejores. Obviamente, cuantos más colores tenga una imagen, mejor. Un equilibrio adecuado entre el contraste y el brillo es el mejor enfoque. Afortunadamente existen muchas aplicaciones que te permiten manipular imágenes en estos parámetros (Photoshop, ACDSee, Retriever, etcétera).
– **Colección de imágenes**. Para obtener mejores resultados, el software de fotomosaico debe tener acceso a una gran colección de imágenes de alta calidad (color de 24 bits), con una resolución promedio de al menos 800 x 600 píxeles. Las imágenes más pequeñas no dan buenos resultados en general.
– **Imágenes con muchos colores**. Es deseable que la colección de imágenes utilizadas sea numerosa y, además, que contenga imágenes con muchos colores. Por ejemplo, utilizamos la colección de 10 mil imágenes de Global Star Software (http://www.globalstarsoftware.com/) y descubrimos que, en comparación con las 6 mil imágenes originales utilizadas, las 10 mil contenidas tenían menos distribución de colores que nuestra colección original. ¿La moraleja de la historia? Una colección de imágenes con más balance de color es mejor que una gran cantidad de imágenes (demasiado oscuras o demasiado claras).
– **Las colecciones de imágenes**. Una gran colección de imágenes puede dar mejores resultados que una colección limitada. Se pueden encontrar colecciones de entre 30 mil y 50 mil imágenes de alta resolución –gratis– en archive.org[4]

[4] https://archive.org/search. php?query=imagen+colección&page=3

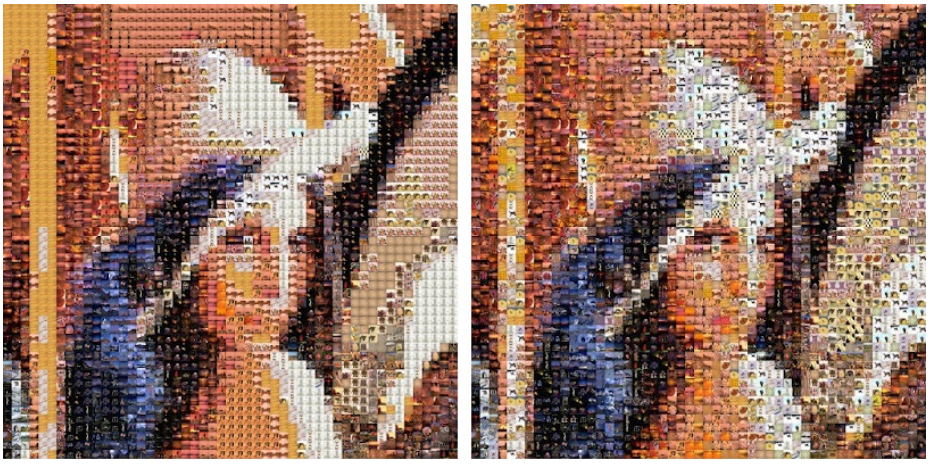


Fig. 4. Imagen de Lena con repetición (izquierda) y sin repetición de fotos.

5.1. Mejorando los fotomosaicos

Los fotomosaicos se pueden mejorar utilizando varias técnicas. Una de ellas es fusionar la imagen original encima de la imagen mosaico con un porcentaje bajo, lo que fusiona ambas imágenes. Esto se llama blending[5] [11].

La combinación mejora los fotomosaicos porque la métrica solo tiene en cuenta el color de cada celda de la cuadrícula y muchas veces no encontrará una imagen que contenga un color similar al de la celda original y, además, si hay sombras o características pintadas en la región, la métrica no los tiene en cuenta. Por tanto, la fusión del fotomosaico con la imagen original (el bendling) (aplicado ligeramente para que no se note demasiado) mejora el fotomosaico final.

Una idea utilizada por Silvers es no utilizar toda la imagen de la colección, sino un fragmento de la misma, buscando un mejor ajuste a la celda de la región analizada. Esta es una buena idea, pero se requiere mucho más procesamiento de imágenes. La repetición de las imágenes en el fotomosaico final genera monotonía.

Para evitar esto, se implementó la siguiente idea: se calcularon las 10 mejores imágenes para cada celda de la cuadrícula y se eligió una de ellas al azar. Por supuesto, esto no siempre le da a la imagen el color más cercano a la celda, pero elimina la monotonía y hace que el fotomosaico se vea mejor.

5.2. Entropía en la información de un fotomosaico

Para evaluar el grado de información transmitida en las regiones de fotografías que forman un fotomosaico podemos usar la entropía de información de acuerdo con Shannon [5], el cual evalúa el grado de información transmitida en las regiones del fotomosaico.

[5] Blending es el proceso de combinar dos o más imágenes en una sola imagen, de manera que el resultado contenga la información visual de todas las imágenes originales.

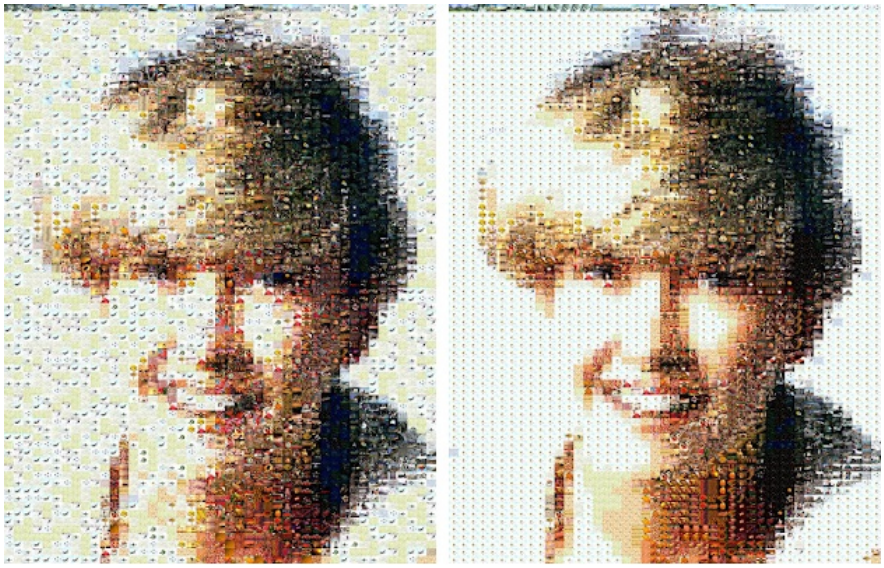


Fig. 5. Imagen sin repetición de fotos (izquierda) y sin repetición (derecha).

El concepto de entropía de la información describe cuanta incertidumbre (información) hay en una señal, en este caso una imagen. Por ejemplo, en una imagen en tonos de gris, con 256 posibles tonos, la entropía E puede calcularse como:

$$E = \sum_{i=0}^{255} p_i \log_2 p_i, \tag{6}$$

donde p_i es la probabilidad que un pixel al azar se elegido, teniendo una intensidad i [1]. En el contexto de una imagen en tonos de gris, la entropía calcula cuántos valores de pixel distintos hay en la imagen y cuánta variación hay en la distribución de esos valores. Si una imagen tiene una distribución uniforme de valores de pixel, entonces su entropía será alta, lo que significa que hay una gran cantidad de aleatoriedad presente.

Si una imagen tiene valores de pixel que se agrupan alrededor de un valor específico, entonces su entropía será baja, lo que significa que hay menos aleatoriedad presente. Por ejemplo, si tenemos una imagen I de color, podemos convertirla en tonos de gris y calcular su intensidad de la entropía.

Un valor de entropía mayor corresponde a una imagen o una región en el borde, mientras que un valor de entropía menor corresponde a un área de imagen suave, es decir, donde hay valores más homogéneos (de color), por ejemplo, cuando se aplica el blending [12, 2].

5.3. ¿Cúal es el mejor fotomosaico?

No hay ningún parámetro definitivo sobre cuando un fotomosaico es visualmente aceptable. Sin embargo, a pesar de que la belleza está en la mirada del observador, es posible definir una métrica en este sentido. Por supuesto que hay criterios variables.

Fig. 6. Imagen con repetición de fotografías (izquierda) y sin repetición (derecha) (en escala de grises).

El hecho de que haya pocas repeticiones debería ser un buen criterio teniendo en cuenta que cuanto más se repiten algunas imágenes, más monótono o simple es el resultado final. Otro criterio a tener en cuenta es usar la fusión de imágenes (blending), es decir, mezclar la imagen original en un porcentaje contra la imagen del fotomosaico, para suavizar la imagen final y no se vea tan pixelada.

Cuando se usa blending en un fotomosaico, el resultado es que baja la entropía de la imagen final. Cuando analizamos cuántas imágenes se repiten en un fotomosaico se puede encontrar un índice muy sencillo que puede indicar claramente si un fotomosaico es mejor que otro a los ojos del observador bajo la premisa de que muchas imágenes repetidas hacen el fotomosaico una imagen monótona.

En este caso lo que se hizo es leer el archivo que describe el fotomosaico, el cual se utiliza para crear la imagen final. Luego, una rutina cuenta cuántas veces se repitieron las imágenes. Los resultados se graficaron para tener una idea de cómo se ven. Se grafica del menor al mayor número de repeticiones.

En un fotomosaico ideal probablemente se tendrá una línea con pendiente cero, es decir, donde todas las imágenes aparecen en la misma cantidad. Pero dado que este caso es el ideal y altamente improbable, tomamos esto simplemente como la referencia teórica. Con estos datos podemos contar cuántas imágenes diferentes aparecen en el mosaico.

Para encontrar el índice, tomamos el número total de imágenes que debe tener el fotomosaico (tomado del mismo archivo de descripción) dividido por el número de imágenes diferentes. En la Figura 4, tenemos que el índice del fotomosaico de Lena (izquierda), es 4.644. En la imagen derecha el índice del fotomosaico es 3.737.

Para esta imagen se usó una colección de 10,019 fotografías. En la siguiente imagen (Figura 5), se tiene un índice de 3.91 para la imagen izquierda y 4.70 para la imagen derecha (5560 imágenes usadas).

5.4. Resultados usando solamente imágenes en tonos de gris

El modelo RGB permite imágenes en tonos de gris, donde $R = G = B$, es decir, los tripletes de Rojo, Verde y Azul tienen el mismo valor, por lo que solo hay 256 tonos de gris... $(0, 0, 0), (1, 1, 1), ..., (255, 255, 255)$. Está claro que la cantidad de imágenes disponibles puede cambiar el índice. Por lo tanto, tratamos de procesar una imagen (en tonos de gris), con una colección de 63.136 imágenes (con una repetición promedio de 246 imágenes con el mismo tono de gris en esa colección).

En el caso de permitir repetición, en la siguiente imagen (figura 6), el índice del fotomosaico fue 13.51. (imagen izquierda). Por otro lado, poniendo el criterio de elegir en cada imagen una de las 50 mejores, encontramos que el índice fue de 1.33. (imagen derecha). Un índice menor muestra menos repeticiones y por tanto, es menos monótono.

Con respecto a la entropía de las imágenes, para el caso de la figura 6, tenemos que el índice de entropía es de 7.9215 (izquierda) contra 7.9013 (derecha). De nuevo, la imagen con menos repeticiones presenta una entropía menor. Estos valores se obtuvieron usando el software MatLab, utilizando la biblioteca de procesamiento de imágenes.

6. Conclusiones

Algunos de los resultados más relevantes son:

– Una imagen sin permitir repeticiones hace un mejor fotomosaico, pero depende de cuantas imágenes consideremos para elegir la mejor. En las imágenes en tonos de gris, dado que solo podemos tener de 0 a 255 tonos diferentes, elegir uno entre los 50 mejores de entre más de 63 mil imágenes parece razonable. Cuanto menor sea el índice de repeticiones, mejor debe ser el fotomosaico.

– Las imágenes con repeticiones tienden a ser muy monótonas y tienden a tener índices más altos y visualmente son menos aceptables.

– El uso de blending suaviza el fotomosaico final y modifica favorablemente la entropía de la image (es decir, baja la entropía). El resultado final del fotomosaico es por ende más aceptable visualmente.

– El número total de imágenes de la colección es otro punto a tener en cuenta. Mientras más imágenes tiene una colección, la posibilidad de hacer un fotomosaico visualmente más aceptable, se incrementan.

– El uso de blending es una manera sencilla y económica para suavizar las imágenes procesadas y quitar en algún grado la pixelización del fotomosaico. Sin embargo, queda abierta la posibilidad de usar algunas técnicas de inteligencia artificial para hallar imágenes para las regiones del fotomosaico que contengan las sombras similares a las regiones en la foto a procesar.

Referencias

1. Farivar, R., Shokrollahi, A.: Adaptive photomosaic based on image entropy. Journal of Real-Time Image Processing, pp. 311–325 (2016)
2. González, R. C., Woods, R. E.: Digital image processing. Pearson Global Edition, pp. 545–549 (2017)
3. Harmon, L. D.: The recognition of faces. Scientific American, pp. 71–82 (1973)
4. Riemersma, T.: Colour metric (1999) www.compuphase.com/cmetric.htm
5. Shannon, C. E.: A mathematical theory of communication. Bell System Technical Journal (1948)
6. Silvers, R.: Photomosaics, Master Thesis. Massachusetts Institute of Technology (1996)
7. Silvers, R.: Photomosaics. Henry Holt and Co (1997)
8. Stricker, M. A., Orengo, M.: Similarity of color images (1995) doi: 10.1117/12.205308
9. Tran, N.: Generating photomosaics (1999) doi: 10.1145/298151.298213 https://doi.org/10.1145/298151.298213
10. Troebs, M.: Algorithms to create image montages (2000) pdfcoffee.com/algorithms-13-pdf-free.html
11. Wittenburg, T.: Alpha blending graphic images. DDJ - Doctor Dobbs Journal (1995)
12. Zhang, L., Ma, K. L., Yu, J.: Adaptively tiled image mosaics utilizing measures of color and region entropy. In: Proceedings of the 9th International Symposium on Visual Information Communication and Interaction (2016) doi: 10.1145/2968220.2968228

ISSN 1870-4069

Selección de metabolitos como características de un modelo de bosques aleatorios para el diagnóstico del COVID-19

Hugo Alexis Torres-Pasillas, José María Celaya-Padilla, Yamilé López-Hernández, Carlos Erick Galván-Tejada, Alejandra García-Hernández, Pedro Daniel Alaniz-Lumbreras, José Alejandro Morgan-Benita

Universidad Autónoma de Zacatecas, Unidad Académica de Ingeniería Eléctrica, México

ylopezher@conacyt.mx {hugo.tpasillas, jose.celaya, ericgalvan, alegarcia, dalaniz, alejandro.morgan}@uaz.edu.mx

Resumen. El COVID-19 es una enfermedad reciente que surgió a finales de 2019 causado por un nuevo tipo de coronavirus. A pesar de los avances en la investigación del virus y el desarrollo tanto de vacunas como de posibles tratamientos, el diagnóstico de la enfermedad, especialmente de forma temprana, continúa siendo una de las mejores herramientas para combatir la enfermedad y su transmisión. El objetivo de este estudio es seleccionar el mejor conjunto de metabolitos como potenciales biomarcadores para el diagnóstico, que son utilizados como características de un modelo de bosques aleatorios. Para ello, se utilizaron 4 diferentes técnicas de selección de características que son utilizadas con frecuencia dentro del Aprendizaje Automático, y un conjunto de datos que contiene mediciones de 110 metabolitos de 158 pacientes sospechosos de COVID-19 (121 enfermos y 37 sanos confirmados por pruebas rt-PCR). Los resultados muestran cuatro distintos conjuntos de metabolitos capaces de diagnosticar el COVID-19 con un alto desempeño en 6 distintas métricas utilizadas. El conjunto con mejor rendimiento en el conjunto de entrenamiento consta de 15 metabolitos y logra tener un desempeño alto en la validación a ciegas (f1=0.921, exactitud balanceada=0.875, AUC=0.910), mientras que el conjunto con menor número de características (5) obtiene el segundo mejor rendimiento en el conjunto de entrenamiento pero el mejor desempeño en la validación a ciegas (f1=0.931, exactitud balanceada=0.896, AUC=0.858).

Palabras clave: COVID-19, aprendizaje automático, metabolitos, selección de características, diagnóstico.

Selection of Metabolites as Features of a Random Forest Model for COVID-19 Diagnosis

Abstract. COVID-19 is a recent disease that emerged in late 2019 caused by a new type of coronavirus. Despite advances in virus research and the development of both vaccines and potential treatments, early and accurate

diagnosis of the disease remains one of the best tools to combat the disease and its transmission. The aim of this study is to select the best set of metabolites as potential biomarkers for diagnosis, which are used as features of a random forest model. To achieve this, four different feature selection techniques that are frequently used in Machine Learning, and a dataset containing measurements of 110 metabolites from 158 suspected COVID-19 patients (121 confirmed patients and 37 confirmed healthy by rt-PCR tests) were used. The results show four different sets of metabolites capable of diagnosing COVID-19 with high performance in six different metrics used. The set with the best performance in the training set consists of 15 metabolites and achieves high performance in blind validation (f1=0.921, balanced accuracy=0.875, AUC=0.910), while the set with the smallest number of features (5) obtains the second best performance in the training set but the best performance in blind validation (f1=0.931, balanced accuracy=0.896, AUC=0.858).

Keywords: COVID-19, machine learning, metabolites, feature selection, diagnosis.

1. Introducción

El Síndrome Respiratorio Agudo Severo 2 (SARS-CoV 2), es un tipo de coronavirus que surgió a finales de 2019 en la ciudad de Wuhan, en la providencia de Hubei, en China central, después de que un hospital notificó el ingreso de un paciente con neumonía grave e insuficiencia respiratoria el 26 de diciembre de 2019 [23].

La enfermedad causada por el SARS-CoV 2, denominada COVID-19, se caracteriza principalmente por síntomas que incluyen fiebre, tos seca y dificultad para respirar, además de síntomas menos comunes como vómito, diarrea, y dolor abdominal que aparecen dentro de los 2 a 14 días siguientes después del contagio [2].

Aunque la enfermedad se presenta mayormente sin síntomas o con síntomas de leves a moderados (en más del 80 % de los casos), en algunos casos estos pueden empeorar y requerir hospitalización y el uso de ventilación artificial, o hasta conducir a la muerte [13].

El día 11 de marzo de 2020, la Organización Mundial de la Salud (OMS) declaró la pandemia del COVID-19, y a marzo de 2023 continúa siendo una problemática actual, con una cifra de contagios que supera ya los 758 millones de individuos y que sigue aumentando a un ritmo de alrededor de 140 mil casos nuevos por día (tomado como el promedio de los últimos 7 días), con más de 6.8 millones de muertes acumuladas, de acuerdo con los datos reportados por la OMS [22].

Aunque ya se han estudiado y propuesto algunos medicamentos como el molnupiravir, la fluvoxamina y el paxlovid que reducen la mortalidad y la hospitalización en aproximadamente un 67 %, a día de hoy no existen tratamientos específicos contra esta enfermedad [19]. Por otro lado, se han desarrollado ya diferentes vacunas en contra de esta enfermedad que han mostrado buenos resultados, las cuales se han aplicado a nivel global desde finales de 2021, con eficiencias mayores al 90 % a 2 meses después de la primera dosis, y 60 % después de los 7 meses [16].

Sin embargo, estas se han encontrado con diferentes desafíos, como el acceso desigual a las diferentes vacunas y problemas de distribución, o la aparición de nuevas variantes del virus que reducen su eficiencia [3]. Por ello, el diagnóstico de la enfermedad sigue siendo una de las mejores herramientas, especialmente si se hace en fases tempranas del contagio.

Durante las últimas décadas, el área de la Inteligencia Artificial (IA), especialmente sus rama de Aprendizaje Automático (ML por sus siglas en inglés) y Aprendizaje Profundo (DL por sus siglas en inglés) han mostrado un gran potencial tanto en el área de la salud como en otras áreas, en aplicaciones como el diagnóstico de enfermedades, recomendación de tratamientos, predicciones de riesgo, entre otras [1], en muchas ocasiones mostrando desempeños incluso superiores a los métodos más clásicos o a los humanos [6].

Un enfoque reciente para el diagnóstico de enfermedades ha sido el uso de la metabolómica, el cual además puede ser utilizado para entender las interacciones del virus en el cuerpo a niveles moleculares [10]. Mediante este enfoque, se ha permitido encontrar biomarcadores para el diagnóstico y pronóstico de enfermedades como el Virus de la Inmunodeficiencia Uumana (VIH), hepatitis B y C, entre otros, así como a identificar rutas metabólicas que son alteradas debido a la presencia de los patógenos causantes de estas enfermedades [4].

En este trabajo estudiamos el uso de metabolitos como características para un modelo de Bosques Aleatorios (RF por sus siglas en inglés) para el diagnóstico de la enfermedad de COVID-19. Utilizando un conjunto de datos presentado por López Y., et al. [17], donde se cuantifican 110 metabolitos de 121 pacientes con diferentes niveles de severidad (agrupados de acuerdo a su nivel de severidad en 3 grupos: 2, 3 y 4) y 37 personas sanas (grupo 1).

Mediante el uso de las técnicas de selección de características de selección hacia adelante, Boruta, algoritmos genéticos y Regresión LASSO, se selecciona el conjunto de metabolitos con mejor desempeño en el conjunto de entrenamiento que consta de 110 individuos (obtenido por validación cruzada dejando uno fuera), y posteriormente se utiliza un conjunto de validación de 48 pacientes (12 por cada grupo).

El artículo está organizado de la siguiente manera: la sección 1 presenta una introducción al tema desarrollado, así como su justificación, y en la sección 2 presentamos los trabajos y resultados anteriores relacionados.

En la sección 3 introducimos los métodos utilizados tanto para la selección del conjunto de metabolitos como para la validación, y en la sección 4 presentamos los experimentos realizados y los resultados obtenidos. En la sección 5 presentamos las conclusiones del presente trabajo. Finalmente, la sección 6 corresponde a los agradecimientos.

2. Trabajos relacionados

Las primeras investigaciones del uso de ML y DL para el diagnóstico de COVID-19 se centraron en el uso de las imágenes médicas de Tórax analizadas por algoritmos de ML y DL.

En [20], por ejemplo, utilizan esta técnica y proponen un sistema de diagnóstico asistido por computadora con un alto radio de detección, proponiendo que el uso de ML para detectar y categorizar a pacientes de COVID-19 mediantes estas imágenes médicas es la mejor forma para diagnosticar el virus.

Además, otros estudios han utilizado técnicas de ML y DL para el diagnóstico de COVID-19 con alto desempeño, mediante el uso de características como síntomas [25], señales de audio [11], entre otras.

Por otra parte, diversos autores han realizado estudios sobre el cambio en los niveles de metabolitos debido a la presencia del COVID-19, y han sugerido su uso como biomarcadores para el diagnóstico y pronóstico de esta enfermedad, enfatizando además las ventajas de reducción del costo y el tiempo de respuesta de las pruebas en los laboratorios de microbiología y virología de diagnóstico [10].

En [21], Bardanzellu F., et al. mencionan que, en el futuro próximo, la combinación de la metabolómica, la microbiómica y el ML, a lo que denominan las 3 M's, serán una herramienta clave para el diagnóstico y pronóstico temprano del COVID-19, así como para realizar predicciones de riesgo, estratificación, manejo de pacientes, y toma de decisiones, conduciendo a la medicina de precisión.

Fraser D., et al. encontraron niveles de metabolitos alterados, y fueron capaces de construir un modelo de ML con una exactitud del 98 % para el pronóstico y un 100 % de exactitud para la predicción de fallecimiento [9], y aunque en este estudio se utilizó un número limitado de pacientes (30 pacientes divididos en 3 grupos), muestran el potencial para la combinación de la metabolómica y el ML para el COVID-19, sobrepasando la necesidad de pruebas rt PCR [21].

3. Métodos y materiales

3.1. Conjunto de datos

El conjunto de datos utilizado en este estudio fue recolectado por López, Y., et al. y se compone de 37 personas confirmadas negativas que se sospechaban enfermas de COVID-19 debido a su contacto con individuos infectados, y 121 pacientes confirmados positivos. En este, se cuantificaron de forma precisa 110 metabolitos mediante la metabolómica dirigida, así como 13 citocinas/quimiocinas. El proceso de recolección de datos se describe detalladamente en [17], y el conjunto de datos completo se encuentra disponible bajo la licencia CC by 4.0 en [1].

3.2. Aprendizaje automático

El aprendizaje automático (ML, por sus siglas en inglés) es una rama de la inteligencia artificial que busca encontrar funciones desconocidas, relaciones, o estructuras entre un conjunto de variables de entradas y salidas a partir de un conjunto de datos, que muchas veces son difíciles de encontrar por algoritmos explícitos [24], mediante modelos que se ajustan al conjunto de datos.

[1] data.mendeley.com/datasets/x9tw3knwsd

El proceso de aprendizaje se realiza al encontrar los parámetros del modelo a partir del conjunto de datos de entrenamiento. Dependiendo de cómo se lleve este proceso de aprendizaje, un algoritmo de ML puede ser supervisado, no supervisado, semi-supervisado o aprendizaje profundo [5].

El ML es una de las tecnologías más prometedoras para la clasificación [12], y comúnmente se realiza mediante aprendizaje supervisado: el conjunto de datos consta tanto de las características de cada elemento a clasificar, como de la clase a la que pertenece.

En este caso, el algoritmo busca clasificar a cada elemento en alguna categoría y, a partir de un conjunto de D características $\vec{x} = (x_1, x_2, \cdots, x_D)$ [5]. En particular, para una clasificación binaria, la etiqueta y pertenece al conjunto $\{0, 1\}$. Aunque existe una gran variedad de algoritmos de ML para la clasificación binaria, dos de ellos utilizados comunmente son:

- **Árboles de decisión.** Un árbol de decisión, como es descrito por Kingsford, C. & Salzberg, S. L. [14], es un algoritmo que clasifica a un elemento mediante una serie de preguntas sobre sus características. Cada pregunta corresponde a un nodo, y cada posible respuesta apunta a un nodo hijo. Por lo tanto las preguntas forman un árbol. El elemento es etiquetado a una clase siguiendo la ruta de preguntas desde el primer nodo, la raíz, a un nodo sin hijos, una hoja, de acuerdo a las respuestas en cada nodo. La clase asignada al elemento es la asociada a la hoja que alcanza.
- **Bosques aleatorios.** Los bosques aleatorios (RF, por sus siglas en inglés), son otro algoritmo de ML, basado en los árboles de decisión. En este caso, se realiza un conjunto de árboles de decisión, y al final la clasificación se realiza mediante una votación (la clasificación es la clase más frecuente), o mediante algún método de promedio.

3.3. Selección de características

El problema de la seleccion de características (FS por sus siglas en inglés) puede ser establecido como la búsqueda de un subconjunto de d características de un total de D disponibles, que logran el mayor desempeño como características para realizar una clasificación [7], de acuerdo con alguna función de criterio.

Aunque se busca el conjunto de características óptimo, que maximice la función de criterio, actualmente no existe un método de FS capaz de resolver este problema en un tiempo razonable, especialmente cuando el conjunto de datos cuenta con varias decenas de características. Por lo tanto, se han desarrollado diferentes métodos que, aunque no garantizan encontrar la solución óptima, permiten obtener un conjunto subóptimo de características. Algunos métodos de FS utiilizados comunmente son:

- **Selección Secuencial hacia Adelante (SFS)**
  El método de Selección Secuencial hacia Adelante (SFS, por sus siglas en inglés) es un procedimiento que iterativamente trata de encontrar la nueva mejor característica que, junto a las demás, aumenta el desempeño. Iniciando con un conjunto con cero características, se agregan las características una a una: si el modelo mejora (o si se obtiene un resultado óptimo con la primera característica), esta se deja en el conjunto seleccionado, de lo contrario se elimina [7].

– **Algoritmos genéticos (GA)**
  El algoritmo de FS mediante Algoritmos Genéticos (GA, por sus siglas en ingles) es un método aleatorio guiado por una cierta medida de ajuste, e inspirado por el proceso natural de evolución. Cada posible conjunto de características (*individuo*) es representado por una cadena de D bits, $\alpha_1, \cdots, \alpha_D$, donde $\alpha_i = 1$ si la i-esima características se encuentra en el conjunto, y $\alpha_i = 0$ de lo contrario. En cada iteración del algoritmo (*generación*), un número fijo de posibles soluciones (*población*) es generado aplicando ciertos operadores genéticos (*recombinación*, *cruzamiento* y *mutación*) en un proceso estocástico. A medida que aumentan las generaciones, la población tiende a tener individuos con mejor desempeño, llegando a una solución subóptima.
– **Boruta**
  El algoritmo de boruta para la selección de características, implementado por primera vez para el lenguaje R en [15], está construido utilizando el algoritmo de bosques aleatorios. Este método consiste en agregar copias mezcladas aleatoriamente de todas las características (las características sombra) al conjunto de datos original, y utilizar un modelo de clasificación de bosques aleatorios con este nuevo conjunto de datos. Aplicando una métrica de importancia de características como la Exactitud de Disminución Media, se evalúa la importancia de cada característica. En cada iteración, el algoritmo de Boruta compara la importancia de cada característica con las características sombra, y elimina aquellas menos importantes, relativo a las características sombra [18].
– **LASSO**
  El algoritmo de Operador de Selección y Contracción Mínima Absoluta (LASSO, por sus siglas en inglés) es otro método de FS. Este modelo toma como base a la regresión lineal y agrega un término de regularización que penaliza la suma del valor absoluto de los coeficientes de regresión, L1, que forza a estos coeficientes predictorios a tender a cero. Para el proceso de FS, las variables que aún tienen un coeficiente diferente a cero después de aplicar la regularización son seleccionadas para el modelo [8].

3.4. Evaluación

Para la evaluación de los modelos existe una gran variedad de métricas. En particular, utilizamos la *exactitud* (probabilidad de que el algoritmo clasifique a alguna instancia correctamente), la *sensibilidad* (probabilidad de que el valor predicho sea correcto dado que el valor real es positivo), la *especificidad* (probabilidad de que el valor predicho sea correcto dado que el valor real es negativo) y la *precisión* (probabilidad de que la clasificación sea correcta, dado que el algoritmo hizo una clasificación como positivo) [5]. Estas métricas pueden ser obtenidos de la siguiente forma:

$$\text{Exactitud} = \frac{T_P + T_N}{T_P + T_N + F_P + F_N}, \tag{1}$$

$$\text{Sensibilidad} = \frac{T_P}{T_P + F_N}, \tag{2}$$

$$\text{Especificidad} = \frac{T_N}{T_N + F_P}, \tag{3}$$

$$\text{Precisión} = \frac{T_P}{T_P + F_P}, \tag{4}$$

donde T_P, T_N, F_P y F_N es el numero de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos, respectivamente. Sin embargo, si el número de instancias positivas y negativas es muy diferente, la exactitud no es una buena métrica. Para ello, consideramos la exactitud balanceada obtenida mediante la media aritmética de la sensibilidad y la especificidad. En el caso en el que el número de instancias positivas y negativas es igual, su valor es igual al de la exactitud normal. Además, utilizamos el f1 que se obtiene como la media armónica de la precisión y la sensibilidad:

$$\text{f1} = \frac{2}{\frac{1}{\text{Precisión}} + \frac{1}{\text{Sensibilidad}}}. \tag{5}$$

Con los cuatro valores anteriores (T_P, T_N, F_P y F_N) se obtiene la matriz de confusión que se forma al acomodar los valores en una matriz de 2×2 de la siguiente forma:

		Predicción	
Valor real		Verdaderos Positivos (T_P)	Falsos Negativos (F_N)
		Falsos Positivos (F_P)	Verdaderos Negativos (T_N)

Para tareas de clasificación binaria (0 o 1), cuando el resultado del modelo corresponde a la probabilidad de pertenecer a la clase positiva (1), se pueden obtener diferentes valores de sensibilidad y especificidad de acuerdo al umbral utilizado (la probabilidad mínima para que el modelo clasifique a la instancia como positiva).

La curva ROC (figura 3.4) se forma al graficar $1 - \text{especificidad}$ contra la sensibilidad al variar el umbral entre 0 y 1. El área bajo la curva ROC (AUC) representa el grado o la medida en la que el modelo es capaz de distinguir entre las dos clases: un valor de 1 indica un modelo sin error, un valor de 0 indica que el modelo realiza una clasificación opuesta (clasifica a los 0's como 1's y viceversa), mientras que un 0.5 indica que el modelo no tiene ninguna capacidad para separar las clases.

3.5. Validación

Comúnmente, el proceso de validación de un modelo consiste en dividir el conjunto de datos en 2 subconjuntos, uno de prueba y uno de validación.

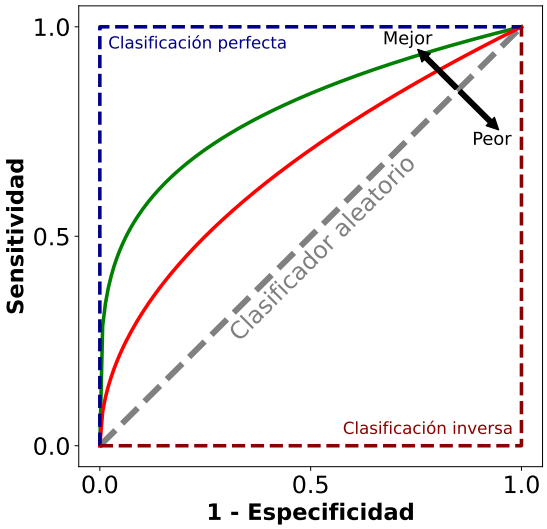


Fig. 1. Curva ROC. El área bajo la curva (AUC) representa el grado en el que el modelo es capaz de diferenciar entre dos clases. Un modelo perfecto (AUC=1) tendrá la curva azul, mientras que un modelo sin ninguna capacidad de clasificación (AUC=0.5) tendrá la curva gris. Una curva por debajo del clasificador aleatorio (curva gris, $\mathrm{AUC} < 0{,}5$), tenderá a hacer clasificaciones inversas (los 0's los clasifica como 1's y visceversa).

El primero de ellos es utilizado durante el entrenamiento, mientras que el segundo es utilizado durante la evaluación. Se le conoce como *generalización* a la capacidad de un modelo para realizar predicciones en datos (pacientes) que no se han utilizado durante el entrenamiento.

El método de validación cruzada dejando uno fuera (LOOCV, por sus siglas en inglés) consiste en realizar el proceso de validación N veces (donde N es el número total de instancias en el conjunto de datos), uno para cada una de las instancias. En cada ocasión, el modelo se prueba con una única instancia y se entrena con las $N - 1$ restantes, de manera que se utiliza la mayor cantidad de datos para el entrenamiento, y cada instancia pasa por la fase de validación.

El metodo de validación a ciegas consiste en separar un subconjunto de datos que no pasa por el proceso de generación y evaluación del modelo, sino que es utilizado para evaluar los modelos una vez ya generados. El conjunto de datos original se separa en dos subconjuntos: entrenamiento y prueba (train y test); el primero se utiliza para generar los modelos y evaluarlos mediante validación cruzada, y el segundo se utiliza una vez que los modelos ya están generados para estimar el rendimiento final.

3.6. Metodología seguida

El proceso general utilizado para la selección de los metabolitos y evaluación de los modelos de RF es el ilustrado en la figura 2, que es el correspondiente a una evaluación a ciegas.

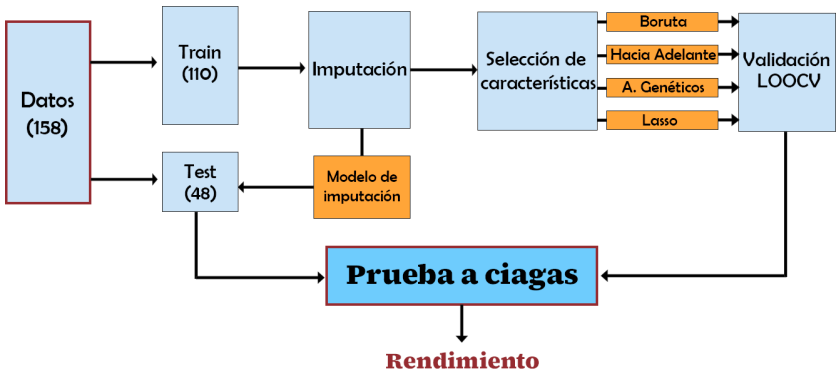


Fig. 2. Esquema general de la metodología seguida para realizar la selección de los metabolitos para el modelo de RF, y su evaluación final mediante una prueba a ciegas.

El conjunto de datos original, que consta de los datos de 158 pacientes, es dividido en 2 subconjuntos mediante una división de 70 % (110 pacientes para entrenamiento) - 30 % (48 pacientes para test).

El primer subconjunto de datos (train) es utilizado para generar el modelo de imputación de datos y para realizar los modelos de RF al seleccionar los metabolitos, los cuales son entrenados con este subconjunto y evaluados mediante una validación cruzada dejando uno fuera (LOOCV).

La validación del modelo a ciegas se realiza utilizando el segundo subconjunto de datos (test), al cual se le realiza la imputación de datos faltantes utilizando el modelo generado con el primer conjunto, y luego es utilizado para evaluar los modelos previamente entrenados con los datos de entrenamiento (train), con lo que se obtiene la estimación del rendimiento final de los modelos.

4. Resultados

Todos los resultados, mostrados en esta sección, fueron obtenidos mediante las cuatro técnicas de selección de características que se encuentran en la sección 3.3, utilizando de base un modelo de bosques aleatorios con 500 árboles de decisión, y el f1 como métrica de evaluación.

4.1. Selección del mejor conjunto de metabolitos

En conjunto, un total de 35 metabolitos diferentes fueron seleccionados por las 4 técnicas de selección de características utilizadas. La figura 3 muestra los metabolitos seleccionados con cada una de las técnicas. Entre ellos, destacan 3 que fueron seleccionados por 3 de las 4 tecnicas: el lysoPC a C26:0, el lysoPC a C14:00 y el radio quinurenina/triptófano, mientras que 10 fueron seleccionados por 2 de las 4 técnicas.

De los cuatro conjuntos seleccionados, el generado mediante selección hacia adelante es el que contiene la menor cantidad de características (5), seguido del generado mediante la técnica de LASSO (6).

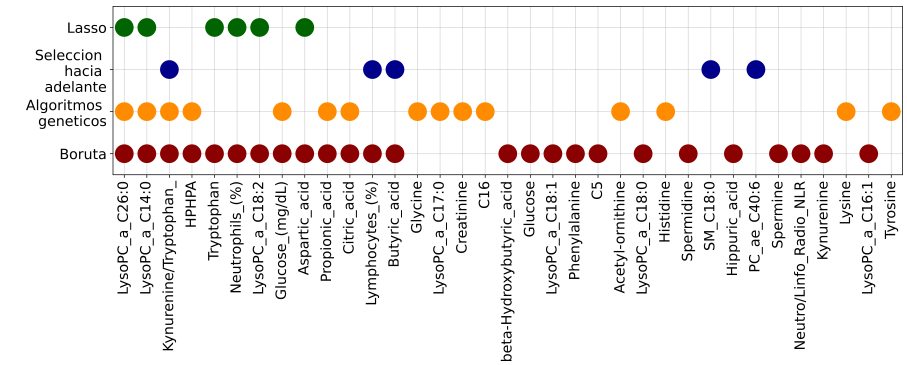
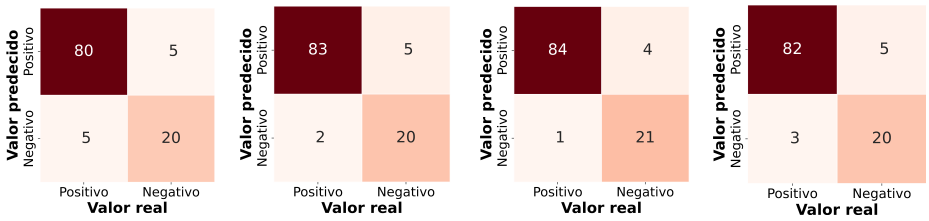


Fig. 3. Conjuntos de metabolitos seleccionados por las 4 técnicas de selección de características.



a) Boruta b) SFS c) GA d) LASSO

Fig. 4. Matrices de confusión para los 4 conjuntos de metabolitos seleccionados mediante técnicas de selección de características al realizar LOOCV en el conjunto de entrenamiento.

En cambio, las técnicas de Algoritmos Genéticos y Boruta generan los conjuntos más grandes, con 15 y 25, respectivamente, haciéndolos los menos prácticos para su uso como herramientas de diagnóstico, en cuestión de datos de laboratorio requeridos.

4.2. LOOCV en el conjunto de entrenamiento

La evaluación de los 4 conjuntos de metabolitos se realizó mediante LOOCV. La figura 4 muestra la matriz de confusión de los 4 modelos obtenida utilizando el conjunto de entrenamiento (con el cual se realizó la selección de características).

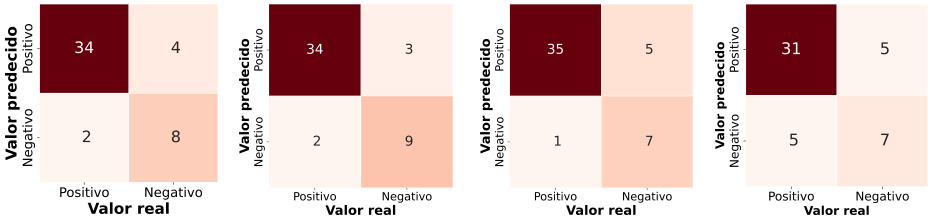
La tabla 1 muestra las diferentes métricas obtenidas a partir de los modelos anteriores. A pesar de la diferencia entre los conjuntos de metabolitos utilizados para cada modelo, los resultados de los 4 modelos son similares en términos de f1, siendo el obtenido por Algoritmos Genéticos el que alcanza un mejor resultado tanto en f1 como en exactitud balanceada.

4.3. Validación a ciegas

Para la validación a ciegas se utilizó el 30 % de los datos del conjunto original, y se utilizó el 70 % restante (el mismo conjunto con el que se realizó la selección de características) tanto para la imputación de datos faltantes como para el entrenamiento de los modelos.

Tabla 1. Métricas para la evaluación de los modelos generados con los 4 conjuntos de metabolitos obtenidos utilizando las técnicas de selección de características en el conjunto de entrenamiento por LOOCV.

	Lasso	**SFS**	**Boruta**	**GA**
Precisión	0.9647	0.9765	0.9412	0.9882
sensibilidad	0.9425	0.9432	0.9412	0.9545
f1	0.9534	0.9595	0.9411	0.9711
Exactitud balanceada	0.9272	0.9363	0.9091	0.9545
Especificidad	0.8696	0.9090	0.8000	0.9545
ROC_AUC	0.9572	0.9435	0.9689	0.9849



a) Boruta b) SFS c) GA d) LASSO

Fig. 5. Matrices de confusión para los 4 conjuntos de metabolitos seleccionados mediante técnicas de selección de características al realizar validación a ciegas.

La figura 5 muestra las matrices de confusión de los 4 conjuntos de metabolitos seleccionados, y en la tabla 2 se muestran las métricas de evaluación de los modelos. Además, en la figura 6 se muestran las curvas ROC tanto para el conjunto de entrenamiento como para la evaluación a ciegas.

Durante la validación a ciegas, el mejor modelo basado en términos de f1 y exactitud balanceada es el obtenido por selección hacia adelante, sin embargo, su rendimiento está muy cercano al obtenido mediante algoritmos genéticos (mejor rendimiento en el conjunto de entrenamiento) que se encuentra como el segundo mejor.

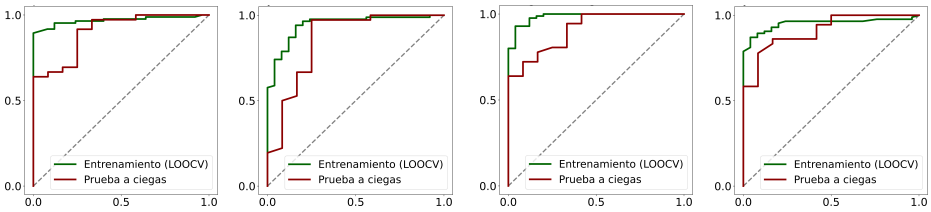
5. Conclusiones

Los resultados obtenidos en el presente trabajo de investigación muestran el desempeño del modelo de bosques aleatorios con niveles de metabolitos como características para el diagnóstico del COVID-19 en pacientes sospechosos, mostrando el alto potencial de la combinación de la metabolómica con el ML para el diagnóstico de esta enfermedad.

El conjunto con mejor desempeño en los datos de entrenamiento, en términos de la exactitud balanceada y el f1, fue el obtenido mediante algoritmos genéticos que consta de 15 metabolitos, y alcanza un f1 de 0.971 y una exactitud balanceada de 0.954. Al realizar la prueba ciega con los 48 pacientes, este conjunto obtiene el segundo mejor desempeño con un f1 de 0.921 y una exactitud de 0.875, muy cercano al obtenido por selección hacia adelante.

Tabla 2. Métricas para la evaluación de los modelos generados con los 4 conjuntos de metabolitos obtenidos utilizando las técnicas de selección de características al realizar validación a ciegas.

	Lasso	**SFS**	**Boruta**	**GA**
Precisión	0.8611	0.9444	0.9444	0.9722
sensibilidad	0.8611	0.9189	0.8947	0.8750
f1	0.8611	0.9315	0.8189	0.9211
Exactitud balanceada	0.7917	0.8958	0.8750	0.8750
Especificidad	0.5833	0.8182	0.8000	0.8750
ROC_AUC	0.9236	0.8576	0.9028	0.9100



a) Boruta b) SFS c) GA d) LASSO

Fig. 6. Curvas ROC para los 4 conjuntos de metabolitos, tanto en el conjunto de entrenamiento (obtenidos mediante LOOCV) como en la validación a ciegas.

En cambio, el conjunto obtenido por selección hacia adelante consta únicamente de 5 características y alcanza un f1 de 0.931 en la prueba a ciegas y una exactitud balanceada de 0.896, muy cercano al rendimiento obtenido al realizar LOOCV en el conjunto de entrenamiento (0.936 y 0.900, respectivamente).

A pesar del limitado número de pacientes utilizados para los modelos, el rendimiento obtenido en la prueba a ciegas es muy similar al obtenido con el conjunto de entrenamiento, lo cual puede observarse tanto al comparar las tablas 1 y 2 como al observar las curvas ROC, mostrando que los modelos tienen un alto nivel de generalización.

Dado el alto rendimiento mostrado en los resultados por los modelos, tanto en el conjunto de entrenamiento y especialmente al realizar la validación a ciegas, se propone el uso de estos conjuntos de metabolitos como biomarcadores de diagnóstico para el COVID-19, así como la futura implementación de diagnóstico automático utilizando tanto bosques aleatorios como otros modelos de ML, entrenados y evaluados con un conjunto de datos de más pacientes.

Agradecimientos. Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico brindado mediante la “Beca Nacional para Estudios de Posgrado” y por el desarrollo del proyecto “Paradigmas y Controversias de la Ciencia 2022” de número 319503 con el que fue obtenido el conjunto de datos, gracias a los cuales se está desarrollando el trabajo de investigación del cual surge el presente artículo.

Referencias

1. Agrebi, S., Larbi, A.: Use of artificial intelligence in infectious diseases. Artificial Intelligence in Precision Health, pp. 415–438 (2020) doi: 10.1016/B978-0-12-817133-2.00018-5
2. Ali, I., Alharbi, O. M.: Covid-19: Disease, management, treatment, and social impact. Science of The Total Environment, vol. 728, pp. 138861 (2020) doi: 10.1016/j.scitotenv.2020.138861
3. Asundi, A., O'Leary, C., Bhadelia, N.: Global COVID-19 vaccine inequity: The scope, the impact, and the challenges. Cell Host & Microbe, vol. 29, no. 7, pp. 1036–1039 (2021) doi: 10.1016/j.chom.2021.06.007
4. Azad, A. K., Hakim, A., Hasan-Sohag, M. M., Rahman, M.: Metabolomics in clinical diagnosis, prognosis, and treatment of infectious diseases. Metabolomics, pp. 71–119 (2023) doi: 10.1016/B978-0-323-99924-3.00003-0
5. Burkov, A.: The hundred-page machine learning book. vol. 1 (2019)
6. Davenport, T., Kalakota, R.: The potential for artificial intelligence in healthcare. Future healthcare journal, vol. 6, no. 2, pp. 94–98 (2019) doi: 10.7861/futurehosp.6-2-94
7. Ferri, F. J., Pudil, P., Hatef, M., Kittler, J.: Comparative study of techniques for large-scale feature selection. Machine Intelligence and Pattern Recognition, vol. 16, pp. 403–413 (1994) doi: 10.1016/B978-0-444-81892-8.50040-7
8. Fonti, V., Belitser, E.: Feature selection using lasso. VU Amsterdam Research Paper in Business Analytics, vol. 30, pp. 1–25 (2017)
9. Fraser, D. D., Slessarev, M., Martin, C. M., Daley, M., Patel, M. A., Miller, M. R., Patterson, E. K., O'Gorman, D. B., Gill, S. E., Wishart, D. S., Mandal, R., Cepinskas, G.: Metabolomics profiling of critically ill coronavirus disease 2019 patients: Identification of diagnostic and prognostic biomarkers. Critical Care Explorations, vol. 2, no. 10 (2020) doi: 10.1097/CCE.0000000000000272
10. Hasan, M. R., Suleiman, M., Perez-Lopez, A.: Metabolomics in the diagnosis and prognosis of COVID-19. Frontiers in Genetics, vol. 12, pp. 721556 (2021) doi: 10.3389/fgene.2021.721556
11. Hemdan, E. E. D., El-Shafai, W., Sayed, A.: CR19: A framework for preliminary detection of COVID-19 in cough audio signals using machine learning algorithms for automated medical diagnosis applications. Journal of Ambient Intelligence and Humanized Computing, pp. 1–13 (2022) doi: 10.1007/s12652-022-03732-0
12. Hossain, B., Morooka, T., Okuno, M., Nii, M., Yoshiya, S., Kobashi, S.: Surgical outcome prediction in total knee arthroplasty using machine learning. Intelligent Automation & Soft Computing, vol. 25, no. 1, pp. 105–115 (2019)
13. Jamil, S., Mark, N., Carlos, G., Cruz, C. S. D., Gross, J. E., Pasnick, S.: Diagnosis and management of COVID-19 disease. American Journal of Respiratory and Critical Care Medicine, vol. 201, no. 10, pp. P19–P20 (2020) doi: 10.1164/rccm.2020C1
14. Kingsford, C., Salzberg, S. L.: What are decision trees? Nature biotechnology, vol. 26, no. 9, pp. 1011–1013 (2008) doi: 10.1038/nbt0908-1011
15. Kursa, M. B., Rudnicki, W. R.: Feature selection with the Boruta package. Journal of Statistical Software, vol. 36, no. 11, pp. 1–13 (2010) doi: 10.18637/jss.v036.i11
16. Lin, D. Y., Gu, Y., Wheeler, B., Young, H., Holloway, S., Sunny, S. K., Moore, Z., Zeng, D.: Effectiveness of COVID-19 vaccines over a 9-month period in North Carolina. The New England Journal of Medicine, vol. 386, no. 10, pp. 933–941 (2022) doi: 10.1056/NEJMoa2117128
17. López-Hernández, Y., Monárrez-Espino, J., Herrera-van Oostdam, A. S., Castañeda-Delgado, J. E., Zhang, L., Zheng, J., Oropeza-Valdez, J. J., Mandal, R.,

Ochoa-González, F. L., Borrego-Moreno, J. C., Trejo-Medinilla, F. M., López, J. A., Enciso-Moreno, J. A., Wishart, D. S.: Targeted metabolomics identifies high performing diagnostic and prognostic biomarkers for COVID-19. Scientific Reports, vol. 11, no. 1, pp. 14732 (2021) doi: 10.1038/s41598-021-94171-y

18. Perlato, A.: Feature selection using boruta algorithm (2020) www.andreaperlato.com/mlpost/feature-selection-using-boruta-algorithm/
19. Pourkarim, F., Pourtaghi-Anvarian, S., Rezaee, H.: Molnupiravir: A new candidate for COVID-19 treatment. Pharmacology Research & Perspectives, vol. 10, no. 1 (2022) doi: 10.1002/prp2.909
20. Shahin, O. R., Alshammari, H. H., Taloba, A. I., Abd El-Aziz, R. M.: Machine learning approach for autonomous detection and classification of COVID-19 virus. Computers and Electrical Engineering, vol. 101, pp. 108055 (2022) doi: 10.1016/j.compeleceng.2022.108055
21. Tounta, V., Liu, Y., Cheyne, A., Larrouy-Maumus, G.: Metabolomics in infectious diseases and drug discovery. Molecular Omics, vol. 17, no. 3, pp. 376–393 (2021) doi: 10.1039/D1MO00017A
22. Who coronavirus (COVID-19) dashboard, https://covid19.who.int/
23. Wu, F., Zhao, S., Yu, B., Chen, Y. M., Wang, W., Song, Z. G., Hu, Y., Tao, Z. W., Tian, J. H., Pei, Y. Y., Yuan, M. L., Zhang, Y. L., Dai, F. H., Liu, Y., Wang, Q. M., Zheng, J. J., Xu, L., Holmes, E. C., Zhang, Y. Z.: A new coronavirus associated with human respiratory disease in China. Nature, vol. 579, no. 7798, pp. 265–269 (2020) doi: 10.1038/s41586-020-2008-3
24. Zhang, C., Yao, J., Hu, G., Schøtt, T.: Applying feature-weighted gradient decent K-Nearest neighbor to select promising projects for scientific funding. CMC Computers, Materials & Continua, vol. 64, no. 3, pp. 1741–1753 (2020) doi: 10.32604/cmc.2020.010306
25. Zoabi, Y., Deri-Rozov, S., Shomron, N.: Machine learning-based prediction of COVID-19 diagnosis based on symptoms. npj Digital Medicine, vol. 4, no. 1, pp. 3 (2021) doi: 10.1038/s41746-020-00372-6

ISSN 1870-4069

Reconocimiento del habla en videos digitales usando redes neuronales convolucionales

Cesar E. Embriz-Islas, Cesar Benavides-Alvarez,
Carlos Avilés-Cruz, Arturo Zúñiga-López

Universidad Autónoma Metropolitana,
Unidad Azcapotzalco,
Departamento de Electrónica,
México

{al2211800610, cesarbenavides,
caviles, azl}@azc.uam.mx

Resumen. El reconocimiento del habla con contexto visual es una técnica que utiliza el procesamiento digital de imágenes, para detectar el movimiento de los labios dentro de los cuadros de imagen de un video para predecir las palabras que está exclamando un hablante. Aunque ya existen modelos con resultados sobresalientes, la mayoría están enfocados en ambientes muy controlados con pocas interacciones del hablante. En este trabajo, se propone una nueva implementación de un modelo, basado en redes neuronales convolucionales (CNN), considerando los cuadros de imagen y tres modelos de uso del audio, por medio de espectrogramas. Los resultados obtenidos son muy alentadores en el campo del reconocimiento automático del habla.

Palabras clave: CNN, inteligencia artificial, aprendizaje profundo, reonocimiento del habla.

Speech Recognition in Digital Videos Using Convolutional Neural Networks

Abstract. Visual contextual speech recognition is a technique that uses digital image processing to detect lip movement within video image frames to predict the words a speaker is exclaiming. Although there are already models with outstanding results, most are focused on highly controlled environments with few speaker interactions. In this work, a new implementation of a model is proposed, based on convolutional neural networks (CNN), considering image frames and three models of audio use, by means of spectrograms. The results obtained are very encouraging in the field of automatic speech recognition.

Keywords: CNN, artificial intelligence, deep learning, speech recognition.

1. Introducción

El reconocimiento del habla con contexto visual es el proceso de detectar las palabras o vocablos emitidos por una persona a través de un video. Este tema, en los últimos años se ha desarrollado con gran interés, por las múltiples aplicaciones que puede tener, algunas de ellas son: el reconocimiento del habla para videos de seguridad, lectura de labios para oyentes con problemas de audición, detección de videos con audio alterado, entre otros.

Para reconocer el habla, en los estudios de Assael et al. [1] y de Garg et al. [6] se basan en procesar el video digital como cuadros de imagen (*frames*[1], nombre en en inglés), para extraer las regiones de los labios que funcionan como entrada para un modelo basado en redes neuronales convolucionales (*CNN*). Por otro lado, los trabajos [5, 2, 8] han demostrado que tomar en cuenta el audio (además de las regiones de los labios), mejoran considerablemente la precisión del reconocimiento.

En el presente artículo, se propone un nuevo modelo de inteligencia artificial basado en redes neuronales convolucionale para trabajar con videos de personas para reconocer el habla de forma automática. El modelo propuesto es un híbrido de las arquitecturas que se han utilizado, y considerando una forma diferente para entrenar los videos, se contempla la utilización de las palabras cortas y no las oraciones completas.

En la figura 1, se describen los bloques principales de nuestra propuesta para el reconocimiento automático del habla. Inicialmente se parte de una base de datos de videos (figura 1a); posteriormente, se pre-procesan los videos para extraer cuadros de imagen de la región de interés (labios), y asimismo, se procesa el audio teniendo una representación en imagen (figura 1b). Finalmente, se extraen las características importantes de las imágenes procesadas, para obtener la clasificación o reconocimiento de la palabra, ver figura 1c.

La parte del pre-procesamiento, comprende la delimitación de los labios del hablante, por cada cuadro de imagen extraído del video. Asimismo, se procesa el audio como un canal independiente considerando tres diferentes tipos de espectrogramas, en donde la elección del tipo de espectrograma adecuado es fundamental para obtener un mejor desempeño en la clasificación.

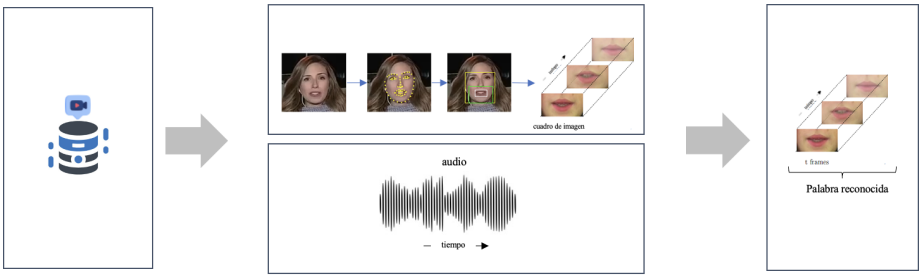
Finalmente, en la parte de clasificación, comprende todo el entrenamiento para asociar el movimiento de los labios con la palabra correcta. Se utiliza una *CNN*, como entrada, se consideran los cuadros de imagen de los labios, previamente procesados.

De igual forma para el audio, es una entrada para otra *CNN* independiente a la de los labios. Estos procesos, sirven para extraer las características de ambas entradas descritas, y con una capa *softmax*[2] se clasifican las palabras por el movimiento de los labios; más detalles serán descritos en el capítulo de Metodología.

El resto del artículo esta constituido de la Metodología, descrita en el capítulo 3. En el capítulo 4 se describe la experimentación y resultados; finalmente, las conclusiones y perspectivas son descritas en el capítulo 5.

[1] Mínima imagen completa registrable de un video.

[2] La función softmax convierte un vector de K números reales en una distribución de K resultados posibles.



(a) Base de datos (b) Pre-procesamiento (c) Clasificación

Fig. 1. Esquema general del proceso de reconocimiento del habla.

2. Estado del arte

El artículo de Belhan et al. [2] hace un comparativo entre modelos de reconocimiento del habla inglesa, considerando un entrenamiento sin audio implementando la red LipNet, otro con audio (LipNet-audio) y finalmente, un tercero con el audio aplicando la transformada de Fourier de tiempo corto, *STFT* por sus siglas en inglés, (LipNet-audio-STFT).

El modelo LipNet desarrollado por Assael et al. [1] es clásico y conocido; se apoya de capas *GRU* bidireccionales en lugar de unidireccionales. Por otro lado LipNet-audio procesa los datos visuales utilizando capas GRU unidireccionales y los datos de audio como señales 1D sin STFT. Finalmente LipNet-audio-STFT, procesa los datos visuales igual que LipNet-audio y los datos de audio con señales 3D con STFT con una redimensión a resolución 64×64 píxeles.

Al utilizar datos de audio, el modelo mejora considerablemente los resultados. Por otro lado, el modelo se desempeña mejor cuando se utiliza el audio con señales 1D en comparación con 3D y STFT, sin considerar el tiempo computacional extra por aplicar STFT. El considerar una resolución completa del audio con STFT mejoraría los resultados pero con más costo computacional.

Por otro lado, el trabajo de Feng [5] propone un modelo con una red neuronal recurrente multimodal, *m-RNN* (por sus siglas en inglés), para el reconocimiento del habla audiovisual. La estructura del modelo, consta de dos componentes: una parte visual y una de audio.

La parte visual, contiene una capa CNN más una *LSTM* bidireccional; y la parte de audio, una LSTM bidireccional. Ambas partes contienen capas de estado ponderado, para generar resultados semánticamente coherentes para la fusión. Basado en las capas de estado ponderado, la RNN multimodal utiliza una capa multimodal para fusionar ambas modalidades, y una capa *softmax* para la salida.

De los estudios más recientes, Jeon et al. [8] presenta una arquitectura de lectura de labios para el reconocimiento del habla visual a nivel de oración. La arquitectura está compuesta por tres módulos de extracción de características visuales, y se aplican múltiples métodos de extracción de características visuales, que logran una mejor predicción del movimiento de los labios. En los estudios anteriormente mencionados, se ha utilizado la base de datos GRID [4].

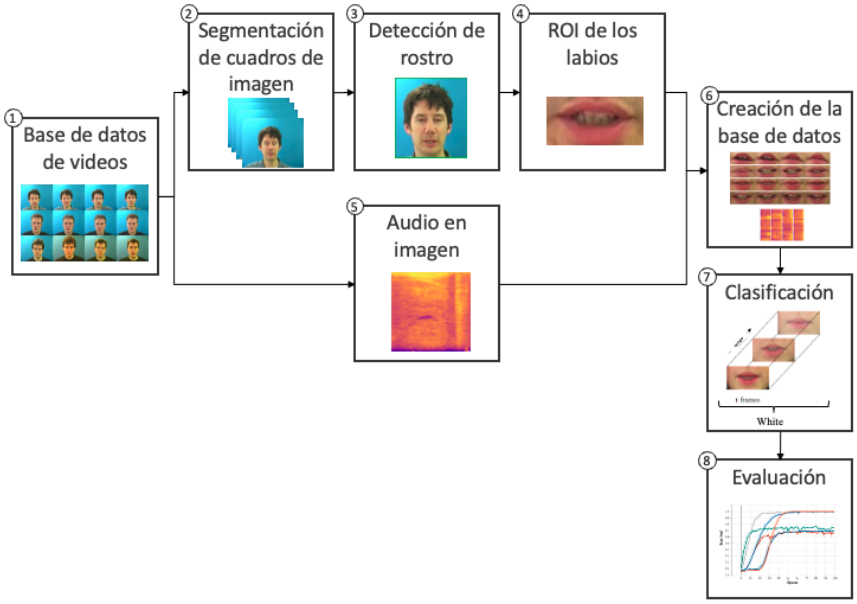


Fig. 2. Metodología del sistema de reconocimiento del habla.

Otro enfoque importante es el que propone Rossler et al. [7] utilizando la base de datos [9], la cual está constituida por falsificaciones faciales, plantea sobre la construcción de un modelo que reconoce falsificación de videos cuando el audio no coincide con el movimiento de labios.

Para detectar el video alterado, propone un extractor espacio-temporal de características, seguido de una CNN temporal para la lectura de labios. Posteriormente, se congela el extractor de características y se ajusta la red temporal de falsificación de datos.

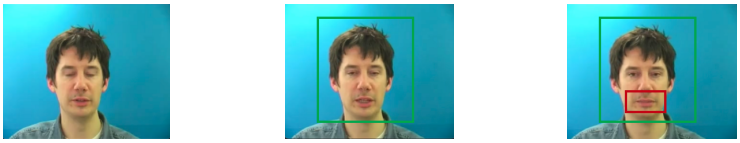
3. Metodología

El modelo propuesto y los bloques metodológicos se muestra en la figura 2, cada uno de los ocho bloques constitutivos se describen a continuación:

3.1. Base de datos de videos

Se crea a partir de las palabras extraídas de la base de datos *GRID* [4], el corpus de oraciones audiovisuales de múltiples hablantes. El corpus, consta de grabaciones de audio y video, donde se aprecia perfectamente el rostro. La resolución del video es de 360×288 píxeles y un audio a $50kHz$.

El corpus, contiene 1,000 oraciones pronunciadas por cada uno de los 33 hablantes (17 hombres y 16 mujeres), con un total de 33,000 oraciones. La estructura de las oraciones es la siguiente:



(a) Cuadro extraído (b) Detección del rostro (c) Detección de ROI

Fig. 3. Detección del rostro y ROI de los labios.

$$\text{comando}(4) + \text{color}(4) + \text{preposición}(4) + \text{letra}(25) + \text{dígito}(10) + \text{adverbio}(4)$$

donde (#) indica la cardinalidad del conjunto de palabras para cada una de las 6 categorías:

- *comando:* {bin, lay, place, set},
- *color:* {blue, green, red, white},
- *preposición:* {at, by, in, with},
- *letra:* {A,. . . ,Z}/{W},
- *dígito:* {zero,. . . , nueve},
- *adverbio:* {again, now, please, soon}.

Y un ejemplo es: *'set blue by A four please'*.
La base de datos contiene un total de 51 palabras diferentes.

3.2. Segmentación de cuadros de imagen

Se procesan los videos digitales, para extraer 30 cuadros de imagen por segundo del video. En la figura 3a se muestra un cuadro extraído del video.

3.3. Detección del rostro

Por cada cuadro de imagen extraído del video, se detecta el rostro y obtienen las coordenadas que lo comprenden. En la figura 3b se presenta un ejemplo de la detección del rostro dentro del cuadro de color verde.

3.4. Región de interés (*ROI*) de los labios

Una vez detectado el rostro, se obtienen las coordenadas de los labios. Estas últimas coordenadas se utilizan para considerar un área rectangular, y delimitar la *ROI* que se utiliza para extracción de características. Del largo de los labios, se consideran 5 píxeles adicionales, tanto para la izquierda como la derecha; y para la altura, se consideran 10 píxeles, para arriba y abajo; con esto se delimita el rectángulo mencionado. En la figura 3c se muestra un ejemplo de la detección de *ROI* de los labios en color rojo.

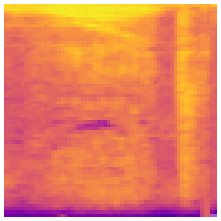


Fig. 4. Espectrograma tipo 1 del audio correspondiente a la palabra "*white*".

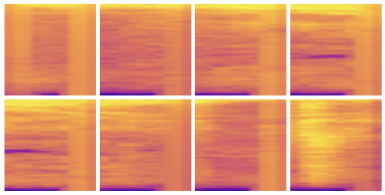


Fig. 5. Espectrograma tipo 2 del audio correspondiente a la palabra "*white*".

3.5. Transformación audio en imagen

Por cada segmento de audio (señal unidimensional), se realiza una transformación Tiempo-Frecuencia (señal bi-dimensional, imagen), llamado espectrograma, para convertir el audio a imagen; la transformación se aplica con la ecuación 1:

$$V_g f(x,\omega) = \int_{-\infty}^{\infty} f(t) g(t-x) e^{-2\pi i t\dot{\omega}}\, dt, \tag{1}$$

donde ω representa el tiempo del audio, y para este proyecto, se contemplan tres posibles variaciones.

1. Un único espectrograma de la palabra, ver Figura 4. Se toma el audio completo de la palabra para realizar la transformación Tiempo-Frecuencia.
2. En la figura 5, se observa la cantidad de espectrogramas alineados al mismo número de cuadros de imagen del video. En este caso, se genera un espectrograma para cada *frame*. Así, se tienen igual número de *frame* y espectrogramas.
3. En este última opción, cada segundo de la palabra se divide en 10 intervalos equi-espaciados, obteniendo solo diez espectrogramas por segundo (ver figura 6).

Este preprocesamiento se ejecuta en paralelo al de la extracción de la ROI de los labios.

3.6. Creación de base de datos

Con los pre-procesamientos anteriores, se genera una nueva base de datos, que comprende 30 cuadros por segundo de la ROI de los labios, con dimensiones de 160×80. En la figura 7 se muestran ejemplos de cuadros de imagen procesados con la ROI seleccionada.

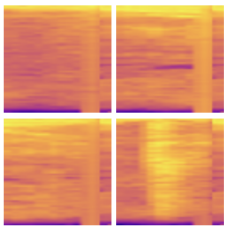


Fig. 6. Espectrograma tipo 3 del audio correspondiente a la palabra "*white*".

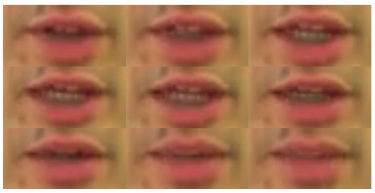


Fig. 7. Labios procesados del cuadro de imagen correspondientes a la palabra "*again*".

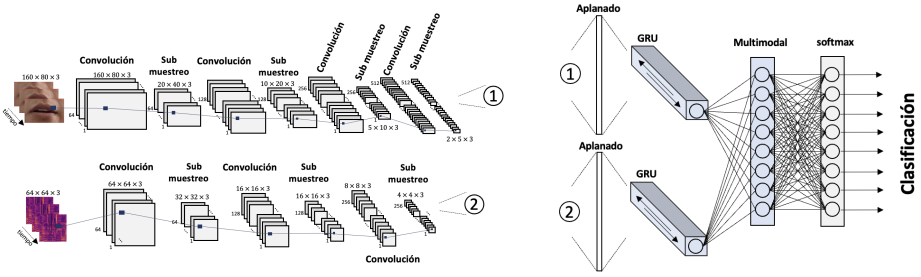
Por otro lado, también se tienen imágenes con un tamaño de 64×64 pixeles para los diferentes tipos de espectrogramas planteados anteriormente vistos en las figuras 4, 5 y 6.

3.7. Clasificación - Modelo de aprendizaje profundo -

Para la clasificación se plantea un modelo de Inteligencia Artificial basado en aprendizaje profundo. Particularmente, se usan redes neuronales convolucionales, redes neuronales retroalimentadas y redes neuronales de clasificación.

El modelo propuesto empata la información del movimiento de los labios, con el espectrograma proveniente del audio. El modelo de aprendizaje profundo propuesto (ver figura 8a) es detallado a continuación:

(a) Una CNN para la extracción de características importantes de las imágenes de los labios. La CNN consta de cuatro capas convolucionales; la primera capa la constituyen 64 filtros con un kernel de $(3, 5, 5)$, un paso de $(1, 2, 2)$, y un relleno igual; para la segunda, 128 filtros con un kernel de $(3, 5, 5)$; la tercera y cuarta tienen 256 y 512 filtros, respectivamente, y ambas utilizando los parámetros por defecto con un kernel de $(3, 3, 3)$. Cada una de las capas utiliza una función de activación *relu* y cada capa es acompañada por un sub-muestreo (*max-pooling*) y una normalización (*batch_normalization*). Asimismo, la entrada de esta red, es una serie de cuadros de imagen de $160 \times 80 \times 3$ píxeles que representan la palabra expresada.
(b) Una *CNN* para la extracción de características importantes del espectrograma. La *CNN* consta de tres capas convolucionales, tres de sub-muestreo y tres de normalización. Esta red, es una copia hasta la tercer capa de la anteriormente descrita, la diferencia es que existen tres entradas diferentes, que hacen referencia a los tres tipos de espectrogramas.



(a) Primera parte de la red

(b) Segunda parte de la red

Fig. 8. Modelo propuesto.

(c) Dos capas de *aplanamiento* sobre tiempo distribuido, una por cada *CNN*.
(d) Dos capas bidireccionales *GRU* por cada *aplanamiento*. Para ambas redes, los filtros que se aplican son 2048 y 1024.
(e) Una capa densa por cada red, cada una de 512 filtros con activación *leakyrelu*.
(f) Una capa *multimodal* que concatene las dos salidas anteriores.
(g) Una serie de 3 capas densas con filtros descendientes de 512, 256 y 128; todas con una activación *relu*.
(h) Finalmente, una capa *softmax* con 51 clases (número de palabras) para la clasificación.

En la figura 8 se muestra, de manera gráfica, la implementación del modelo, en donde se describen las convoluciones con la reducción de dimensiones de las imágenes y el incremento de filtros; finalmente la red recurrente con la multimodal para definir el clasificador. Por otra parte, en la figura 9 se muestra la implementación del modelo con tensorflow con mucho más detalle de la evolución de dimensiones y la capas que se han aplicado.

4. Experimentación y resultados

Se hace una selección de dos observaciones por palabra y por hablante. Obteniendo un total de 102 videos de palabras por cada hablante, en total 3,366 videos de las 51 clases de palabras. Siguiendo los pasos de la metodología; para los videos, se segmentan en cuadros de imagen, detecta el rostro y se extrae la ROI de los labios; para el audio, se generan los tres tipos de espectrogramas.

En ambos procesos, se guardan las imágenes correspondientes, creando una nueva base de datos. Con el objetivo de tener más estabilidad en el modelo, se genera un aumento de datos, para finalmente entrenar y evaluar el modelo propuesto.

4.1. Procesamiento de los labios

El número de imágenes que genera cada palabra depende de la complejidad, tamaño y pronunciación. Hay algunas palabras cortas como la letra *I*, que la representan seis cuadros de imagen.

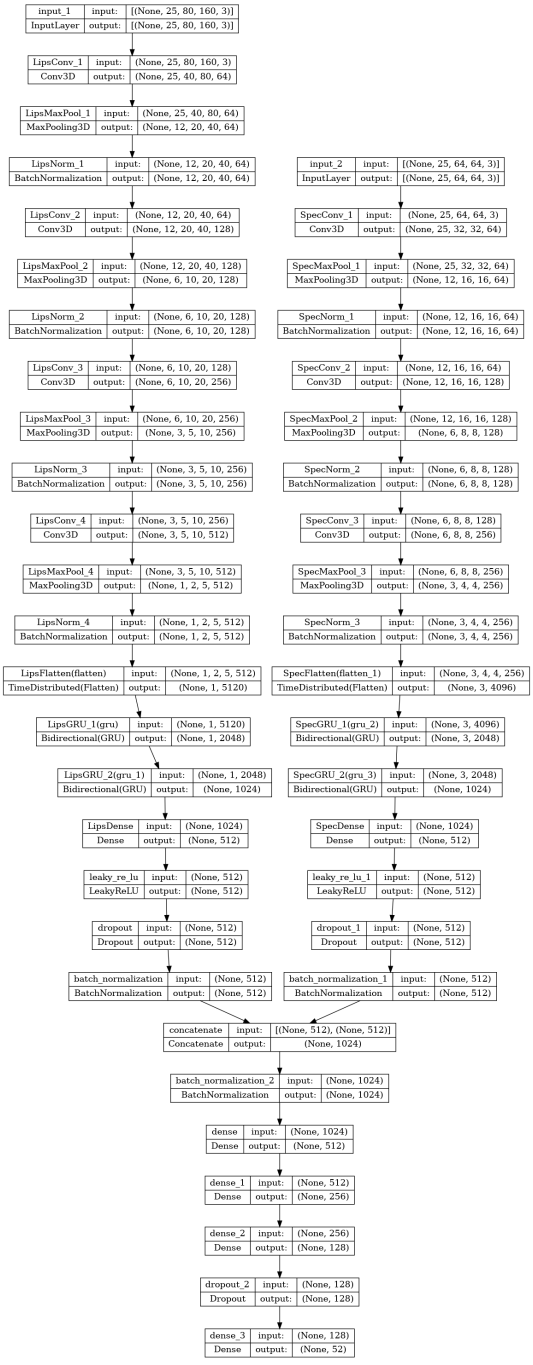


Fig. 9. Modelo implementado en Tensorflow.

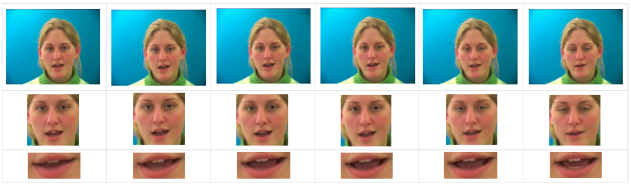


Fig. 10. Procesamiento de la letra *I*.

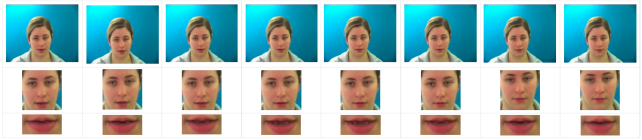


Fig. 11. Procesamiento de la palabra *again*.

En la figura 10 se muestra la etapa del procesamiento; primero, la conversión del video a cuadros de imagen; el segundo, en donde se detecta el rostro por cada cuadro de imagen; y finalmente los labios, que es la región de interés, y entrada más importante del modelo.

En las figuras 11 y 12 se ven dos ejemplos más del procesamiento que se realiza sobre los videos. Para la primera imagen, la componen un total de 8 cuadros de imagen; y para la segunda, un total de 6, similar al de la letra *I*. Con estos ejemplos se ilustra que los cuadros que representan la pronunciación de una letra o palabra no son de un tamaño en específico.

Otro factor importante, es la entonación sobre la palabra que hace cada persona. En la figura 13 se muestra el procesamiento para la palabra *by* de un hablante diferente a la expresada en la figura 12. En este segundo ejemplo, los cuadros de imagen que representan a la palabra son 4, a diferencia de la primera, que muestran 6.

4.2. Procesamiento del audio

Por otro lado, tomando de referencia la palabra *again*, se aplica el procesamiento para el audio, el cual genera tres tipos de secuencias diferentes:

1. Espectrograma tipo 1: esta transformación considera un único espectrograma de toda la palabra sin importar el número de cuadros de imagen que la componen. En la figura 14a se muestra la representación de la palabra.
2. Espectrograma tipo 2: para este caso, se obtienen un número de espectrogramas igual a el número de cuadros de imagen que representa la palabra, es decir, para la palabra *again* del ejemplo anterior, tiene un total de 6 cuadros de imagen, y cada cuadro está representado por un espectrograma como se muestra en la figura 14c.
3. Espectrograma tipo 3: en este tipo hace referencia a la duración de la pronunciación de la palabra, recordando que son 10 espectrogramas por segundo, para la palabra *again* sólo se consideran 4 espectrogramas, que es la parte proporcional del segundo, como se muestra en la figura 14c.

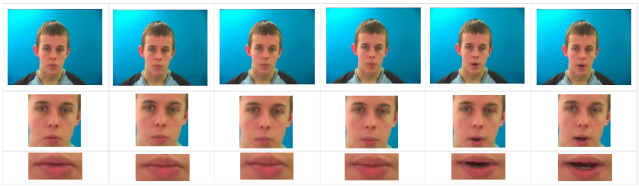


Fig. 12. Procesamiento de la palabra *by*.

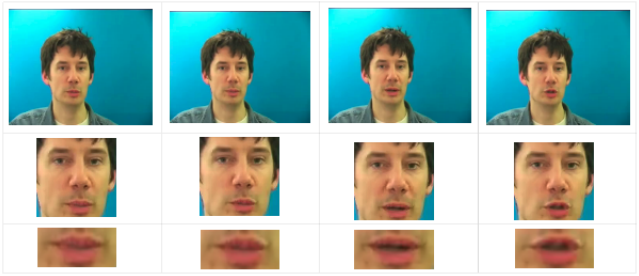


Fig. 13. Procesamiento de la palabra *by*.

De la mima manera, se genera un *aumento de datos*[3][10].

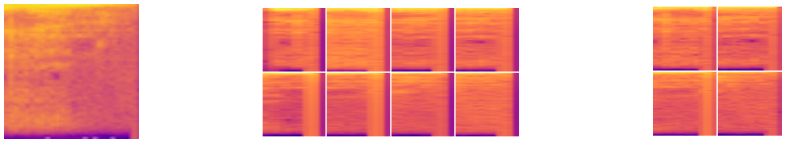
Una vez generada la base de datos aumentada, se divide en dos segmentos, el primero para entrenamiento y el segundo para validación. Para la creación de la partición de entrenamiento $train$, se considera un $77\,\%$ del total de los datos; y para la validación val, se toma el $33\,\%$ restante.

4.3. Entrenamiento

Una vez listo el pre-procesamiento de los datos, se aplica el entrenamiento del modelo de aprendizaje profundo, considerando el conjunto de datos previamente particionado. Se realizan múltiples iteraciones ajustando diferentes parámetros como la función de pérdida (*loss function*), tasa de aprendizaje (*learning rate*), épocas (*epochs*), tamaño de lote (*batch size*), entre otros. Lo siguiente es considerado en la ejecuciones de entrenamiento:

1. Función de pérdida: *entropía cruzada* entre las etiquetas y las predicciones. Se utiliza esta función ya que existen más de dos clases.
2. Optimizador: $Adam$. La optimización de Adam es un método de descenso de gradiente estocástico que se basa en la estimación adaptativa de momentos de primer y segundo orden.
3. Métricas de seguimiento: exactitud (accuracy) y exactitud en la validación (val_accuracy).

[3] El aumento de datos es una técnica utilizada en el aprendizaje automático y la visión por computadora para aumentar el tamaño y la variabilidad de un conjunto de datos de entrenamiento mediante la generación de nuevos ejemplos de entrenamiento a través de varias transformaciones de los datos originales.



(a) Espectrograma tipo 1. (b) Espectrograma tipo 2. (c) Espectrograma tipo 3.

Fig. 14. Espectrogramas de la palabra *again*.

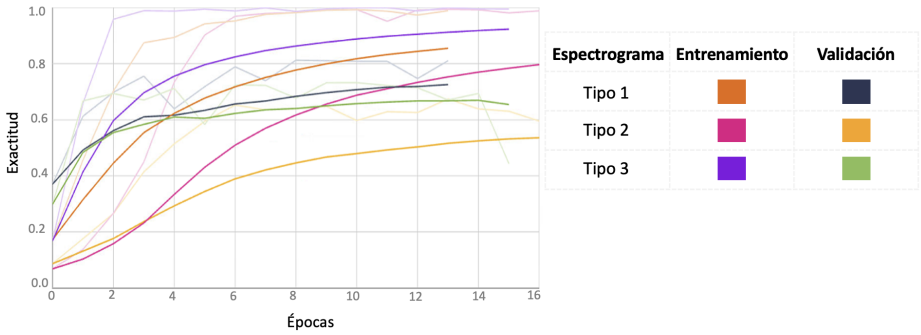


Fig. 15. Exactitud por época, evaluando cada tipo de espectrograma.

4.4. Evaluación

La aplicación del modelo se utiliza el siguiente *hardware*: Una computadora marca *Alienware*, con procesador intel core i9 12th generación, tarjeta gráfica NVIDIA GeForce RTX 3090, 24 GB GDDR6X, memoria RAM DDR5 de 64 GB y 2Tb de SSD.

Por otro lado, las consideraciones para el *software*, son las siguientes: sistema operativo *Ubuntu*, versión 20.04, lenguaje de programación *Python* 3.9.13, librería *OpenCV* [3], librería *face_recognition*[4] 1.2, librería *tensorflow*[5] 2.r2.9

Con las especificaciones anteriores, y en la evaluación, se consideran tres ejecuciones del modelo, una por cada tipo de espectrograma planteado. En la figura 15 se muestran las tres diferentes ejecuciones que se han realizado, puntualizando la exactitud por cada época para los conjuntos de entrenamiento y validación.

Los resultados obtenidos, tabla 1, muestran que los tres tipos de espectrograma presentan un comportamiento similar de aprendizaje. Sin embargo, el tipo 1 es el que demuestra un mejor desempeño con una alta exactitud tanto en entrenamiento como en validación, y con un bajo número de épocas.

Cabe destacar que el tipo 1 corresponde a una única imagen para el audio transformado. En contraste, el tipo 2 y 3 presentan una exactitud más baja en ambas fases de evaluación.

[4] https://github.com/ageitgey/face_recognition\#face-recognition
[5] https://www.tensorflow.org/?hl=es-419

Tabla 1. Exactitud por tipo de espectrograma.

Tipo	Entrenamiento (exactitud)	Validación (exactitud)
1	0.85	0.73
2	0.78	0.53
3	0.92	0.65

5. Conclusiones y trabajo futuro

Se pueden plantear las siguientes conclusiones, se analizó y se programó exitosamente el modelo de aprendizaje profundo propuesto. Por otro lado, se evaluaron las tres representaciones Tiempo-Frecuencia de audio en espectrogramas correspondientes (1) Un espectrograma por palabra, (2) un espectrograma por cuadro (frame), y (3) diez espectrogramas por segundo.

Los resultados obtenidos indican que los tres tipos de espectrogramas muestran un comportamiento similar de aprendizaje, pero el tipo 1, que corresponde a una única imagen para el audio transformado, muestra el mejor desempeño con una buena exactitud tanto en el conjunto de entrenamiento como en el conjunto de validación.

Los resultados sugieren que el tipo 1 es una buena opción para la red propuesta. Asimismo, es importante resaltar que al utilizar el tipo 1 del espectrograma, el tiempo de transformación, en la etapa del pre-procesamiento, es más eficiente.

Hasta el momento no se ha encontrado un trabajo que utilice la base de datos *GRID* [4] como en éste. Los videos utilizados, han sido descompuestos para tener la palabra, letra o número de manera particular y no dentro de una oración, como en el video original. Debido a lo anterior, es complicado tener una referencia para medir la eficiencia del modelo.

Como trabajo futuro, se propone explorar otras arquitecturas y optimización de hiper-parámetros para mejorar la exactitud en clasificación. También sería interesante evaluar el rendimiento del modelo en otros conjuntos de datos e incluso en videos en idioma español.

Referencias

1. Assael, Y. M., Shillingford, B., Whiteson, S., de Freitas, N.: LipNet: End-to-end sentence-level lipreading (2017) https://openreview.net/forum?id=BkjLkSqxg
2. Belhan, C., Fikirdanis, D., Cimen, O., Pasinli, P., Akgun, Z., Yayci, Z. O., Turkan, M.: Audio-visual speech recognition using 3D convolutional neural networks. In: 2021 Innovations in Intelligent Systems and Applications Conference (ASYU), pp. 1–5 (2021) doi: 10.1109/ASYU52992.2021.9599016
3. Bradski, G.: The opencv library. Dr. Dobb's Journal: Software Tools for the Professional Programmer, vol. 25, no. 11, pp. 120–123 (2000)
4. Cooke, M., Barker, J., Cunningham, S., Shao, X.: An audio-visual corpus for speech perception and automatic speech recognition. The Journal of the Acoustical Society of America, vol. 120, no. 5, pp. 2421–2424 (2006) doi: 10.1121/1.2229005

5. Feng, W., Guan, N., Li, Y., Zhang, X., Luo, Z.: Audio visual speech recognition with multimodal recurrent neural networks. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 681–688 (2017) doi: 10.1109/IJCNN.2017.7965918
6. Garg, A., Noyola, J., Bagadia, S.: Lip reading using CNN and LSTM. Technical report, Stanford University (2016)
7. Haliassos, A., Vougioukas, K., Petridis, S., Pantic, M.: Lips don't lie: A generalisable and robust approach to face forgery detection. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5039–5049 (2021) doi: 10.1109/CVPR46437.2021.00500
8. Jeon, S., Elsharkawy, A., Kim, M. S.: Lipreading architecture based on multiple convolutional neural networks for sentence-level visual speech recognition. Sensors, vol. 22, no. 1, pp. 72 (2021) doi: 10.3390/s22010072
9. Rossler, A., Cozzolino, D., Verdoliva, L., Riess, C., Thies, J., Niessner, M.: FaceForensics++: Learning to detect manipulated facial images. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 1–11 (2019) doi: 10.1109/iccv.2019.00009
10. Shorten, C., Khoshgoftaar, T. M.: A survey on image data augmentation for deep learning. vol. 6, no. 1, pp. 1–48 (2019)

ISSN 1870-4069

Clasificación contextual de vocalizaciones de perros de asistencia apoyada por segmentación automática

Roilhi Frajo Ibarra-Hernández[1], Luis Villaseñor-Pineda[2], Humberto Pérez-Espinoza[3], Hugo Jair Escalante[2]

[1] Instituto Nacional de Astrofísica, Óptica y Electrónica, Consejo Nacional de Ciencia y Tecnología, México

[2] Instituto Nacional de Astrofísica, Óptica y Electrónica, Coordinación de Ciencias Computacionales, México

[3] Centro de Investigación Científica y de Educación Superior de Ensenada, Unidad de Transferencia Tecnológica, México

roilhi@inaoe.mx, {villasen, hugojair}@inaoep.mx, hperez@cicese.mx,

Resumen. El Aprendizaje Automático ha permitido el desarrollo de métodos de detección de emociones a partir de las vocalizaciones humanas. Este concepto se ha extrapolado al manejo de contextos que describan el estado emocional o fisiológico de los perros de búsqueda y asistencia, con el propósito de predecir y controlar su comportamiento. Sin embargo, para el procesamiento de la señal de audio es importante considerar únicamente aquellas partes que contienen información útil para mejorar el rendimiento del clasificador, esta etapa es conocida como segmentación. En este trabajo se compara el desempeño del algoritmo de clasificación de contextos en las vocalizaciones de un conjunto de perros asistido por segmentación automática, con lo cual es evidente el ahorro en recursos y tiempo de ejecución. Los resultados obtenidos muestran una mejora en la métrica F1-score con respecto a la clasificación asistida por segmentación manual, además de un buen desempeño en la detección de ladridos, dados los altos niveles de relación señal a ruido (SNR) obtenidos entre los audios de la base de datos empleada.

Palabras clave: Clasificación de ladridos, segmentación automática, aprendizaje automático, descriptores acústicos.

Assistance Dogs Bark Classification Supported by Automatic Segmentation

Abstract. Machine Learning has allowed the development of methods to detect emotions from human vocalizations. This concepts has been taken to handle

the context which describe the emotional or physiological states in assistance dogs, when having as a goal to predict and control its behavior. However, to process the audio signal it is relevant to consider those parts which contain useful information to enhance the classification performance, this stage is known as segmentation. In this work, we compare the classifier's algorithm efficiency when it is assisted by automatic segmentation, which also improves the execution time and resources. The obtained results show an increase in the F1-score with respect to the classification scheme assisted by manual segmentation, as well as accuracy when detecting barks, given by the high levels of signal-to-noise ratio (SNR) obtained among the database used.

Keywords: Bark classification, automatic segmentaetion, machine learning, acoustic descriptors.

1. Introducción

La comunicación es una herramienta fundamental en las interacciones sociales para transmitir ideas, pensamientos y estados afectivos. Para expresar algún estado emocional interno, los seres humanos y animales usan diversas expresiones comunicativas, entre las que se encuentran las expresiones vocales.

Dentro del conjunto de vocalizaciones humanas, se sigue un conjunto de reglas simples para codificar el estado interno del hablante a través de parámetros acústicos. La especie humana ha sido capaz de usar estas reglas estructurales para asociar ladridos de perros con diversos contextos que representen su estado emocional [4].

En efecto, gracias a la domesticación ha sido posible que los perros hayan desarrollado habilidades sociales y de comunicación [14]. Estas habilidades han conducido al involucrar a los perros en las tareas de búsqueda y asistencia, en las cuales se desarrolla un proceso de entrenamiento para apoyar a las tareas de rescate y apoyo a personas con alguna discapacidad física o motriz.

Por este motivo, se ha consolidado como un tema importante el conocer el estado interno y emocional de los perros entrenados a través de los contextos de sus vocalizaciones. Por medio de un contexto definido en su ladrido se describen diversos estados psico-emocionales que permitan comprender directamente alguna acción del perro: hambre, soledad, enojo, felicidad, etc.

La comunicación interactiva canino-humana ha sido objeto de estudio debido a que aún existen diversas preguntas abiertas sobre el significado contextual de las vocalizaciones.

Particularmente los etólogos, quienes se dedican a estudiar el comportamiento de los animales, tienen el interés de crear perfiles de conducta para monitorear y predecir el comportamiento de los perros domésticos, especialmente aquellos dedicados al rescate, lazarillos entre otros que tengan algún entrenamiento especializado.

Existen caracterísitcas acústicas importantes en los ladridos y vocalizaciones del perro doméstico, tales como la frecuencia, amplitud, tono, ritmo entre otras [12]. Dichos parámetros son posibles de relacionarse con algún estado emocional, actitud, reacción fisiológica o estado particular del perro, el cual se conoce como *contexto* [6].

Tabla 1. Distribución de etiquetas (contextos) en las señales de audio de vocalizaciones de la base de datos Mudi [13].

contexto	**muestras**
ball	53
stranger	46
food	41
fight	30
walk	29
play	23
alone	22

El presente trabajo tiene como propósito fortalecer una etapa importante en la clasificación de contextos en las vocalizaciones de perros de asistencia: la segmentación. Se ha motivado por los resultados de otras investigaciones previas donde se han seleccionado como características de clasificación descriptores acústicos de bajo (LLDs) y alto nivel (HLDs) [9, 11] para alimentar un clasificador de Aprendizaje Automático basado en Máquinas de Soporte Vectorial (SVM) [2].

Entre los parámetros de caracterización de las vocalizaciones caninas, se encuentran los coeficentes cepstrales de frecuencia Mel (MFCCs), ampliamente empleados por su eficiencia en la detección y clasificación de voz, así como los espectrograms Mel (Melspec) además de otros parámetros espectrales, tales como la energía, flujo espectral, centroide, desvanecimiento, entre otros.

No obstante, en dichas investigaciones la clasificación realizada requiere incluir el proceso de segmentación, el cual consiste en delimitar los tiempos de inicio (*onset*) y fin (*offset*) de ladrido, con el objetivo de extraer y procesar las muestras útiles de la señal y descartar las pausas entre dichos eventos. Para la ejecución de la segmentación, Pérez et al., [9] ha realizado un procesamiento manual, requiriendo el uso de *anotadores* auxiliares en la tarea de registrar los tiempos de onset y offset para cada señal.

Esto puede ser un proceso exhaustivo y monótono para ser desarrollado por humanos, siendo suceptible a errores. Para evitar este consumo de recursos humanos y tiempo, en este trabajo de investigación se propone asistir la clasificación por medio de mecanismos de segmentación automática.

En este trabajo se ha seleccionado un método de segmentación basado en la detección de tosidos en señales acústicas respiratorias [7], debido a que estas señales tienen similitudes con las vocalizaciones caninas en sus componentes espectrales.

Para evaluar si es adecuado el complementar la clasificación conn el método automático de segmentación, se comparará el desempeño del algoritmo por medio de la métrica F1-score con y sin segmentación.

El artículo se divide de la siguiente manera: en la sección 2 se describirá la base de datos utilizada y los contextos objetivo de la clasificación, así como una descripción detallada del algoritmo de segmentación automática, además de la extracción y selección de características.

Posteriormente, en la sección 3 se describen los procedimientos, parámetros y algoritmos de clasificación empleados, así como los resultados producidos de la experimentación.

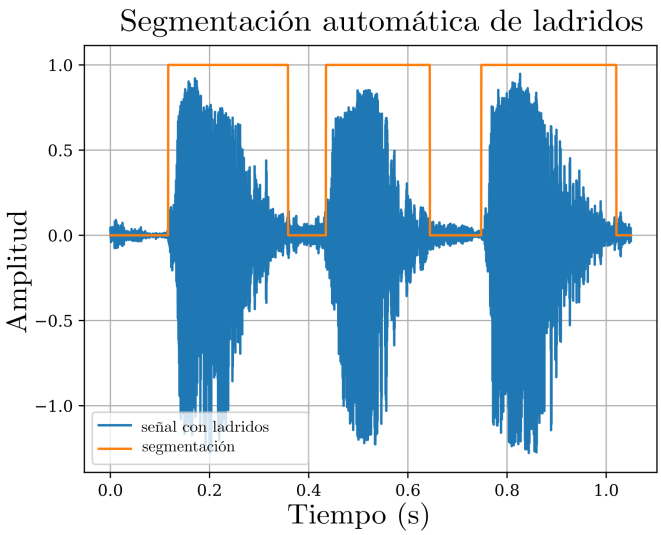


Fig. 1. Realización de la segmentación del audio con vocalizaciones. Se han detectado 3 eventos de ladrido automáticamente.

Finalmente, la sección 4 expone las conclusiones referentes a los resultados y hallazgos derivados de los experimentos conducidos en este trabajo, además de las líneas a seguir de trabajo futuro.

2. Metodología

2.1. Conjunto de datos de vocalizaciones

Para el desarrollo de los experimentos realizados en esta investigación, se ha empleado el conjunto de datos de Póngracz et al. [13], el cual comprende señales de audio de ladridos a través de los perros de la raza *Mudi*, cuyo origen proviene de Hungría. Las vocalizaciones han sido etiquetadas en siete diferentes contextos, que se muestran a continuación:

1. **Alone:** Se aisló al perro en un área exterior, atándolo. El dueño caminó lejos de la vista del perro.
2. **Ball:** El dueño sostenía una pelota o juguete a 1.5 m enfrente del perro.
3. **Fight:** El entrenador atacó al dueño y al perro. El dueño mantuvo al perro con correa.
4. **Food:** El dueño sostenía un platón de comida a 1.5 m enfrente del perro.
5. **Play:** El dueño jugó algún juego con el perro.
6. **Stranger:** El experimentador apareció en el terreno donde el perro suele convivir o enfrente del perro.
7. **Walk:** El dueño simuló una serie de acciones tal como si fuese a sacar al perro a pasear.

Tabla 2. Distribución de muestras por cada contexto después de realizar la segmentación automática.

contexto	muestras
alone	826
ball	1025
fight	1008
food	829
play	735
stranger	1318
walk	916

Los ladridos fueron registrados en un diferente número de sesiones para cada perro. Las locaciones en que fueron grabados los audios fueron las residencias de los dueños, a excepción de los contextos *Alone* y *Fight*. Se digitalizaron las grabaciones a una frecuencia de muestreo de 22.05 kHz y 16 bits por muestra. Originalmente, éstas fueron registradas con una grabadora de cinta y un micrófono. Se reescaló la amplitud de las formas de onda tal que el pico se ubicase en -6 dB.

La base de datos contiene un total de 244 registros de audio con un formato .wav, cuyas longitudes en tiempo se encuentran en el intervalo $[1.03, 378.24]$ segundos. La Tabla 1 muestra la cantidad de audios etiquetados para cada uno de los contextos de la base Mudi. Se observa que la clase mayoritaria (con mayor número de muestras) está dada por el contexto *ball*, mientras que la clase minoritaria se presenta en el contexto *alone*.

2.2. Segmentación automática

Al igual que otras señales de audio, los registros de vocalizaciones presentan partes con silencios, generadas por las pausas entre ladridos. Sin embargo, con el objetivo de discriminar estas partes y conservar los segmentos con ladridos como unidades de análisis, se ha empleado un método de segmentación automática basado en señales de tosidos humanos [7], dado que esta señal presenta similitudes en cuanto a la duración temporal y contenido frecuencial con los ladridos [1].

Para efectos de incrementar la rapidez en el cálculo, el audio del ladrido se submuestreó a 8kHz, permitiendo una visualización frecuencial de las componentes frecuenciales adecuadas para poder detectar las características de la señal [12].

El algoritmo de segmentación está basado en el cálculo de la relación señal a ruido SNR, se compara mediante una *histéresis* las regiones de la forma de onda cuyos picos sufren cambios rápidos en potencia.

La señal deberá ser normalizada en el intervalo $[-1, 1]$, para identificar más rápidamente estos cambios. El cálculo de la SNR se realiza mediante el procedimiento descrito en la Ecuación 1 :

$$SNR = 20 \log_{10} \left(\frac{\sqrt{\frac{1}{|x_s|} \sum_{x(n) \in x_s} x(n)^2}}{\sqrt{\frac{1}{|x_n|} \sum_{x(n) \in x_n} x(n)^2}} \right), \tag{1}$$

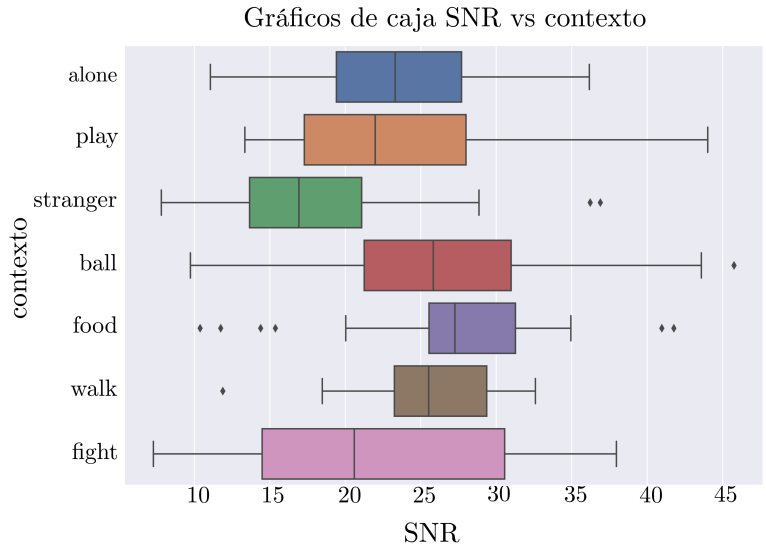


Fig. 2. Gráficos de caja para mostrar la distribución de la SNR por contexto.

donde x_s es una muestra de señal (tosido/ladrido) determinada durante el proceso de segmentación, x_n una muestra de ruido (muestra que no es considerada como señal) y $x(n)$ es la secuencia completa de audio que contiene muestras tanto del tipo x_s como del tipo x_n.

En concreto, la SNR nos indicará el cociente de la energía de los eventos de ladrido y las pausas, teniendo en esta magnitud una referencia para evaluar el desempeño del segmentador y su capacidad de detección de ladridos.

La Figura 1 muestra la segmentación automática aplicada a un segmento de audio que contiene 3 ladridos. Se ha observado que el algoritmo es capaz de detectar diversos tipos de ladridos. Sin embargo, existen otros parámetros necesarios de ajustar para poder obtener resultados más precisos, tales como:

- *padding:* Intervalo de tiempo mínimo entre eventos. Es decir, la longitud de tiempo mínima al inicio y al final de un ladrido. En este caso de acuerdo con lo realizado por Póngracz et al., [12] se estableció en 5 ms (*inter-bark interval*).
- *min_length:* Intervalo de tiempo mínimo a considerar como un evento de ladrido. Es decir, la longitud mínima en tiempo que pudiese tener un ladrido. Se estableció este parámetro en 20 ms.
- *th_l_multiplier:* Umbral de energía mínimo para el comparador de histéresis, en valor de raíz cuadrática media (RMS). Se estableció como 0.2.
- *th_h_multiplier:* Umbral de energía máximo para el comparador de histéresis, en valor de raíz cuadrática media (RMS). Se estableció como 2.

Derivado del proceso de segmentación, se han detectado 6657 eventos de ladrido. La distribución de muestras para cada clase se muestra en la Tabla 2, donde ahora el contexto *stranger* representa la clase mayoritaria, mientras que el contexto *play* la clase minoritaria.

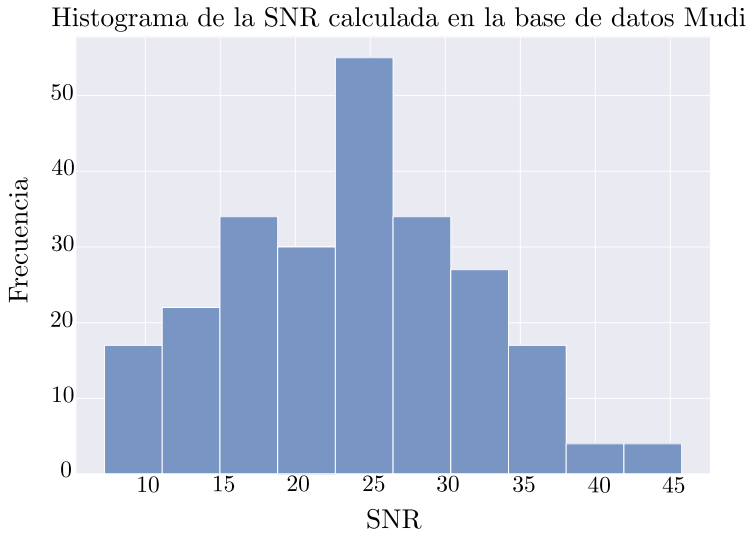


Fig. 3. Histograma general del cálculo de la SNR a cada señal de la base Mudi.

Para verificar el desempeño del algoritmo de segmentación, se calcula la SNR de acuerdo con la ecuación 1, donde se pondera la trama que ha sido detectada como ladrido frente a las pausas o silencios, que son considerados como *ruido*.

La Figura 3 muestra un histograma del cálculo de la SNR a cada audio de la base Mudi. Se observa que el valor de la media se encuentra entre los 15 y 30 dB, además, en cada una de las muestras existieron segmentos de ladrido correctamente detectados, ya existieron cerca de 20 señales con una SNR cercana a 10 dB.

La verificación del desempeño por medio de la SNR también fue distribuída de acuerdo con los contextos y nombres de los individuos de la base de datos Mudi. En la Figura 2 se muestra la distribución de la SNR por contexto, donde se corrobora que el valor de la mediana está localizado entre los 15 y 30 dB, tal y como ocurrió en el histograma general.

De manera particular, los valores más bajos de SNR fueron calculados en el contexto *stranger* y *fight*, esto se debe a los pequeños tiempos de intervalo entre ladridos. De igual manera se observa una más alta variabilidad en la SNR del contexto *fight*, lo cual es también proporcional a la variación alta en las longitudes de semento detectadas para este contexto.

Por otra parte, la Figura 4 muestra la distribución de SNR por cada uno de los 12 individuos. Particularmente se observa una variabilidad alta en la mayoría de los perros, excepto romanecsutka y romannefecske.

2.3. Extracción de características

Las características extraídas para la experimentación de este trabajo están basadas en descriptores de bajo nivel (LLDs), lo cual fue realizado mediante la herramienta openSMILE [3].

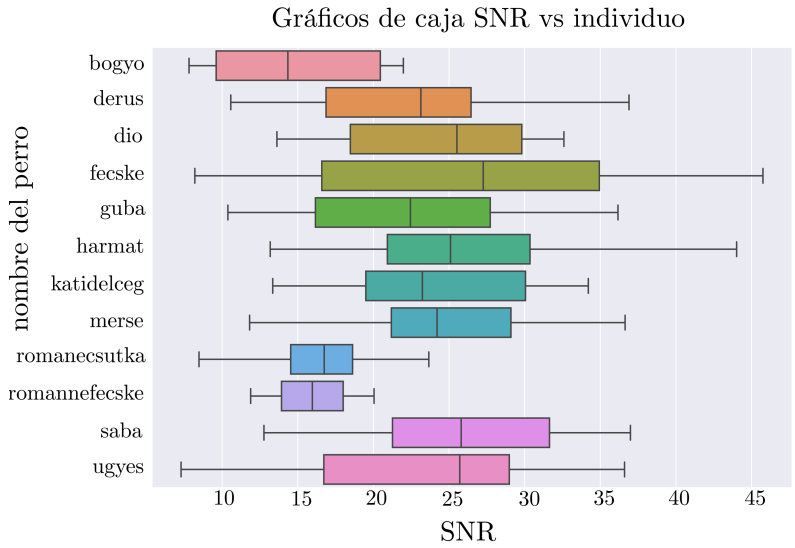


Fig. 4. Gráficos de caja para mostrar la distribución de la SNR por individuo.

Este software presenta tiempos de cómputo muy ágiles y además puede integrarse de manera muy sencilla con el lenguaje de programación Python, el cual ha sido empleado para realizar los experimentos en esta investigación, debido a la popularidad de las librerías integradas para aprendizaje automático, tales como scikit-learn.

Las características seleccionadas están basadas en el conjunto *emolarge*, diseñado para la detección de emociones en la voz humana y además previamente utilizado para la clasificación de vocalizaciones de ladridos de perros por medio de contextos con segmentación de tipo manual [9].

2.4. Selección de características

Por medio del conjunto *emolarge* de openSMILE es posible extraer 6652 parámetros LLDs, usando una ventana de 25 ms y un corrimiento de 10 ms, además del cálculo de coeficientes de regresión delta y doble-delta.

Con el objetivo de establecer una comparación entre el trabajo previo de clasificación con segmentación manual [9] y la segmentación automática propuesta, se ha seleccionado el método *Relief Attribute* como técnica de selección de características, para agilizar el entrenamiento y reducir la dimensionalidad de dicho conjunto. La herramienta Weka [5] se ha empleado para llevar a cabo esta tarea.

En la Tabla 3 se muestra la cantidad de parámetros originales que extrae el conjunto *emolarge* de la librería openSMILE, la cantidad de características reducidas por Pérez et al. [9] haciendo uso de la segmentación manual y finalmente el número de parámetros que se han reducido por medio del método *Relief Attribute* en weka añadiendo el método de segmentación automática propuesto en el presente trabajo. Se observa que los parámetros mayormente seleccionados forman parte de las categorías **mfcc** y **melspec** respectivamente.

Tabla 3. Comparación de las cantidades de parámetros del conjunto *emolarge* de openSMILE, la clasificación realizada con segmentación manual [9] y con segmentación automática (reportada en este trabajo) usando una reducción *Relief*.

Características	**Cantidad de parámetros**		
	emolarge	**Pérez et al.** [9]	**este trabajo**
mfcc	1521	224	196
melspec	3042	66	201
energy	117	37	12
fband	468	8	9
RollOff	468	69	29
Flux	117	4	25
spectralCentroid	117	17	7
MinPos	117	0	3
voiceProb	117	26	15
F0	117	20	3

3. Clasificación de los ladridos

3.1. Métodos de la clasificación

Para conducir los experimentos de la clasificación de contextos a partir de los datos extraídos, se ha empleado el algoritmo de máquinas de soporte vectorial (SVM), el cual ha demostrado buenos resultados para dicha tarea en trabajos previos [9, 10].

Se seleccionó un kernel de tipo *polinomial* para el método de SVM. El conjunto de datos se ha dividido en dos subconjuntos aleatorios de *entrenamiento* y *prueba* considerando el 80 % y el 20 % respectivamente.

Para evitar el sobre-ajuste del modelo, se empleó la técnica de validación cruzada (*CV*), agregando métodos de *estratificación* para asegurar que en cada partición se empleará un número de muestras balanceado por cada clase.

3.2. Calibración de hiperparámetros

Dado que la distribución de características muestra clases que no son linealmente separables y el kernel es de tipo polinomial, la SVM empleada en el algoritmo de clasificación consiste en un híperplano de máximo margen.

Dentro de los hiperparámetros a considerar en este método se encuentran el grado del polinomio y el parámetro C, considerado como un *hiperparámetro de calibración*. A través de C el discriminador controla la cantidad de violaciones que pueden darse en los márgenes del híperplano, así como la severidad de las mismas.

En resumen, mediante C se controla el balance entre el *sesgo* y la varianza del modelo. Para evaluar algunas combinaciones posibles entre el grado del polinomio y la calibración C se ha empleado el método de la búsqueda por cuadrícula mediante validación cruzada (*Grid-Search CV*) por sus siglas en Inglés.

Se han evaluado la métrica *F1-score* y un número de particiones o *folds* $N_{folds} = 10$. Se probaron las combinaciones entre los parámetros $C = [1, 10, 100, 100]$ y $p = [2, 3, 4, 5, 6]$ donde p es el grado del polinomio.

Tabla 4. Comparación de los resultados obtenidos para los experimentos del agrupamiento de contextos.

Núm. de experimento	Pérez et al. [9]	Este trabajo
1	0.85	0.88
2	0.85	0.88
3	0.78	0.82

La mejor combinación de parámetros resultó al seleccionar $p = 3$ y $C = 100$ con una puntuación f1-score promedio de $\mu_{f1} = 0.7435$ y desviación estándar $\sigma_{f1} = 0.0178$. La experimentación se condujo en la herramienta scikit-learn [8].

3.3. Agrupamiento de contextos

Se han definido agrupaciones de contextos de acuerdo con la *valencia* y *excitación* de cada uno de ellos. Estos modelos han sido frecuentemente usados para la detección de emociones humanas [15].

Se conoce como Valencia a la atracción (positiva) o aversión (negativa) intrínseca de cada contexto, mientras que la excitación corresponde al nivel reactivo o sensible del contexto. Con base en el trabajo realizado por Pérez et al. [9], los contextos han sido agrupados de la siguiente manera:

- **Experimento 1:** Valencia Negativa (Fight, Stranger, Alone) vs Valencia Positiva (Walk, Ball, Play, Food).
- **Experimento 2:** Alta Excitación (Fight, Stranger, Wall, Ball, Play) vs Baja Excitación (Alone, Food).
- **Experimento 3:** Valencia Negativa y Alta Excitación (Fight, Stranger) vs Valencia Positiva y Alta Excitación (Walk, Ball, Play) vs Baja Excitación(Alone, Food).

En la Tabla 4 se muestran los resultados producidos tras conducir los experimentos de agrupación de contextos. Se ha observado un aumento en la métrica f1-score en todos los experimentos al introducir en este trabajo la segmentación automática frente a la segmentación manual del algoritmo desarrollado por Pérez et al. [9].

3.4. Clasificación individual de los contextos

Como experimento final, se probó el desempeño de la clasificación con el subconjunto de prueba, correspondiente a una selección aleatoria proporcional al 20 % del conjunto de datos. De igual manera, se consideró la métrica F1, en conjunto con la precisión y el *recall*. Los resultados obtenidos se muestran en la Tabla 5.

4. Conclusiones

En el presente trabajo se ha evaluado el desempeño de la clasificación de los contextos emocionales en los ladridos al introducir un algoritmo de segmentación automática.

Tabla 5. Resultados de la clasificación de contextos en ladridos sobre el subconjunto de prueba.

class	**precision**	**recall**	**f1-score**
alone	0.82	0.76	0.79
ball	0.62	0.65	0.64
fight	0.87	0.86	0.86
food	0.70	0.79	0.74
play	0.71	0.65	0.68
stranger	0.78	0.82	0.80
walk	0.73	0.65	0.69
precisión total	0.75		
macro avg	0.75	0.74	0.74
media ponderada	0.75	0.75	0.75

Dicho algoritmo está basado en la detección de señales auditivas correspondientes a tosidos, cuyo contenido frecuencial y duración en tiempo presentan similitudes con respecto a los ladridos.

Los resultados muestran una mejora en el desempeño, al incrementar la métrica F1-score al hacer una agrupación de contextos de acuerdo a la valencia y a la excitación. Posteriormente se ha determinado que la clasificación presenta los mejores resultados al calibrar hiperparámetros mediante una búsqueda en cuadrícula de la validación cruzada.

Para el clasificador seleccionado, dado por un algoritmo de máquinas de soporte vectorial con kernel polinomial, se ha demostrado que el orden $p = 3$ y el hiperparámetro de calibración $C = 100$ son óptimos para la clasificación, ya que mediante estos se produjeron las métricas f1 más altas.

Se ha comparado el desempeño de la clasificación al introducir la técnica *Relief* para la selección de características. De igual manera, se ha mejorado el desempeño al realizar un agrupamiento de contextos en valencia positiva y negativa, así como excitación alta y baja.

Los resultados muestran un incremento en la métrica F1 de 0.85 a 0.88 al clasificar mediante la valencia, del mismo modo en el agrupamiento mediante excitaciones. Para la tercer experimentación, conjuntando valencia con excitación alta, el F1 ha incrementado de 0.78 a 0.82.

En general, se ha encontrado que los contextos de valencia alta: *alone*, *fight* y *stranger* son más aptos para clasificar mediante los métodos presentados en este trabajo, ya que presentaron métricas F1 de 0.80, 0.84 y 0.79 respectivamente. De igual manera, se ha comprobado la robustez del algoritmo de segmentación automática aplicado, ya que los valores medios de SNR oscilan entre los 15 y 30 dB, teniendo con ello una alta precisión en discriminar las pausas y detectar eficientemente los eventos de ladrido.

Finalmente, se ha evaluado el desempeño del algoritmo al clasificar los contextos del subconjunto de entrenamiento, obteniendo una precisión ponderada de 0.75, resultado cercano a lo obtenido por Perez et al., [9] usando segmentación manual.

De manera similar, se ha comprobado que los contextos de alta valencia como *fight* y *alone* presentan las métricas F1 más altas y por lo tanto mayor potencial de ser detectados mediante los métodos expuestos en este trabajo.

Dados los resultados del presente trabajo mediante diversas experimentaciones, se ha demostrado que la segmentación automática no sólo es una herramienta que contribuye a automatizar el proceso de clasificación emocional de ladridos y eficientar el tiempo, sino que además mejora el desempeño del clasificador.

Como trabajo futuro se aplicará el método de segmentación automática a bases de datos más extensas, con el objetivo de automatizar el proceso y evitar los cuellos de botella en el mismo. Además, se utilizarán técnicas de aprendizaje profundo.

Agradecimientos. El primer autor agradece al proyecto *TZUKU: Métodos computacionales para el análisis del comportamiento de perros de búsqueda y asistencia* por su apoyo en el financiamiento de este trabajo de investigación. También se agradece al Laboratorio de Súper Cómputo del INAOE por facilitar sus recursos para el desarrollo de los experimentos conducidos en este trabajo de investigación, al igual que al Consejo Nacional de Ciencia y Tecnología (CONACYT).

Referencias

1. Chang, A. B.: The physiology of cough. Paediatric Respiratory Reviews, vol. 7, no. 1, pp. 2–8 (2006) doi: 10.1016/j.prrv.2005.11.009
2. Cortes, C., Vapnik, V.: Support-vector network. Machine Learning, vol. 20, pp. 273–297 (1995)
3. Eyben, F., Wöllmer, M., Schuller, B.: Opensmile: The munich versatile and fast open-source audio feature extractor. In: Proceedings of the 18th ACM International Conference on Multimedia, pp. 1459–1462 (2010) doi: 10.1145/1873951.1874246
4. Faragó, T., Takács, N., Miklósi, Á., Pongrácz, P.: Dog growls express various contextual and affective content for human listeners. Royal Society Open Science, vol. 4, no. 170134, pp. 1–11 (2017) doi: 10.1098/rsos.170134
5. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I. H.: The WEKA data mining software: An update. ACM SIGKDD Explorations Newsletter, vol. 11, no. 1, pp. 10–18 (2009) doi: 10.1145/1656274.1656278
6. Molnár, C., Kaplan, F., Roy, P., Pachet, F., Pongrácz, P., Dóka, A., Miklósi, Á.: Classification of dog barks: A machine learning approach. Animal Cognition, vol. 11, pp. 389–400 (2008) doi: 10.1007/s10071-007-0129-9
7. Orlandic, L., Teijeiro, T., Atienza, D.: The COUGHVID crowdsourcing dataset, a corpus for the study of large-scale cough analysis algorithms. Scientific Data, vol. 8, no. 156, pp. 1–10 (2021) doi: 10.1038/s41597-021-00937-4
8. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in python. The Journal of Machine Learning Research, vol. 12, pp. 2825–2830 (2011)
9. Pérez-Espinosa, H., Pérez-Martınez, J. M., Durán-Reynoso, J. Á., Reyes-Meza, V.: Automatic classification of context in induced barking. Research in Computing Science, vol. 100, pp. 63–74 (2015) doi: 10.13053/rcs-100-1-6
10. Pérez-Espinosa, H., Reyes-García, C. A., Villaseñor-Pineda, L.: Acoustic feature selection and classification of emotions in speech using a 3D continuous emotion model. Biomedical Signal Processing and Control, vol. 7, no. 1, pp. 79–87 (2012) doi: 10.1016/j.bspc.2011.02.008

11. Pérez-Espinosa, H., Reyes-Meza, V., Aguilar-Benitez, E., Sanzón-Rosas, Y. M.: Automatic individual dog recognition based on the acoustic properties of its barks. Journal of Intelligent & Fuzzy Systems, vol. 34, no. 5, pp. 3273–3280 (2018) doi: 10.3233/JIFS-169509
12. Pongrácz, P., Molnár, C., Miklósi, Á.: Acoustic parameters of dog barks carry emotional information for humans. Applied Animal Behaviour Science, vol. 100, no. 3-4, pp. 228–240 (2006) doi: 10.1016/j.applanim.2005.12.004
13. Pongrácz, P., Molnár, C., Miklósi, A., Csányi, V.: Human listeners are able to classify dog (canis familiaris) barks recorded in different situations. Journal of Comparative Psychology, vol. 119, no. 2, pp. 136–144 (2005) doi: 10.1037/0735-7036.119.2.136
14. Range, F., Virányi, Z.: Tracking the evolutionary origins of dog-human cooperation: The "Canine Cooperation Hypothesis". Frontiers in psychology, vol. 5, no. 1582, pp. 1–10 (2015) doi: 10.3389/fpsyg.2014.01582
15. Scherer, K. R.: Psychological models of emotion. The neuropsychology of emotion, pp. 137–162 (2000)

ISSN 1870-4069

Método de recomendación para pruebas eléctricas fallidas en la industria de manufactura aplicando aprendizaje automático

Maximiliano Ponce Marquez[1], Samuel González-López[1], Aurelio López-López[2], Guillermina Muñoz-Zamora[1]

[1] Instituto Tecnológico de Nogales, División de Estudios de Posgrado e Investigación, México

[2] Instituto Nacional de Astrofísica, Óptica y Electrónica, Coordinación de Ciencias Computacionales, México

{samuel.gl,M22340764,guilermina.mz}@nogales.tecnm.mx, allopez@inaoep.mx

Resumen. La aplicación de la inteligencia artificial en la industria manufacturera tiene grandes oportunidades. Este artículo presenta la comparación entre diferentes técnicas de aprendizaje automático y aprendizaje profundo para recomendar una posible causa raíz de una falla en un módulo electrónico para vehículos. Se experimentó bajo tres arquitecturas: la plataforma de aprendizaje automático de Microsoft, una red neuronal con Tensorflow y un modelo de clasificación con modelos preentrenados BERT-multilenguaje. Los modelos presentados sugieren una recomendación para el técnico de análisis y no una respuesta absoluta, pero puede ser de ayuda al personal inexperto o en entrenamiento. La herramienta de Microsoft obtuvo una precisión de 51.24 % con el algoritmo de LbfgsMaximumEntropyMulti, la red neuronal alcanzó una precisión del 43.13 %, mientras que el escenario de clasificación multiclase alcanzó un 65.40 %. La selección de estas 3 herramientas se basó en la utilidad y el tiempo de implementación, la herramienta de Microsoft es muy sencilla de utilizar a diferencia de la red neuronal artificial y el uso del modelo BERT. Sin embargo, la herramienta de Microsoft no se puede modificar a diferencia de las otras dos alternativas.

Palabras clave: Método de recomendación, aprendizaje automático, industria manufaturera, detección de causa de falla.

Recommendation Method for Failed Electrical Testing in the Manufacturing Industry Using Machine Learning

Abstract. The application of artificial intelligence in the manufacturing industry has great opportunities. This paper compares different machine learning and deep learning techniques to recommend a possible root cause of a failure in a

vehicle electronics module. It was experimented under three architectures: the Microsoft machine learning platform, a neural network with Tensorflow, and a classification model with pre-trained BERT-multilanguage models. The models presented to suggest a recommendation for the analysis of a technician and not an absolute answer, but they may be helpful to inexperienced personnel or trainees. The Microsoft tool achieved a precision of 51.24% with the LbfgsMaximumEntropyMulti algorithm, the neural network achieved a precision of 43.13%, while the multiclass classification scenario achieved 65.40%. The current selection of these three methods was based on the utility and time implementation, for example, the Microsoft AI wizard is easy to use compared to the artificial neural network programmed with Tensor Flow and the BERT model. However you can not modify the models utilized in the wizard unlike the other two alternatives.

Keywords: Recommendation method, machine learning, manufacturing industry, failure cause detection.

1. Introducción

Dentro de la industria manufacturera de partes para automóviles existen diferentes áreas encargadas de revisar las piezas producidas. Específicamente en el área de Análisis, de una empresa de partes de autos de la ciudad de Nogales Sonora, se encargan de revisar las piezas que fallaron en pruebas eléctricas en el piso de producción. Dado el fallo, el analista revisa el historial de la unidad y con base en la falla, dicho operario es capaz de determinar si es una falla falsa o si es una falla real. En este último caso, es necesario determinar la causa de la falla.

Generalmente en el caso de las fallas reales, se busca el componente en específico que causó la falla, para ello el analista debe usar sus capacidades técnicas y analíticas para determinar la causa de la falla. El analista cuenta con diversas herramientas como multímetro y diagramas eléctricos interactivos para encontrar la falla fácilmente.

Sin embargo, existen fallas que son muy difíciles de identificar y que en ocasiones causan mayores perdidas debido a que pueden salir unidades con la misma falla consecutivamente, es por ello la importancia de poder determinar la falla rápidamente.

Nuestro trabajo se enfocará principalmente en la herramienta de aprendizaje automático de Microsoft, el desarrollo de una red neuronal utilizando Tensorflow con Python en un entorno local y el ajuste con nuestros datos de un modelo pre-entrenado llamado BERT (Representación de Codificador Bidireccional de Transformadores) en su versión multilingual.

Se realiza la comparación entre las tres arquitecturas mencionadas haciendo énfasis en la la métrica Precisión. Cabe mencionar que los datos fuente tienen una variedad alta y desbalanceada de categorías.

2. Trabajo relacionado

El campo de la inteligencia artificial ha tenido grandes avances en los últimos años, en parte por las mejoras en el hardware y los desarrollos e investigaciones en nuevos

algoritmos de inteligencia artificial. En la actualidad se utiliza la inteligencia artificial para la mejora de procesos industriales.

Algunas de las aplicaciones son la optimización de la eficiencia general de los equipos (OEE por sus siglas en inglés) a través de la reparación y mantenimiento predictivo, además de implementaciones en el área de calidad y robótica [8].

Otra de las aplicaciones de la inteligencia artificial en la industria es la reducción de costos, un ejemplo es la implementación de un sistema inteligente que utiliza un sistema de visión e inteligencia artificial para optimizar los cortes en tuberías de metal, en el cual se utilizaron la técnica de análisis de componentes principales (PCA, un controlador PID que interactúa con un modelo de inteligencia artificial basado en máquinas de vectores de soporte (SVM) [7].

En [3] se habla de la posibilidad de poder utilizar los datos generados por una red de sensores para diseñar un modelo predictivo. En [4] los autores desarrollaron un modelo en un ambiente industrial real, el cual consistió en un modelo basado en el algoritmo de aprendizaje Random Forest.

La complejidad de los sistemas de manufactura y la gran oportunidad que la inteligencia artificial ofrece para conseguir productos de mayor calidad es uno de los retos en la actualidad. La obtención de los datos relevantes para entrenar modelos, en algunas empresas, requiere de un preprocesamiento ya que no están estructurados para poder ser utilizados en un modelo de inteligencia artificial directamente [5]. Así el primer paso es el filtrado de la información, es decir obtener la información relevante para la tarea a resolver.

Hay empresas que por peticiones del cliente o por reglas normativas tienen que utilizar o implementar un sistema de trazabilidad de sus productos, para poder detectar la causa raíz de problemas. Estos sistemas ofrecen una gran cantidad de información que puede ser utilizada para construir modelos de predicción.

El trabajo realizado en [6] propone un marco de trabajo de inteligencia artificial utilizando redes bayesianas, tomando el conocimiento de expertos para encontrar desviaciones de calidad y facilitar el análisis de la causa raíz, y se demuestra cómo afecta el conocimiento de expertos en el modelo.

En el trabajo [10] desarrollaron un análisis factorial para determinar la ventana de predicción, que es el tiempo de antelación para prever un fallo. Los autores utilizaron tres algoritmos ML para evaluar la ventana de predicción: bosque aleatorio, máquinas de soporte vectorial y regresión logística.

La falta de datos es un reto en el mantenimiento predictivo, en [9] los autores desarrollaron un método de entrenamiento para transferir el conocimiento aprendido de una falla en otra. En nuestro trabajo, el reto es la gran variedad de los datos que se genera en la industria automotriz.

3. Antecedentes

Anteriormente, dentro de la empresa que se analiza en este trabajo, se desarrolló un sistema web para registrar las fallas que el equipo de análisis encuentra diariamente, para que así posteriormente el analista pueda hacer una búsqueda en el historial de qué componentes han causado cierta falla en específico.

Tabla 1. Datos antes y después del filtrado.

Descripción	Cantidad de datos
Data original	59574
Data filtrada	14311

El desarrollo de este sistema ofreció buenos resultados ya que el personal de reciente ingreso o que no está relacionado con ciertos productos puede buscar fallas fácilmente en el sistema y dar una pista del componente que puede ser la causa de la falla.

No obstante, este sistema aun tenía más oportunidades, con el paso del tiempo la base de datos de esta información fue creciendo, abriendo la posibilidad de crear un modelo para poder recomendar el componente que ocasionó la falla.

4. Propuesta

4.1. Procesamiento de los datos

El primer paso fue el pre-procesamiento de los datos antes de entrenar el modelo. Inicialmente existían 59,754 entradas en la base de datos, pero no todos estos datos son válidos para el modelo. En particular, las pruebas relacionadas con la soldadura de las unidades resultaron inútiles, ya que no proveen información de valor para poder determinar la causa raíz de la falla.

Adicionalmente hay pruebas de unos equipos en especial, llamados ICT (In Circuit Test, por sus siglas en ingles) que ya dan directamente la respuesta, por lo que ya no es necesario predecir estas fallas. También se extrajeron solo aquellos resultados en los que se registró un solo componente, esto debido a que es mas sencillo poder categorizar esta información y acelerar el procesamiento para el modelo.

El dataset cuenta con columnas que ofrecen información del modelo, familia del producto, número de serie, estación en la que se probó la unidad, descripción de la falla, valor de la prueba, componente que ocasiono la falla y un comentario que el analista agrega de manera opcional.

Para este trabajo se decidió trabajar con las columnas de descripción de la prueba eléctrica, valor numérico de la prueba, causa de la falla o componente y comentario, ya que con estos datos es suficiente para poder ofrecer un resultado.

Cada producto tiene su propio grupo de pruebas, por lo que no hay problema al omitir la columna del modelo en este trabajo. Con estos datos se plantearon tres escenarios de experimentación:

– En el primer escenario se utilizaron las columnas TestDescription, Value como entradas y Component como salida. Para esto se implementó la herramienta de Aprendizaje Automático de Microsoft.
– Posteriormente se realizó con la misma configuración del escenario uno, la utilización de una red neuronal con TensorFlow.
– Con la finalidad de obtener una alternativa de recomendación, se configuró el tercer escenario bajo un esquema de agrupamiento, donde se identificaron aquellos comentarios que tuvieran similitud. Por ejemplo: 'Arrancado pad dañado' y 'c1418 capacitor dañado' se encuentran en el mismo grupo o categoría.

Tabla 2. Ejemplo del conjunto de datos pre-procesado.

TestDescription	Value	Component	Comment
504090_AI_P0 Pull up FB	3.828	IC101	IC101 Corte por soldadura
504090_AI_P0 Pull up FB	3.94	IC101	IC101 Corte por soldadura
504090_AI_P0 Pull up FB	3.94	IC101	IC101 Corte por soldadura
503020_LIN Voltage	3.647	D401	D401 Danado
502020_PCHK X400_3 LED_CATHODE	36.34	D401	D401 Danado
509050_BST CHK Phase 3	36	R411	R411 Contaminado insuficiencia

Asi obtuvimos 10 categorías o grupos. Posteriormente se ajustó un modelo de BERT. En este escenario se busca predecir la categoría dependiendo de la prueba realizada. Así, el operador puede revisar la categoría predecida para ver posible causa-raíz.

4.2. Construcción del modelo utilizando herramienta de aprendizaje automático de Microsoft

Una vez obtenido un conjunto de datos limpio y listo, se optó primeramente por utilizar la herramienta de Microsoft disponible en Visual Studio 2022. Esta herramienta permite seleccionar un caso de uso, un dataset y en base a ello construir un modelo de inteligencia artificial.

Como se mencionó anteriormente se utilizaron las columnas de descripción de la falla y el resultado de la prueba como entrada para el modelo y la columna de componente como salida.

La herramienta de Microsoft se encarga por completo de construir el modelo, en el que va comparando diferentes modelos de aprendizaje de máquina preestablecidos y al final seleccionar el que tuvo el mejor porcentaje de aciertos.

En este proyecto nos enfocamos en la métrica de precisión ya que el modelo predice los resultados en base al historial registrado en la base de datos de análisis, y resulta imposible poder predecir una falla que nunca se ha registrado en la base de datos ya que es un proceso muy complejo que requeriría de muchísima mas información, como la construcción de la unidad, los circuitos eléctricos, interconexiones, envío de mensajes, etc. Además, son muchos productos los que son analizados.

Resultados del modelo de Microsoft. Se realizaron 13 iteraciones con diferentes modelos, obteniendo los siguientes resultados:

De las 13 iteraciones, estas son las 5 con los mejores resultados:

El mejor modelo que se pudo entrenar fue el de la iteración 11 con una precisión del 51.24 %.

Tabla 3. Resultados del modelo de Microsoft - 13 iteraciones.

Modelo	Precisión	Tiempo de entrenamiento (s)	#Iteración
SdcaMaximumEntropyMulti	0.3190	12,1	0
SdcaLogisticRegressionOva	0.0705	130.8	1
FastForestOva	0.4614	142.5	2
LightGbmMulti	0.36	4.6	3
LbfgsMaximumEntropyMulti	0.4936	159.8	4
FastForestOva	0.4674	151.6	5
FastTreeOva	0.4835	211.8	6
SdcaMaximumEntropyMulti	0.3559	11.4	7
LbfgsLogisticRegressionOva	0.4936	54.9	8
SdcaLogisticRegressionOva	0.3418	123.7	9
LightGbmMulti	0.0981	5.2	10
LbfgsMaximumEntropyMulti	0.5124	320.1	11
FastForestOva	0.4527	179	12

Tabla 4. 5 Mejores resultados del modelo de Microsoft.

Modelo	Precisión	Tiempo de entrenamiento (s)	#Iteración
LbfgsMaximumEntropyMulti	0.5124	320.1	11
LbfgsLogisticRegressionOva	0.4936	54.9	8
LbfgsMaximumEntropyMulti	0.4936	159.8	4
FastTreeOva	0.4835	211.8	6
FastForestOva	0.4674	151.6	5

Cabe mencionar que en este primer escenario la cantidad de componentes que pueden ser la causa-raíz son muchos, por que creemos que ese comportamiento nos produce un resultado relativamente bajo.

4.3. Desarrollo de red neuronal con TensorFlow

También se optó por implementar una red neuronal utilizando la librería Tensorflow en Python. Para empezar se importaron las siguientes librerías:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import tensorflow as tf
```

Después se importó el conjunto de datos de la siguiente manera:

```
dataset = pd.read_csv('filtered-dataset.csv')
X = dataset.iloc[:,:-1].values
y = dataset.iloc[:,-1].values
```

Importar el conjunto de datos no es suficiente, en el caso de la herramienta de Microsoft lo hacia por nosotros, pero ahora sera necesario categorizar la informacion de forma manual. Se utilizó la libreria de Scikit learn para esta tarea con las siguientes instrucciones:

```
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder

ctPruebas = ColumnTransformer(transformers=
  [('encoder', OneHotEncoder(sparse=False),[0])],remainder='passthrough')
X = np.array(ctPruebas.fit_transform(X))

ctUnidadesMedida = ColumnTransformer(
  transformers=[('encoder',
      OneHotEncoder(sparse=False),
      [-2])],remainder='passthrough')
X = np.array(ctUnidadesMedida.fit_transform(X))

ctPruebas = ColumnTransformer(transformers=[('encoder',
OneHotEncoder(sparse=False), [0])],remainder='passthrough')
wy = np.array(ctPruebas.fit_transform(y.reshape(-1,1)))
```

Estas instrucciones convierten las columnas categorías en una colección de 1s y 0s, por ejemplo, la prueba '503010 Current Consumption503010 Current Consumption' pudo ser transformada a un arreglo como [0,0,0,1,0,1]. En este caso, dado que hay 1853 fallas y 521 componentes posibles, los arreglos serán de los tamaños correspondientes.

Esto se puede comprobar con el código siguiente:

```
print(X[0].shape)
print(y[0].shape)
```

Después se divide el conjunto de datos en un conjunto de entrenamiento y un conjunto de pruebas, con una proporción de 80/20.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=0)
```

Posteriormente se hace una normalización de los datos, especialmente para los valores de las pruebas, para que todos estén en el rango de 0 a 1

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

Ahora que el conjunto de datos está listo, se comienza a construir la arquitectura de la red neuronal, en este caso se utilizó la siguiente selección de capas en la red neuronal artificial. A continuación, se presenta el código para construir la red neuronal.

```
ann = tf.keras.models.Sequential()
ann.add(tf.keras.layers.Dense(units = 1853, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 2000, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 2000, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 3000, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 3500, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 3500, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 2000, activation='relu'))
```

```
ann.add(tf.keras.layers.Dense(units = 2000, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 1500, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 1000, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 1000, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 800, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 800, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 700, activation='relu'))
ann.add(tf.keras.layers.Dense(units = 521, activation='softmax'))
```

Observe que se utilizó la función de activación ReLU para todas las capas con excepción de la capa de salida en la que se optó por softmax, esto a que es mas adecuado para datos categóricos como en este caso.

Finalmente se entrenó el modelo con los siguientes parámetros:

```
ann.compile(optimizer = 'adam', loss = 'categorical_crossentropy',
metrics = ['accuracy'])
ann.fit(X_train, y_train, batch_size = 32, epochs = 500)
```

El modelo descrito obtuvo una precisión de 43.13 %, un poco menor a los resultados obtenidos con la herramienta de Microsoft.

4.4. Implementación del modelo BERT-multilingual

A continuación mostramos algunas secciones de la codificación para lograr la tarea de clasificación multiclase.

```
import tensorflow as tf
print(tf.__version__)
import pandas as pd
df = pd.read_excel('data.xlsx')
df.head()
```

En la siguiente figura se muestra la distribución de las 10 categorías. Se puede observar que las categorías 3 y 6 tienen mayor cantidad de ejemplos (ver Fig. 1).

A continuación se muestra la sección del código para calcular las métricas (en este caso Precisión y Exactitud):

```
n_epochs = 10
METRICS = [
      tf.keras.metrics.CategoricalAccuracy(name="Acccuracy"),
      tf.keras.metrics.Precision( name="Precision"),
      balanced_recall,
      balanced_precision,
      balanced_f1_score
]
earlystop_callback = tf.keras.callbacks.EarlyStopping(monitor =
"val_loss", patience = 3, restore_best_weights = True)
model.compile(optimizer = "adam",
              loss = "categorical_crossentropy",
              metrics = METRICS)
model_fit = model.fit(x_train,
                      y_train,
                      epochs = n_epochs,
```

```
                        validation_data = (x_test, y_test),
                        callbacks = [earlystop_callback])
```

Los resultados en 10 épocas lo podemos encontrar en la siguiente tabla:
El valor mas alto para la precision se alcanza en el época 9 con un valor de 0.6540.

5. Resultados

Generalmente se utiliza la métrica de recuerdo o recall cuando queremos que el modelo detecte tantos casos reales como sea posible, y se utiliza la métrica de precisión o exactitud (accuracy) cuando necesitamos que el modelo sea lo mas correcto posible. En este caso debido a que se busca un modelo de recomendación y que está basado totalmente en el historial generado por el personal, se busca obtener la mejor precisión posible. Recordemos que en este modelo, predecir una falla que no está registrada en el conjunto de datos de entrenamiento no ofrecerá una respuesta válida ya que no hay forma de relacionar la respuesta.

La herramienta de Microsoft obtuvo una precisión de 51.24 % con el algoritmo de LbfgsMaximumEntropyMulti, la red neuronal personalizada una precisión del 43.13 %, mientras que el escenario de clasificacion multiclase alcanzó un 65.40 %.

Los otros algoritmos utilizados por la herramienta de Microsoft arrojaron unos resultados muy similares, sin embargo, su desventaja es que no permite realizar muchas configuraciones. La herramienta automáticamente va actualizando los parámetros y seleccionando otros algoritmos de acuerdo con los resultados y después de un determinado tiempo selecciona el que condujo al mejor resultado.

El otro enfoque, utilizando una red neuronal codificada desde 0, dió un resultado menor, pero permite ser configurado completamente, pudiendo cambiar la forma en que se preprocesan los datos e incluso cambiar la arquitectura de red.

Finalmente creemos que el escenario tres muestra lo que pudiera ser una solución, siempre y cuando se realice un filtrado en diferentes categorías por un experto. Tambien es necesario buscar el balanceo de las clases, algo complicado en la industria donde hay datos irregulares.

Los resultados parecen ser no muy favorecedores, pero aún hay oportunidades de mejora. Es posible enfocarse en un solo producto y utilizar más datos del sistema de producción. Sin embargo, con estos resultados el analista puede obtener una recomendación que le podría ayudar en su proceso de búsqueda de fallas.

6. Conclusiones

En otros trabajos relacionados con inteligencia artificial en la manufactura se obtiene resultados mucho mayores como el 99.95 % de precisión obtenido en el sistema de cortes para minimizar costos [7], sin embargo, la naturaleza de los sistemas es distinto.

Se decidio utilizar estas tres herramientas de manera exploratoria, principalmente por el tiempo que se puede utilizar para construir y entrenar el modelo.

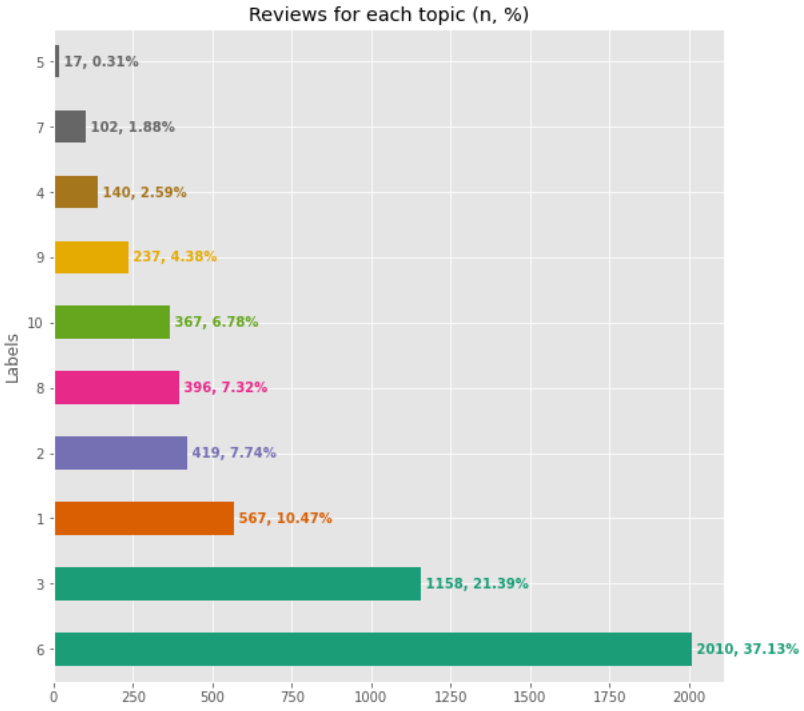


Fig. 1. Distribución de las clases.

Tabla 5. 10 épocas del modelo BERT-multilingual.

Época	Precisión	Exactitud
1	0.5140	0.3851
2	0.5856	0.4437
3	0.5986	0.4708
4	0.6215	0.4846
5	0.6208	0.4900
6	0.6276	0.5068
7	0.6354	0.5060
8	0.6407	0.5159
9	0.6540	0.5275
10	0.6442	0.5150

Recordemos que con forme pasan los días se van registrando nuevas fallas o componentes generadores de fallas, por lo que es necesario entrenar nuevamente el modelo de nuevo para que sea capaz de predecir o dar resultados en base a estas nuevas fallas.

En el caso del problema presentado en este trabajo se basa completamente en el historial generado por el personal de análisis, además, existe una gran variedad de fallas posibles para una gran diversidad de productos o modelos diferentes.

Los resultados obtenidos por los tres modelos reportados demuestran la complejidad del problema, sin embargo, es el inicio para explorar otras alternativas.

Es posible, por ejemplo, implementar un sistema mas robusto para obtener la información y probar otros algoritmos de aprendizaje. En base a los resultados se puede observar que en algunas ocasiones las soluciones mas sencillas son las mas adecuadas. Sin embargo, es importante realizar la exploración y experimentación.

En este caso un algoritmo de regresión logística multinomial ofreció un mejor resultado que una red neuronal programada con Tensor Flow. Se deben atacar las problemas con las herramientas adecuadas, pero para ello es necesario conocer el repertorio de herramientas que hay disponibles.

Agradecimientos. El primer autor es apoyado por Conacyt, becario 1244665.

Referencias

1. Ponce-Cruz, P.: Inteligencia artificial: Con aplicaciones a la ingeniería. Alpha Editorial (2010)
2. Ertel, W.: Introduction to artificial intelligence. Undergraduate Topics in Computer Science, Springer International Publishing (2017)
3. Sittón, I., Hernandez, E., Rodríguez, S., Santos, M. T., González, A.: Machine learning predictive model for industry 4.0. Knowledge Management in Organizations (KMO 2018), Communications in Computer and Information Science, vol. 877 (2018) doi: 10.1007/978-3-319-95204-8_42
4. Paolanti, M., Romeo, L., Felicetti, A., Mancini, A., Frontoni, E., Loncarski, J.: Machine learning approach for predictive maintenance in industry 4.0. In: 14th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA), pp. 1–6 (2018) doi: 10.1109/MESA.2018.8449150
5. Wuest, T., Weimer, D., Irgens, C., Thoben, K. D.: Machine learning in manufacturing: Advantages, challenges, and applications. Production and Manufacturing Research, vol.4, no. 1, pp. 23–45 (2016) doi: 10.1080/21693277.2016.1192517
6. Lokrantz, A., Gustavsson, E., Jirstrand, M.: Root cause analysis of failures and quality deviations in manufacturing using machine learning. Procedia CIRP, vol. 72, pp. 1057–1062 (2018) doi: 10.1016/j.procir.2018.03.229
7. Acosta, P., Terán, H., Arteaga, O., Terán, M.: Machine learning in intelligent manufacturing system for optimization of production costs and overall effectiveness of equipment in fabrication models. Journal of Physics: Conference Series, vol. 1432 (2020) doi: 10.1088/1742-6596/1432/1/012085
8. Ng-Corrales, L. C., Lambán, M. P., Hernández-Korner, M. E., Royo, J.: Overall equipment effectiveness: Systematic literature review and overview of different approaches. Applied Sciences, vol. 10, no. 18, pp. 6469 (2020) doi: 10.3390/app10186469
9. Li, Z., Kristoffersen, E., Li, J.: Deep transfer learning for failure prediction across failure types. Computers and Industrial Engineering, vol. 172 (2022) doi: 10.1016/j.cie.2022.108521
10. Leukel, J., González, J., Riekert, M.: Machine learning-based failure prediction in industrial maintenance: Improving performance by sliding window selection. The International Journal of Quality and Reliability Management, vol. 40, no. 6 (2022) doi: 10.1108/IJQRM-12-2021-0439

ISSN 1870-4069

Implementación de probabilidades a una ontología para la búsqueda de objetos cotidianos del hogar por un robot de servicio

Nayely Morales-Ramírez, Antonio Marín-Hernández,
Alejandro Guerra-Hernández

Universidad Veracruzana,
Instituto de Investigaciones en Inteligencia Artificial,
México

nayely0199@gmail.com, {anmarin, aguerra} @uv.mx

Resumen. Los robots de servicio se han desarrollado como una herramienta de apoyo que se pueden emplear en el ámbito doméstico para facilitar la vida cotidiana de las personas. Dentro de las actividades que una persona realiza diariamente, inherentemente está la búsqueda de objetos. Para la solución de la problemática de la búsqueda de objetos, en este trabajo se propone emplear una ontología con las relaciones de los objetos y lugares dentro del hogar, incorporando a ella probabilidades, esto con el fin de determinar los lugares más viables en donde se puedan encontrar los objetos. Estas probabilidades se modificarán a lo largo de los experimentos debido a que se irán adaptando a los resultados obtenidos. Para validar la propuesta se realizaron varios experimentos en simulación de varios ambientes domésticos con un robot de servicio de manejo diferencial, en donde se hicieron búsquedas por cada objeto, dando como resultado adaptaciones en las probabilidades de la red semántica inicial. Como resultados preliminares se comprueba que usando la red semántica se realiza la búsqueda inteligente de manera exitosa, ya que logra ubicar a los objetos objetivos en una menor cantidad de movimientos.

Palabras clave: Ontología, red semántica, robot de servicio, búsqueda de objetos.

Implementation of Probabilities to an Ontology for the Search of Everyday Household Objects by a Service Robot

Abstract. Service robots have been developed as a support tool that can be used at home to make people's daily lives easier. Within the activities that a person performs daily, inherently is the search for objects. For the solution of the problem of the search for objects, in this work it is proposed to use an ontology with the relationships of objects and places within the home, incorporating probabilities in it, in order to determine the most viable places where the objects can be found. These probabilities are modified throughout the experiments because they will be

adapted to the results obtained. To validate the proposal, several experiments were carried out in simulation of various domestic environments with a differential handling service robot, where searches were made for each object, resulting in adaptations in the probabilities of the initial semantic network. As preliminary results, it is verified that using the semantic network, the intelligent search is carried out successfully, since it manages to locate the objective objects in a smaller number of movements.

Keywords: Ontology, semantic network, service robot, object search.

1. Introducción

El desarrollo de la robótica ha tenido un crecimiento exponencial, tanto que, en muy poco tiempo se convivirá con los robots de manera cotidiana; y en un futuro cercano, se espera tener una interacción más natural con ellos. Esto será de ayuda apoyando a las personas en sus actividades cotidianas. Esta interacción será de gran ayuda, al permitir un apoyo más cercano a las personas en sus actividades cotidianas.

En la actualidad la robótica ha tenido grandes avances diseñando herramientas de apoyo en diversos dominios, particularmente, es con la ayuda de distintas especialidades como: la mecánica, la electrónica, el control y la informática que se pueden generar soluciones a problemas cotidianos que anteriormente parecían ficción.

Sin embargo, para problemas y soluciones más complejas, se debe potenciar con otras ramas del conocimiento, como pueden ser la inteligencia artificial o la visión artificial.

Pensando en la forma en que los robots incursionan en la vida cotidiana, surge la necesidad de encontrar una herramienta de apoyo en esta área, es aquí donde se encuentran los robots de servicio.

Se puede describir al robot de servicio como un robot semiautónomo o totalmente autónomo que opera para el bienestar de los humanos y equipos [1]. Se categorizan según su rango de autonomía: los semiautónomos, son aquellos que requieren en determinadas ocasiones intervención humana, y los completamente autónomos, que no requieren de dicha intervención (al menos para ciertas tareas).

Dentro de los robots de servicio encontramos los robots de servicio personal o domésticos, que son aquellos diseñados para el mantenimiento del hogar, ofrecer ayuda con tareas domésticas y hasta entretener a la familia.

El propósito de un robot de servicio es ayudar a los humanos. Estos robots son sistemas diseñados para realizar tareas consideradas peligrosas y también actividades o trabajos repetitivos. Algunas de estas actividades son: la manipulación de objetos, los trabajos de limpieza, la organización de artículos en estantes, es decir, los robots que no realizan ninguna actividad industrial o de manufactura [11].

Una de las necesidades actuales dentro del ambiente doméstico es la búsqueda de objetos cotidianos. Diferentes propuestas se han realizado a lo largo del mundo, en donde, para realizar la búsqueda de objetos se realiza un escaneo exhaustivo de la zona, sin embargo, esto aún dista de la búsqueda que haría un humano en estos ambientes.

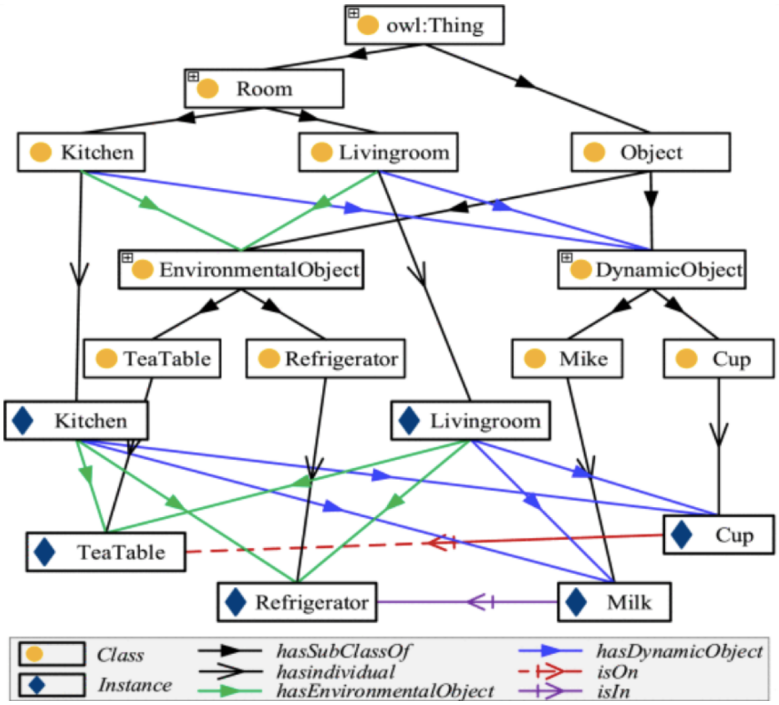


Fig. 1. Ejemplo de esquema de modelo semántico.

En este trabajo se propone realizar la búsqueda inteligente incorporando probabilidades a una ontología del ambiente doméstico. Una ontología es aquella representación que expone las relaciones entre los entes, que es todo aquello tangible, o la relación entre un acto y sus participantes.

De esta manera, en este trabajo se propone generar conexiones entre tres conceptos, los cuales son: habitaciones, objetos referencia y objetos objetivos, para encontrar de manera eficaz el plan de búsqueda de objetos cotidianos aumentando la tasa de éxito.

Un aspecto importante a tomar en cuenta en el presente trabajo, es que se está considerando que la búsqueda de objetos no está limitada a pensar que estos se encuentran en el mismo lugar siempre, sino que, estos cambian de ubicación conforme se interactúa en el ambiente; especialmente si hablamos de ambientes tipo hogar donde los habitantes hacen usos de los objetos de manera cotidiana modificando su localización.

2. Trabajos relacionados

Particularmente atendiendo la problemática de la búsqueda de objetos, distintos autores han dado solución mediante técnicas diversas, de manera paulatina con los avances que se van realizando. A continuación se analizarán algunos de los trabajos más relevantes en el área.

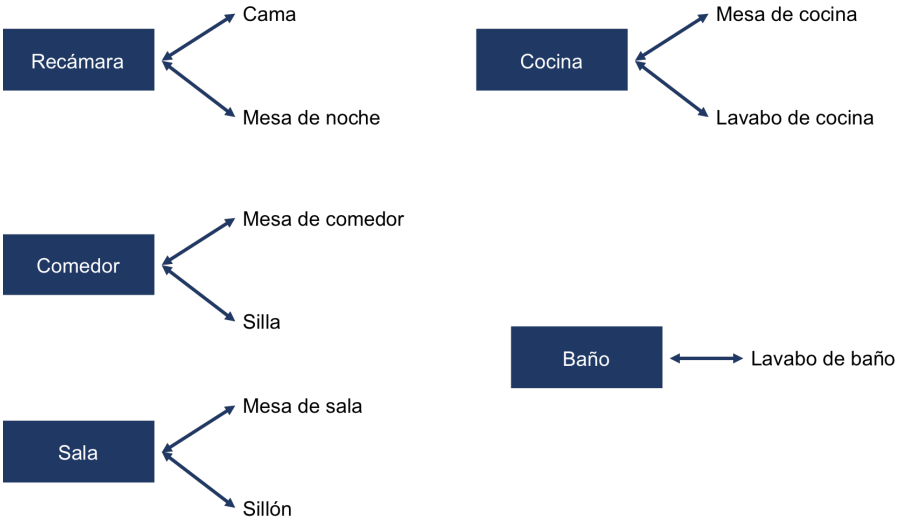


Fig. 2. Relaciones objeto-lugar, objetos estáticos con habitaciones.

Kemps [8] menciona que los robots han tenido mucho éxito en la manipulación en entornos controlados y de simulación, así como presenta sus limitantes dentro de los ambientes humanos y su capacidad de aprendizaje.

Un ejemplo más reciente del trabajo de los robots de servicio es el de Veloso [14], donde utiliza robots colaborativos navegando de forma autónoma en edificios. Él propone una nueva localización episódica, es decir, una ubicación en cada área definida como episodios, que genera una movilidad confiable.

Una rama importante sobre la que se ha estado trabajando con ayuda de los robots de servicio es la búsqueda de objetos, partiendo de un ambiente bidimensional para después trasladarse a uno tridimensional. Se puede abordar la búsqueda de objetos dependiendo de distintas características como, por ejemplo, Arévalo[3] que reconoce objetos por color y forma.

De otra manera, Shubina [12] maneja la probabilidad de encontrar el objeto dando un límite de costo fijo en términos del número total de acciones robóticas requeridas para encontrar el objeto visual.

Una forma distinta de enfrentar el problema de la búsqueda de objetos es modelarlo como un Proceso de Decisión de Markov Parcialmente Observable o POMDP. Una forma de aplicar esta manera es realizando una búsqueda visual activa de objetos robusta y a gran escala como en el trabajo de Aydemir [5]. Otra opción es realizando la búsqueda de objetos en un ambiente desconocido, empleando un mapa de ruta de creencias. Este mapa está basado en co-ocurrencia de objetos y lugares para una planificación eficiente de rutas de búsqueda de objetos, como lo planteó Wang [15].

Haciendo uso de las co-ocurrencias objeto-objeto y objeto-lugar, nace el término búsqueda indirecta, es decir, emplear objetos intermedios o de referencia para encontrar nuestro objeto final.

Fig. 3. Relaciones objeto-objeto, objetos estáticos con objetos dinámicos.

Se desarrollaron varios trabajos siguiendo la misma idea, como por ejemplo, el de Kollar [9] que utiliza un modelo probabilístico sobre posibles ubicaciones de objetos empleando el contexto objeto-objeto y objeto-escena.

Una alternativa es trabajar con relaciones espaciales cualitativas tales como: *muy cerca*, *cerca*, *lejos*, *dentro*, *sobre*, *delante de*, *a un lado*. Distintos autores nos muestran como estas relaciones espaciales pueden usarse como una forma de guiar la búsqueda visual de objetos.

Esto proporciona un enfoque para realizar una búsqueda indirecta en el que, el robot puede hacer uso de relaciones espaciales conocidas o supuestas entre los objetos. Esta técnica aumenta significativamente la eficiencia de la búsqueda al momento de ubicar primero un objeto intermedio que sea más fácil de encontrar, generalmente estos objetos son de un tamaño mayor que el objeto principal.

En el trabajo propuesto por Zeng [16], hace la introducción a los mapas de enlaces semánticos, él basa su proyecto explotando el conocimiento previo sobre las relaciones espaciales comunes entre los puntos de referencia y los objetos destino.

Tabla 1. Tabla con los porcentajes de las relaciones objeto-objeto.

Objeto	Objeto R.	Porcentaje	Objeto	Objeto R.	Porcentaje
Vaso	M. Cocina	0.25	Celular	M. Cocina	0.05
Vaso	M. Com.	0.2	Celular	M. Com.	0.01
Vaso	M. Sala	0.1	Celular	M. Sala	0.2
Vaso	Cama	0.05	Celular	Cama	0.2
Vaso	M. Noche	0.1	Celular	M. Noche	0.25
Vaso	L. Baño	0.05	Celular	L. Baño	0.01
Vaso	L. Cocina	0.15	Celular	L. Cocina	0.01
Vaso	Silla	0.05	Celular	Silla	0.05
Vaso	Mueble	0.05	Celular	Mueble	0.13
Libro	M. Cocina	0.05	E. lentes	M. Cocina	0.1
Libro	M. Com.	0.1	E. lentes	M. Com.	0.15
Libro	M. Sala	0.25	E. lentes	M. Sala	0.17
Libro	Cama	0.2	E. lentes	Cama	0.15
Libro	M. Noche	0.25	E. lentes	M. Noche	0.2
Libro	L. Baño	0.01	E. lentes	L. Baño	0.05
Libro	L. Cocina	0.01	E. lentes	L. Cocina	0.03
Libro	Silla	0.03	E. lentes	Silla	0.05
Libro	Mueble	0.1	E. lentes	Mueble	0.1
Pastillero	M. Cocina	0.1	Pastillero	Mueble	0.04
Pastillero	M. Com.	0.2	Pastillero	Silla	0.01
Pastillero	M. Sala	0.1	Pastillero	L. Cocina	0.05
Pastillero	Cama	0.1	Pastillero	L. Baño	0.2
Pastillero	M. Noche	0.2			

Zhang [17] busca emplear redes semánticas, en este trabajo se estableció un modelo probabilístico general y un modelo semántico(Figura 1). Estos modelos se crearon investigando las relaciones de ubicación espacial típicas entre el objeto y el tipo de habitación.

El fin de crear dos modelos es guiar al robot para priorizar el esfuerzo de búsqueda de los espacios que son más prometedores para encontrar el objeto objetivo. Proyectado como un esquema de conexión a tierra semántica eficaz para robots móviles a largo plazo para la búsqueda dinámica de objetos en entornos domésticos abiertos/dinámicos.

Con los trabajos analizados previamente, se aprecia que la búsqueda de objetos se ha resuelto de distintas maneras. Por ejemplo, Arévalo [3] y Shubina [12] utilizan técnicas por características físicas del objeto y costo de acciones, mientras que Aydemir [5] y Wang[15] lo realizan por medio de POMDP, trabajos que distan de lo realizado en este proyecto. Trabajos que se asemejan a lo que se desarrolla en esta propuesta son los de Zeng[16] y Zhang [17], debido a que hacen uso de mapas y enlaces semánticos, con relaciones de objetos y lugares.

Después del análisis presentado se establece que si bien el uso de ontologías se ha utilizado para la búsqueda de objetos, no se han empleado probabilidades dentro de dicha ontología. Este trabajo contribuye sustancialmente al proponer una solución en dónde se establece la conjunción de ambos para una búsqueda eficaz.

Tabla 2. Tabla con los porcentajes de las relaciones objeto-lugar.

Habitación	**O. Destino**	**Porcentaje**	**Habitación**	**O. Destino**	**Porcentaje**
Recámara	Celular	0.45	Recámara	Vaso	0.15
Sala	Celular	0.33	Sala	Vaso	0.15
Comedor	Celular	0.15	Comedor	Vaso	0.25
Cocina	Celular	0.06	Cocina	Vaso	0.4
Baño	Celular	0.01	Baño	Vaso	0.05
Recámara	E. lentes	0.35	Sala	Libro	0.45
Sala	E. lentes	0.27	L. room	Libro	0.35
Comedor	E. lentes	0.2	Comedor	Libro	0.13
Cocina	E. lentes	0.13	Cocina	Libro	0.06
Baño	E. lentes	0.05	Baño	Libro	0.01
Recámara	Pastillero	0.3	Cocina	Pastillero	0.15
Sala	Pastillero	0.14	Baño	Pastillero	0.2
Comedor	Pastillero	0.21			

3. Descripción de las redes semánticas

Los mapas conceptuales, también llamados mapas o redes semánticas, sirven para describir y comunicar los conceptos que los sujetos tienen incorporados en su memoria y se alimentan de la teoría constructivista de la asimilación [4]. Las redes semánticas son un tipo de representación de datos que incorpora información lingüística que describe conceptos u objetos y la relación o dependencia entre ellos.

Las redes semánticas son esquemas de representación del conocimiento que involucran nodos y enlaces (arcos o flechas) entre nodos. Los nodos representan objetos o conceptos o situaciones y los enlaces representan relaciones entre nodos. Los enlaces están dirigidos y etiquetados; por tanto, una red semántica es un grafo dirigido. Los nodos suelen estar representados por círculos o cajas y los enlaces se dibujan como flechas entre los círculos.

Algunos de los arcos más comunes son del tipo es-un o tiene-un. Es-un se usa para mostrar la relación de clases; es decir, que un objeto pertenece a una clase o categoría mayor de objetos. Los enlaces tiene-un se utilizan para identificar características o atributos de los nodos de objetos. Otros arcos se utilizan con fines de definición. Las redes semánticas pueden mostrar herencia, son una representación visual de las relaciones y se pueden combinar con otras representaciones [2].

Considerando el uso de una red semántica como representación del conocimiento, el uso de una ontología pretende responder a desafíos referentes a compartir el conocimiento en común. Esto quiere decir ponerse de acuerdo sobre la información que conformará el conocimiento y brindar dicho conocimiento de una manera fácil de comprender para los robots.

La primera definición de ontología ha sido propuesta por Guber [7] como: “una ontología es una especificación explícita de una conceptualización”. Posteriormente, Borst [6], ha introducido dos nuevos conceptos que definen una ontología como una “especificación formal de una conceptualización compartida”. Los nuevos conceptos son las nociones de “formal” y “compartido”.

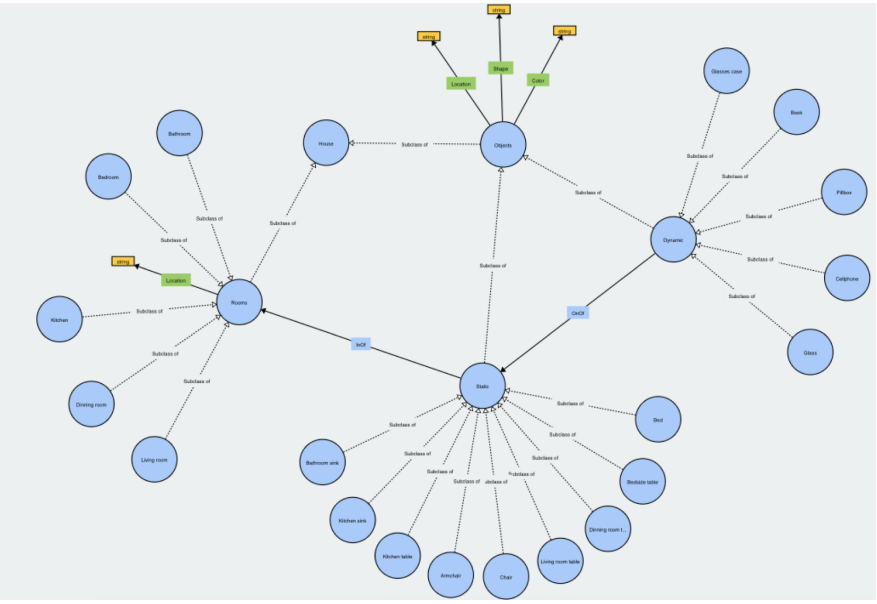


Fig. 4. Red de objetos y habitaciones.

Combinando estas dos definiciones, Studer [13] llegó a la definición completa donde dice que: “Una ontología es una especificación formal y explícita de una conceptualización compartida”. Con esta última definición, podemos observar que una ontología puede ser una herramienta poderosa para crear una Base de conocimiento (KB) común a toda una arquitectura y empleada para el sistema general de la red semántica.

4. Probabilidades y redes semánticas

En esta sección daremos los detalles de la metodología a implementar para la incorporación de las probabilidades en una ontología. Comenzamos definiendo a nuestros actores, las relaciones existentes entre ellos, así como la probabilidad asociada a estas relaciones. Posteriormente, se crea la red semántica incorporando la información definida previamente.

4.1. Relaciones y probabilidades

Dentro del desarrollo de la red semántica, primeramente, se definen los actores a intervenir. Se generan dos clases generales dentro del hogar: habitaciones y objetos.

Las habitaciones cuentan con la propiedad de ubicación, mientras los objetos tienen las propiedades de color, forma y ubicación. En la sección de habitaciones dentro del hogar, se definen cinco espacios:

– Recámara, Sala, Cocina, Comedor, Baño.

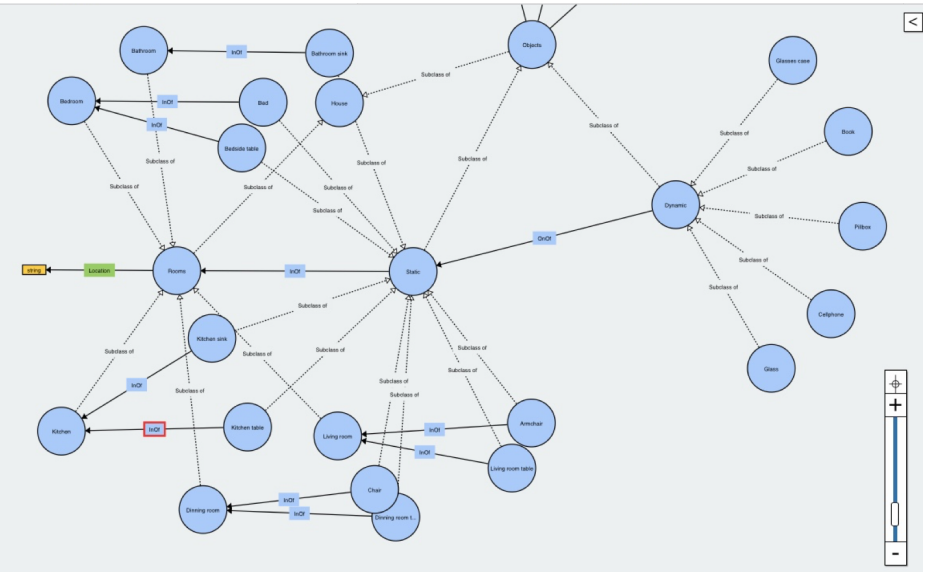


Fig. 5. Red con arcos de dependencia entre las clases.

Como mobiliario u objetos estáticos dentro del ambiente tipo hogar, en donde se pueden encontrar los objetos destino, se definen los siguientes:

– Cama, Mesa de comedor, Mesa de sala, Mesa de cocina, Silla, Sillón, Mesa de noche, Lavabo del baño, Lavabo de la cocina.

Por último, se definen los objetos objetivos, los cuales serán los que el robot debe localizar dentro del ambiente tipo hogar, como propuesta inicial se consideran cinco objetos, de forma y tamaño distinto:

– Celular, Vaso, Libro, Estuche de lentes, Pastillero.

Con los actores definidos, el siguiente paso es generar las relaciones correspondientes entre ellos. Las primeras a definir son las objeto-lugar, haciendo referencia a los objetos referencia dentro del ambiente tipo hogar, estas relaciones serán fijas, ya que dicho mobiliario no podrá cambiar de habitación. Las relaciones se muestran en la Figura 2.

Se encuentran en total nueve relaciones objeto-lugar dentro del ambiente doméstico. Las siguientes relaciones a generar son las objeto-objeto, aquí se generan las relaciones entre cada objeto referencia con cada objeto objetivo en cuestión. Las relaciones se muestran en la Figura 3.

Al final se crearon 45 relaciones objeto-objeto. Se puede identificar que cada objeto destino puede encontrarse sobre cada uno de los objetos estáticos, sin embargo, sabemos que no existe la misma probabilidad en cada uno de los casos. Por lo tanto, el siguiente paso es crear la tabla de porcentajes de probabilidad para cada relación objeto-objeto.

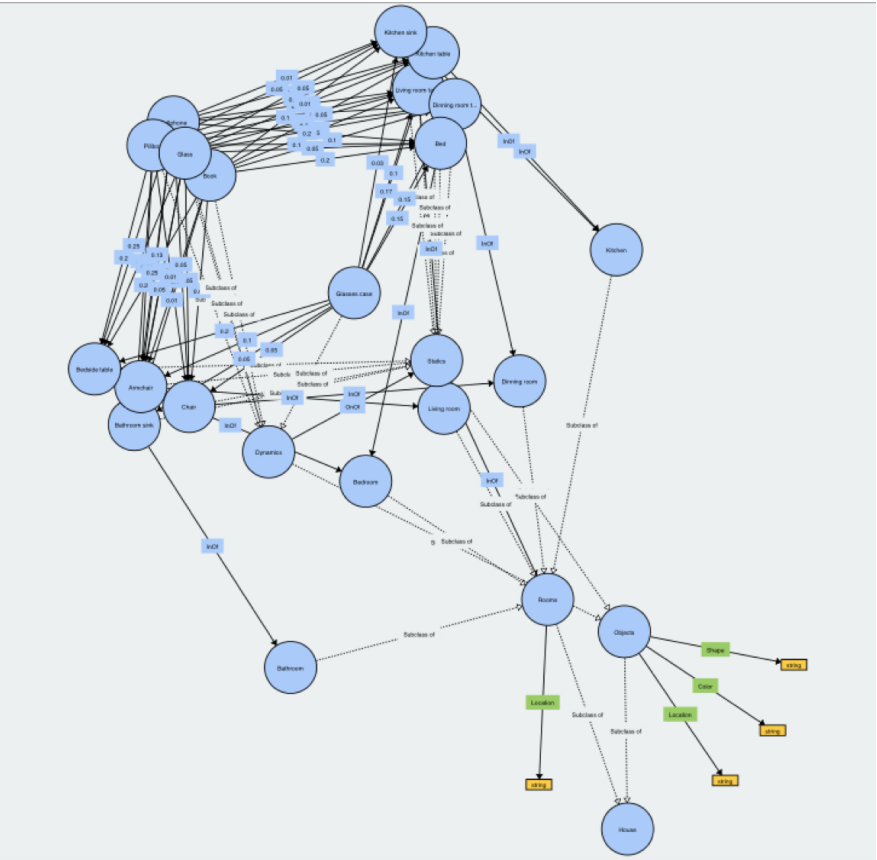


Fig. 6. Red completa con incorporación de probabilidades.

Los datos proporcionados son asignados previamente con el conocimiento adquirido del contexto en el que se encuentra el proyecto, por lo que son datos iniciales tentativos. Los de porcentajes de las relaciones objeto-objeto se muestran en la Tabla 1. Al sumar todas las probabilidades de los objetos referencia con cada objeto destino el resultado debe dar 1.

Como ya se conocen las relaciones objeto_referencia-lugar, se conoce la ubicación de cada uno de ellos dentro del ambiente, por lo que se pueden obtener los porcentajes de probabilidad de las relaciones objeto-lugar. Esto se logra con las habitaciones y los objetos objetivos, sumando las probabilidades de cada uno de los objetos referencia dentro de cada habitación específica. Los porcentajes de las relaciones objeto-lugar se muestran en la Tabla 2.

De igual manera, al sumar todas las probabilidades de las habitaciones en cada objeto destino, el resultado debe dar 1. Con esta información de las relaciones objeto-objeto, objeto-lugar y sus probabilidades correspondientes, ya se puede crear la red semántica adecuada.

(a) (b) (c)

Fig. 7. a) Robot diferencial. b) Simulación del ambiente tipo hogar. c) Mapa creado por el robot con las ubicaciones: A-lavabo de baño, B-mueble, C-mesa de sala, D-mesa de comedor, E-mesa de cocina, F-silla, G-lavabo de cocina, H-mesa de noche, I-cama.

4.2. Red semántica

La creación de la red semántica se realiza dentro del ambiente de programación WEBVOWL [10] correspondiente al lenguaje de OWL. Primeramente, se crea la red semántica únicamente con las relaciones objeto-objeto y objeto-lugar. La red se visualiza en la Figura 4.

Posteriormente, se generan los arcos correspondientes: Es-un para las subclases de objetos y habitaciones, dentro-de para los objetos estáticos dentro de las habitaciones y sobre-de para los objetos dinámicos sobre los objetos estáticos. La red se visualiza en la Figura 5.

El último paso es incorporar las probabilidades dentro de la red semántica, correspondientes a las diferentes relaciones existentes, el resultado de la red se visualiza en la Figura 6.

5. Implementación

La implementación de la red se realiza con un robot diferencial que cuenta con una plataforma Pioneer, un láser Sick LMS 200, una cámara RGB-D Asus Xtion, sonares, bumpers, altavoces, una platina móvil; la unidad de procesamiento es un NUC de Intel. El robot se muestra en la Figura 7(a).

El desarrollo del proyecto se llevó a cabo dentro de una simulación de un ambiente tipo hogar, que cuenta con cinco habitaciones (sala, cocina, comedor, recámara y baño), y diferentes objetos de mobiliario. El ambiente está mostrado en la Figura 7(b).

Con ayuda del láser del robot se realizó un mapeo del lugar para poder localizar las ubicaciones de los objetos estáticos dentro de las habitaciones y así pueda navegar el robot dentro del lugar. En el mapa se encuentran los distintos objetos estáticos repartidos entre las diferentes habitaciones. El mapa se puede visualizar en la Figura 7(c).

En el procedimiento, lo primero es seleccionar el objeto objetivo a buscar (vaso, estuche, celular, libro o pastillero).

Tabla 3. Resultados de las pruebas realizadas.

Objeto	N. Prueba	Ubicación	N. Lugar	Cambio en valores
Vaso	1	M. Com.	2	no
Vaso	2	M. Cocina	1	no
Vaso	3	L. Baño	9	si
Vaso	4	M. Com.	2	no
Celular	1	M. Noche	1	no
Celular	2	Mueble	4	si
Celular	3	M. Com.	5	si
Celular	4	M. Sala	2	no
E. lentes	1	M. Sala	2	no
E. lentes	2	M. Noche	1	no
E. lentes	3	M. Cocina	6	si
E. lentes	4	Mueble	5	si
Pastillero	1	M. Com.	1	no
Pastillero	2	M. Cocina	6	si
Pastillero	3	L. Baño	2	no
Pastillero	4	Cama	6	si
Libro	1	Mueble	5	si
Libro	2	M. Noche	1	no
Libro	3	M. Sala	2	no
Libro	4	Cama	3	no

Posteriormente, se le proporciona esta información a la red, mediante terminal, para que se realice la evaluación de la ontología, determinando el lugar más probable en donde se pueda encontrar dicho objeto.

A continuación, el robot recibe la ubicación determinada según la búsqueda realizada con la ontología, para que realice el movimiento al lugar. Si al llegar a la ubicación, el robot no encuentra el objeto en cuestión, se regresa a la red para modificar la ubicación posible con la siguiente de mayor probabilidad y así sucesivamente hasta encontrar el objeto.

Si el objeto fue encontrado en una ubicación con baja probabilidad al inicio, esto marcado a partir de la cuarta ubicación, se hace un ajuste de probabilidades en 0.2, aumentando en la ubicación donde se encontró y descontando a la ubicación más probable. Este proceso de actualización se realiza en cada búsqueda, con la finalidad de tener probabilidades según el contexto de las ejecuciones.

6. Resultados

Se realizaron 4 ejecuciones con cada uno de los objetos objetivos, para un total de 20 experimentos. En todos se partió de las probabilidades iniciales y posteriormente se fueron actualizando según si era el caso.

Para determinar la ubicación de cada objeto en cada prueba se consultó a una persona de la tercera edad, sobre en donde coloca dichos objetos en su rutina cotidiana. Todos los experimentos iniciaron con el robot en la misma ubicación.

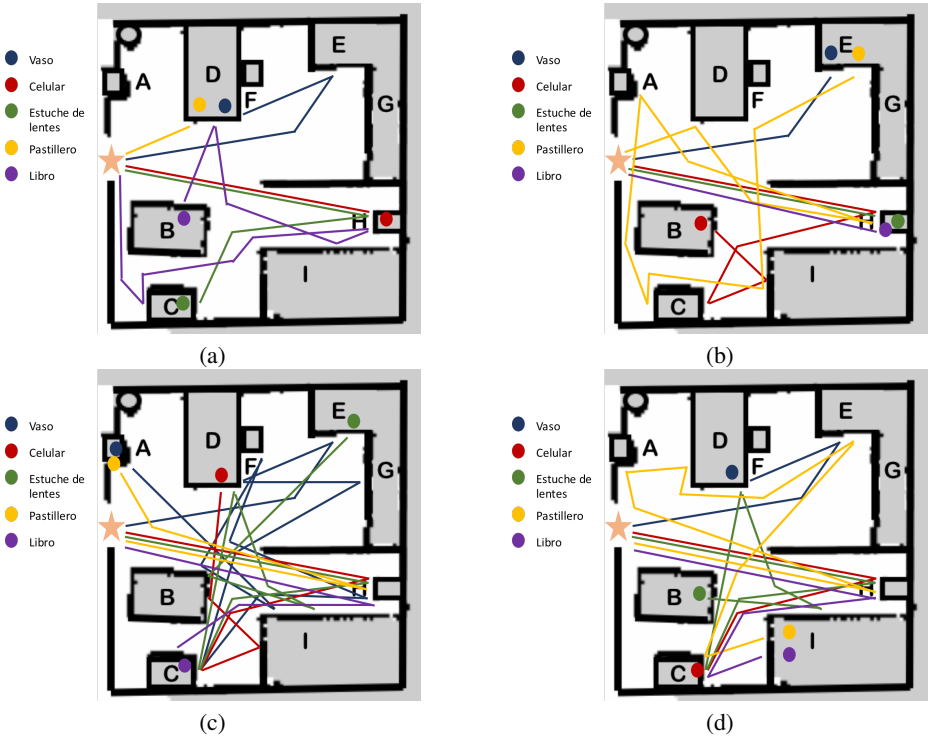


Fig. 8. Mapas con las rutas obtenidas en la búsqueda de objetos. El robot inicia en la estrella naranja. a) Prueba uno. b)Prueba dos. c)Prueba tres. d)Prueba cuatro.

Los resultados obtenidos se encuentran en la siguiente tabla, en donde se muestra el objeto buscado, el número de la prueba correspondiente, la ubicación donde fue encontrado el objeto, el número de la posición según la probabilidad de la red y si hubo cambio en los valores de las probabilidades (Tabla 3).

Detallado en la tabla de resultados, tanto el celular como el estuche y el pastillero sufrieron dos modificaciones en sus porcentajes, siendo el pastillero el del cambio más significativo debido a que se cambió la ruta de inicio de búsqueda.

Por otro lado, el vaso y el libro únicamente sufrieron un cambio de porcentaje, sin alterar las posiciones de los lugares para la búsqueda. Estos porcentajes se van modificando y ajustando cada vez más a la rutina de la persona conforme se incrementa el número de experimentos.

De igual forma, los resultados se pueden apreciar en los mapas de la Figura 8, en donde se muestran las rutas recorridas por el robot en cada prueba, correspondiente a cada objeto. El inicio siempre se marca en la estrella y el fin es donde se coloca el punto, dependiendo del color de cada objeto objetivo.

Se muestra en el mapa que el lugar más común, prueba 1, se adapta más a la red inicial propuesta que el tercer lugar predilecto, prueba 3, en donde el robot si tuvo que realizar una mayor serie de visitas a lugares para encontrar el objeto.

7. Conclusiones

Como se desprende de la sección anterior, en los experimentos realizados con la red semántica basada en una ontología sobre las relaciones objeto-objeto y objeto-lugar de un ambiente doméstico, se puede determinar que el robot realiza la búsqueda de manera eficiente con ayuda de la red; además de que, la red es adaptable a cualquier rutina de cualquier persona, esto gracias a las modificaciones de los porcentajes de probabilidad conforme se realizan los experimentos.

Según los resultados descritos en la sección anterior, se llega a la conclusión de que el uso de las redes semánticas con probabilidades genera rutas de búsqueda que reflejan caminos apropiados para la localización de los objetos, creando una búsqueda inteligente dentro del ambiente doméstico.

Como trabajo futuro se propone incorporar a la red una relación de los objetos por habitación, de tal manera que sí existe más de una ubicación con alta posibilidad por habitación se revisen todas ellas antes de partir de dicho cuarto. Esto con la finalidad de disminuir la cantidad de recorridos realizados por el robot, evitando tener que regresar a la habitación.

Referencias

1. Aracil, R., Balaguer, C., Armada, M.: Robots de servicio. Revista Iberoamericana de Automática e Informática Industrial RIAI, vol. 5, no. 2, pp. 6–13 (2008) doi: 10.1016/S1697-7912(08)70140-7
2. Aronson, J.: Encyclopedia of information systems. Academic Press (2003) https://www.sciencedirect.com/referencework/9780122272400/encyclopedia-of-information-systems
3. Arévalo-Vázquez, E., Zúñiga-López, A., Villegas-Cortez, J., Avilés-Cruz, C.: Implementación de reconocimiento de objetos por color y forma en un robot móvil. Research in Computing Science, vol. 91, pp. 21–31 (2015) doi: 10.13053/rcs-91-1-2
4. Ausubel, D., Novak, J., Hanesian, H.: Psicología educativa : Un punto de vista cognoscitivo. Trillas, pp. 623 (1983)
5. Aydemir, A., Sjöö, K., Folkesson, J., Pronobis, A., Jensfelt, P.: Search in the real world: Active visual object search based on spatial relations. In: 2011 IEEE International Conference on Robotics and Automation, pp. 2818–2824 (2011) doi: 10.1109/ICRA.2011.5980495
6. Borst, W. N.: Construction of engineering ontologies for knowledge sharing and reuse. Ph. D. thesis, Research UT, graduation UT, University of Twente, Centre for Telematics and Information Technology (CTIT) (1997)
7. Gruber, T. R.: A translation approach to portable ontology specifications. Knowledge Acquisition, vol. 5, no. 2, pp. 199–220 (1993) doi: 10.1006/knac.1993.1008
8. Kemp, C. C., Edsinger, A., Torres-Jara, E.: Challenges for robot manipulation in human environments [grand challenges of robotics]. IEEE Robotics and Automation Magazine, vol. 14, no. 1, pp. 20–29 (2007) doi: 10.1109/MRA.2007.339604
9. Kollar, T., Roy, N.: Utilizing object-object and object-scene context when planning to find things. In: 2009 IEEE International Conference on Robotics and Automation, pp. 2168–2173 (2009) doi: 10.1109/ROBOT.2009.5152831
10. Lohmann, S., Negru, S., Haag, F., Ertl, T.: Visualizing ontologies with VOWL. Semantic Web, vol. 7, no. 4, pp. 399–419 (2016) doi: 10.3233/SW-150200

11. Morales, E., Sucar, L.: Los robots del futuro y su importancia para México. Komputer Sapiens, vol. 1, pp. 1–11 (2009)
12. Shubina, K., Tsotsos, J.: Visual search for an object in a 3D environment using a mobile robot. Computer Vision and Image Understanding, vol. 114, no. 5, pp. 535–547 (2010) doi: 10.1016/j.cviu.2009.06.010
13. Studer, R., Benjamins, V. R., Dieter, F.: Knowledge engineering: Principles and methods. Data and Knowledge Engineering, vol. 25, no. 1, pp. 161–197 (1998) doi: 10.1016/S0169-023X(97)00056-6
14. Veloso, M. M., Biswas, J., Coltin, B., Rosenthal, S.: CoBots: Robust symbiotic autonomous mobile service robots. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015), pp. 4423–4429 (2015)
15. Wang, C., Cheng, J., Wang, J., Li, X., Meng, M.: Efficient object search with belief road map using mobile robot. IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 3081–3088 (2018) doi: 10.1109/LRA.2018.2849610
16. Zeng, Z., Röfer, A., Jenkins, O.: Semantic linking maps for active visual object search. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 1984–1990 (2020) doi: 10.1109/ICRA40945.2020.9196830
17. Zhang, Y., Tian, G., Shao, X., Zhang, M., Liu, S.: Semantic grounding for long-term autonomy of mobile robots toward dynamic object search in home environments. IEEE Transactions on Industrial Electronics, vol. 70, no. 2, pp. 1655–1665 (2023)

ISSN 1870-4069

Planificación inteligente para búsqueda de personas basada en sensores mediante un robot móvil autónomo

Francisco Rosas-Rebolledo, Antonio Marín-Hernández,
Sergio Hernández-Méndez, Roberto Cruz-Estrada

Universidad Veracruzana,
Instituto de Investigaciones en Inteligencia Artificial,
México

frosasreb@gmail.com,
{anmarin,sergihernandez,rcruz}@uv.mx

Resumen. En tareas de búsqueda y rescate el tiempo es vital para la atención de las víctimas. En un ambiente interior como un edificio público, la atención temprana para la búsqueda de personas puede ser compleja debido a la estructura de los espacios y a la poca información que se puede tener del lugar. En este trabajo se propone utilizar un robot para explorar un espacio interior conocido en busca de personas, basándose en la información actual de la ocupación de los espacios. Conocer la información sobre la ocupación de estos espacios ayudaría a generar una planificación adecuada que priorice buscar en los espacios con mayor cantidad de personas, procurando igualmente hacerlo minimizando la distancia de recorrido. La planificación se generará de acuerdo al lugar donde se encuentre el robot y mediante el algoritmo de la colonia de hormigas. Para la medición de ocupación de los espacios se ha considerado una red heterogénea de sensores, la cual se encuentra en una fase de desarrollo. Esta red permitiría estimar la cantidad de personas en un espacio de manera alternativa a las técnicas de conteo mediante visión por computadora, evitando así una invasión a la privacidad.

Palabras clave: Robots de rescate, internet de las cosas, planificación, búsqueda de personas.

Intelligent Planning for Sensor-Based Person Search Using an Autonomous Mobile Robot

Abstract. In search and rescue tasks, time is vital for the victims' attention. In an indoor environment, such as a public building, early attention for the search of people can be complex due to the structure of the spaces and the limited information that can be obtained from the place. In this paper, we propose using a robot to explore a known indoor space in search of people, based on current information of indoor occupancy. Knowing information about space occupancy would help generate adequate planning that prioritizes searching in spaces with a higher number of people, while also minimizing travel distance. The search plan will be generated according to the location of the robot and using the ant

colony optimization algorithm. For measuring indoor occupancy, a heterogeneous sensor network has been considered, which is currently under development. This network would estimate the number of people in a space in an alternative way to counting techniques using computer vision, thus avoiding an invasion of privacy.

Keywords: Rescue robots, internet of things, planning, person search.

1. Introducción

En el momento de que ocurre algún desastre o una situación de riesgo, la atención temprana es crucial; siendo la prioridad principal la de salvaguardar la integridad de las personas mediante acciones de búsqueda, rescate y atención hospitalaria.

Dentro del contexto de estas tareas de búsqueda y rescate de personas o atención a situaciones de riesgo, los robots son útiles tanto para la atención temprana de alguna eventualidad o emergencia como para las tareas posteriores al imprevisto.

De igual manera, bajo el concepto de las ciudades y edificios inteligentes se puede aprovechar la información que pueden generar un conjunto de dispositivos inteligentes para una mejor toma de decisiones logísticas en situaciones de búsqueda y rescate.

Entendiéndose a los dispositivos inteligentes como aquellas herramientas que están conectadas a internet y tienen la capacidad de recopilar, procesar, transmitir datos y, en algunos casos, tomar decisiones y acciones en función de esos datos.

Este trabajo está enfocado a la generación inteligente de planes de búsqueda de personas para un robot móvil autónomo, esto bajo un esquema similar al problema del agente viajero con ganancias. El plan de búsqueda definirá, de acuerdo a la ubicación del robot, los espacios donde convendría explorar en un ambiente definido y cerrado para encontrar la mayor cantidad de personas con la menor distancia recorrida posible. Esto se hace mediante la información de ocupación del lugar, obtenida mediante la propuesta del uso de módulos de sensores que permitirían contabilizar las personas en el espacio.

Los objetivos de trabajo se sintetizan en dos puntos: el primero va hacia la propuesta de un modelo para generar planificaciones inteligentes basadas en información proveniente de la red heterogénea de sensores para así mejorar el tiempo de respuesta de un robot móvil autónomo en la búsqueda de personas en espacios cerrados; y la segunda está en discutir el planteamiento para la implementación de técnicas no invasivas a la privacidad para contabilizar personas en espacios interiores. Este último punto implica el restringir el uso de técnicas de visión por computadora para dicha estimación.

Para validar la implementación y su utilidad, se ha decidido diseñar escenarios de simulación. Igualmente, se contempla el desarrollo con un robot físico, mostrado en la figura 1, cuyo espacio de trabajo será una de las plantas de nuestra institución. En ambos casos, en esta etapa preliminar del trabajo, se plantean en simulación el nivel de ocupación de los espacios, emulando así la información de los sensores.

El presente artículo consiste de diversas secciones que explican el desarrollo del trabajo. Primero se habla de antecedentes referentes al uso de la robótica en tareas de búsqueda de personas y en el uso de sensores distribuidos para el conteo de personas.

Fig. 1. Robot a utilizar en la implementación.

En las siguientes secciones se habla sobre las técnicas a utilizar para generar los planes de búsqueda y sobre el ambiente de pruebas a trabajar. Finalmente, se muestran algunos planes generados junto con ciertas discusiones sobre los resultados y mejoras a futuro.

2. Trabajos relacionados

Los robots en tareas de búsqueda y rescate comienzan a tener un nivel de relevancia después de su utilidad en desastres como en 9/11, el huracán Katrina y el terremoto del 2011 en Japón. El manejo de este tipo de robots puede ser remota a través de la teleoperación, permitiendo a los operadores humanos interactuar con entornos peligrosos de manera segura.

Sin embargo, la teleoperación puede ser limitada en situaciones donde la conectividad es limitada o nula, lo que hace que un cierto nivel de autonomía del robot sea importante para mejorar la efectividad y el tiempo de respuesta de las posibles tareas. Para dotar de autonomía a los robots se integran sensores, sistemas de planificación y control, así como técnicas de aprendizaje automático y visión por computadora [12, 11].

Igualmente, tecnologías como el Internet de las Cosas o IoT (por sus siglas en inglés, Internet of Things) pueden apoyar en estas tareas de rescate mediante el uso de redes de sensores distribuidos para conocer el contexto de los espacios. Un caso común es utilizar una red de sensores en los espacios interiores para detectar la presencia de personas o detectar peligros como fuego o gases.

Este tipo de trabajos utilizan la información de los sensores para la coordinación de los equipos de rescate, tanto haciendo un monitoreo del equipo como con actualizaciones sobre la situación del entorno.

Sabiendo del crecimiento del uso de dispositivos remotos conectados a internet, como en la tendencia de las ciudades inteligentes, la información que se puede obtener de los diferentes sensores permitirán conocer mejor el contexto del ambiente a trabajar [10, 3].

Para la búsqueda de personas utilizando robots generalmente se combinan técnicas de exploración junto con métodos para la detección de personas. En [6] se hace referencia a un robot que utiliza la técnica de exploración de frontera y técnicas de visión para identificar a las personas en un mapa que se crean mientras se explora.

Por otro lado, en el trabajo de [2] se genera un mapa con un robot en la que se localizan a las personas y posibles peligros, generando después una ruta para llegar a esas personas, evitando las zonas de riesgo.

La búsqueda de personas puede ser hecha igualmente basada en información previa. Un ejemplo de ello es el trabajo de [9], el cual inicia haciendo una petición de búsqueda de ciertos usuarios, y tomando la probabilidad de que estos usuarios se encuentren en diversas habitaciones, genera un plan de búsqueda basada en el Agente Viajero Selectivo u OP (por sus siglas en inglés, Orienteering Problem).

El trabajo de [5] propuso utilizar igualmente el OP para la búsqueda de personas, utilizando las probabilidades a priori de encontrar a los usuarios en ciertas zonas, con la diferencia de hacer la implementación en un espacio abierto.

Directamente en situaciones de búsqueda y rescate, se han desarrollado soluciones basadas en variantes del problema del agente viajero, como el problema de ruta de vehículos. Estos trabajos van buscando solucionar problemas como el definir los mejores lugares para albergues, distribución de bienes en ciertos centros y atención de emergencias [1].

Dentro del tema referente al conteo de personas en un espacio cerrado, comúnmente se toman técnicas de conteo basadas en visión por computadora o en la localización de dispositivos individuales debido a que son las que generan mayor precisión. El problema de utilizar estos métodos está relacionado con la posible invasión a la privacidad que se puede tener.

La utilización de elementos como cámaras en espacios cerrados; si bien puede justificarse en ciertos casos, puede generar algunos conflictos, que van desde cuestiones de incomodidad por parte de las personas, hasta generar problemas legales por la autorización de las personas de ser grabadas u observadas. Es por eso que existen estudios referentes a técnicas alternas para el conteo de personas [7].

Entre las técnicas que se encuentran podemos encontrar aplicaciones que contabilizan el paso de personas en un punto. Ya sea en la puerta de una habitación o un pasillo. Esta solución puede definir si una persona entra o sale utilizando sensores en dos puntos estratégicos, y dependiendo de que punto pase primero la persona se defina la entrada o salida [15].

Los sensores que generalmente son utilizados son los pasivos infrarrojos o PIR, sensores infrarrojos, ultrasónicos y sensores de tiempo de vuelo.

Como punto de atención, estos métodos tienen la dificultad de definir si la detección fue provista por un humano o algún otro objeto [7].

Otra manera de definir la cantidad de personas es mediante la medición de cambios en el ambiente y generación de inferencias basadas en esas mediciones. Las variables que pueden medirse pueden ser referentes a la humedad relativa, presión atmosférica, temperatura, medición de algunos gases como CO_2 , entre otros [14].

Otra manera es utilizar sensores PIR análogos que permita no solo medir la presencia de personas sino la posible cantidad de las mismas [8]. En ambos casos, se etiquetaron de antemano los datos y se utilizaron técnicas de Aprendizaje Máquina para generar la inferencia. Estas técnicas pueden estar limitadas a cambios muy repentinos de aforo o a errores de medición en aforos muy grandes, además de ser sensibles a posibles ruidos de las condiciones del ambiente [7].

3. Propuesta conceptual

Para la implementación se deben definir dos elementos relevantes: el plan de búsqueda y los módulos para el conteo de personas. El plan de búsqueda debe dar prioridad de búsqueda a lugares con mayor cantidad de personas, tomando en cuenta la distancia con respecto a la ubicación del robot.

Los módulos contadores de personas deben generar un estimado de la presencia y cantidad de personas en los espacios, esto tomando en cuenta sus distintos diseños y tamaños de los espacios.

3.1. Problema del agente viajero

El problema del agente viajero es, en síntesis, es un problema de optimización combinatoria en el que se busca encontrar el camino más corto que visita un conjunto de ciudades dadas y regresa a la ciudad de origen.

Comúnmente se agregan ciertas restricciones o se agregan ganancias a las ciudades para adaptar el problema a diversas situaciones reales. Para su aplicación generalmente se diseña un grafo, cuyos nodos representan las ciudades a visitar y las aristas representan un camino entre un nodo y otro.

En esta implementación cada nodo representa un espacio de interés y tiene asociada una ganancia, la cual hace referencia a la probable cantidad de personas que hay en ese espacio. Las aristas tienen asociado un costo que representa la distancia que hay entre esos espacios. Esta distancia está dada por el recorrido que un robot podría hacer para llegar de un espacio al otro.

Este problema es considerado NP-duro, por ende no existe una solución algorítmica eficiente para resolverlo en todos los casos, por lo que se utilizan diversas técnicas para encontrar soluciones aproximadas.

Mientras que los algoritmos con enfoque heurístico busca generar una solución aceptable en menor tiempo. En la mayoría de los casos se asume que las ganancias de los nodos son fijas, aunque se han hecho alternativas con pesos dinámicos [13].

El algoritmo de optimización por colonia de hormigas o ACO (por sus siglas en inglés, Ant Colony Optimization) es escogido para la generación del plan de búsqueda

debido a su facilidad de implementación, facilidad de modificación para agregar restricciones, facilidad para lidiar con diversos escenarios, y su costo bajo de recursos computacionales con respecto a otras soluciones. Debido a su enfoque heurístico, con ACO se puede tener menor precisión para encontrar una ruta óptima, pero permite obtener soluciones funcionales.

ACO utiliza una colonia de hormigas artificiales que se mueven dentro del grafo. Cada hormiga sigue una heurística y deja feromonas en su camino, lo que permite a otras hormigas decidir su siguiente movimiento.

En el problema del agente viajero con ganancias se busca minimizar la distancia a recorrer maximizando la ganancia. Por lo tanto, cada hormiga realiza un recorrido de acuerdo con una función de probabilidad que considera tanto la distancia como la ganancia. La feromona depositada por cada hormiga se actualiza en función del beneficio obtenido en cada ciudad visitada.

Al final de cada iteración, las feromonas son actualizadas para reflejar el mejor camino encontrado hasta ese momento. El proceso se repite hasta que se alcanza el límite de iteraciones [4]. Los parámetros principales para modular el comportamiento del algoritmo son:

- **Número de hormigas:** La cantidad de hormigas a utilizar. Cada hormiga construirá una posible solución.
- α: El nivel de importancia del rastro de la feromona cuando la hormiga escoja el siguiente nodo a visitar.
- β: El nivel de importancia de la heurística (en este caso, la ganancia de cada nodo) cuando la hormiga escoja el siguiente nodo a visitar.
- ρ: La velocidad a la que se evapora la feromona entre iteraciones.
- **Número de iteraciones:** El número de iteraciones en las que se ejecutará el algoritmo.

3.2. Conteo de personas

Los módulos de conteo de personas para espacios cerrados son de utilidad en muchas situaciones que van desde apoyo en el uso eficiente de energía de edificios (como con el control de los sistemas de aire acondicionado), estudios de mercado en plazas comerciales, salud (COVID-19), seguridad y logística en situaciones de emergencia.

Ciertas líneas de trabajo procuran utilizar sensores que logren una buena precisión en la estimación y permitan el respeto a la privacidad. Estas estimaciones pueden ser refinadas mediante técnicas de aprendizaje automático y fusión de sensores.

Una primera propuesta de módulo consiste en utilizar sensores PIR para detectar la presencia de personas y se utiliza un algoritmo de conteo distribuido para estimar la cantidad total de personas en el área [15]. Este trabajo presentó tanto los resultados de simulación como datos en ambientes reales, pudiendo proporcionar una estimación con una precisión del 97 % en promedio.

La segunda propuesta presenta un enfoque para estimar los niveles de ocupación en espacios cerrados mediante variables ambientales [14].

Trabajando en escenarios reales como un gimnasio y una sala de estar, se midieron variables como la temperatura, la humedad y el CO2.

A partir de estos datos, desarrollan modelos de aprendizaje automático para generar una inferencia. Se utilizaron dos algoritmos de aprendizaje automático: Regresión Lineal Múltiple y Máquinas de Soporte Vectorial para estimar los niveles de ocupación.

La tercera propuesta es utilizar un PIR análogo para un espacio cerrado. A diferencia de un PIR digital en la que la respuesta del sensor es si hay presencia o no, el PIR análogo puede tener variaciones de acuerdo a la cantidad de personas dentro de su rango de detección.

Para implementarlo se utiliza un modelo de Markov oculto infinito para segmentar y filtrar los datos del sensor PIR, y luego se utiliza un modelo de regresión para predecir la ocupación. El procesamiento y predicción se hace internamente en un microcontrolador [8].

4. Implementación

Para los experimentos se contemplan diversos escenarios simulados y una implementación real dentro del edificio de la universidad. Utilizando el caso de búsqueda de personas en situaciones de riesgo, este proyecto asume algunos puntos importantes sobre el ambiente a trabajar:

- Se asume que el suelo en el que el robot navegará es plano y firme.
- Existe un mapa previo definido del ambiente a trabajar.
- Cada habitación está representada como un nodo de un grafo.

Para alistar la utilización de los mapas de simulación y el ambiente real, se deben definir cuántos y cómo los nodos van a representar los espacios de interés. En la mayoría de los casos una habitación representará un nodo, aunque habrán casos, debido a su complejidad o que la representación de una parte del espacio pueda promover una posible invasión a la privacidad, varios espacios estarán conjuntados con un nodo.

4.1. Plan de búsqueda

El plan de búsqueda será hecho basado en el algoritmo Ant Colony Optimization. El código es hecho en Python para la toma de decisiones. Se tienen definidas las distancias entre los nodos. Se tiene de igual manera una lista con las ganancias asociadas a cada nodo, siendo estas la representación de la cantidad de personas que se estiman dentro de la habitación.

El algoritmo tendrá como entrada el nodo de donde debería comenzar el plan de búsqueda, la matriz de las distancias entre nodos, la lista con la cantidad de personas en cada nodo y los parámetros de ACO. La salida representa una lista en la que se ordena el orden de visita de las habitaciones sugerida para el robot.

Se tiene como entrada del algoritmo el nodo inicial debido a que se busca simular diversos escenarios en donde se deba planificar una búsqueda basada en la ubicación actual del robot, dándole así un cierto dinamismo.

4.2. Robot móvil autónomo

Se utiliza un robot con una plataforma móvil diferencial, el cual tendrá como objetivo llegar a los nodos definidos por el plan de búsqueda. El robot tiene bumpers, sensores ultrasónicos, laser, cámara RGB y una unidad de procesamiento Intel NUC.

Aunque el contexto sea sobre búsqueda en situaciones de riesgo, no se contempló hacer adaptaciones mecánicas al robot para que navegue en terrenos sinuosos o rugosos, así como tampoco algún sistema de control que apoye para este tipo de navegación. Igualmente, la interacción con las personas estará limitada a evitar colisiones con las personas.

Se utiliza Gazebo, el cual es un simulador de robótica 3D de código abierto que permite simular el comportamiento de robots y ambientes en tiempo real. Trabaja con ROS (por sus siglas en inglés, Robot Operating System) y se utiliza para probar implementaciones con robots en un entorno seguro y controlado.

Se plantean en Gazebo tanto los escenarios de simulación como una evaluación del comportamiento en el entorno real definido. Se utiliza ROS para trabajar lo referente a las paqueterías de navegación autónoma del robot. Se enviarán una serie de objetivos de navegación del robot de acuerdo al plan de búsqueda.

Todos los posibles objetivos están referenciados como un punto definido en el mapa, al cual se le enviará al robot como una meta en el orden que el plan de búsqueda lo tenga.

4.3. Construcción de los módulos

Los módulos que se encuentran en desarrollo para el conteo de personas se compondrán de estos elementos básicos:

- Módulo WiFi ESP32.
- Bateria de litio con cargador USB para la batería.
- Regulador de voltaje.
- Sensores.

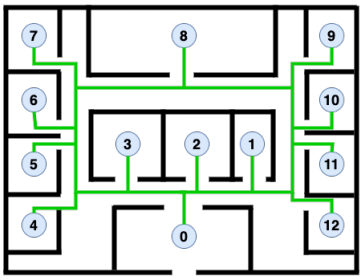
Para el módulo que trabaja con los sensores PIR digitales, se necesitan un par de sensores HC-SR501. El segundo módulo trabaja con el sensor BME280. El PIR análogo a utilizar en la tercera propuesta es el AMN21112.

4.4. Entorno de pruebas

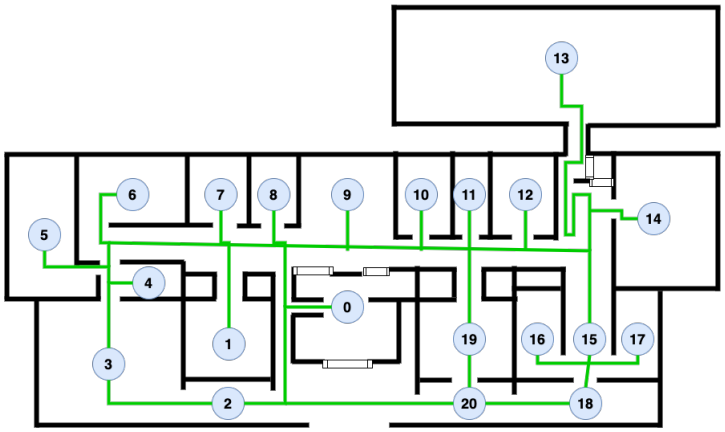
Se contempla una serie de simulaciones mediante 3 mapas con diseños distintos entre si para validar la utilidad que puede dar el plan de búsqueda. Se definen para cada mapa las zonas de interés y la distancia que puede haber entre los puntos, para así construir un grafo representativo de cada ambiente.

Se generaron representaciones de todos los escenarios a trabajar en Gazebo para la simulación. Para cada mapa se define el camino por el cual un robot pueda navegar para llegar a los nodos.

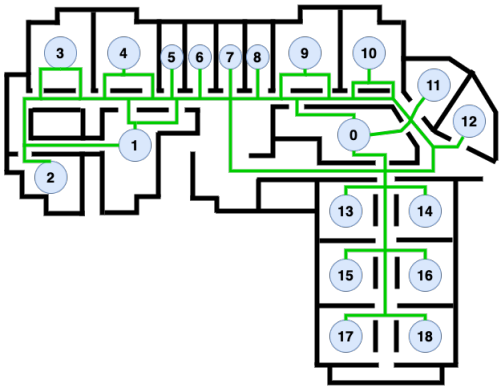
El desarrollo para el ambiente real es hecho en uno de los pisos de nuestro instituto. Para esto se definieron las zonas en las que el robot podría moverse, omitiendo así lugares como escaleras, por ejemplo.



(a)



(b)



(c)

Fig. 2. Mapas de entorno de simulación.

Fig. 3. Representación del ambiente real.

En los mapas se puede notar esta referencia, tanto en la figura 2 para los ambientes de simulación, como en 3 para el entorno real. Los mapas tienen dibujadas líneas que conectan a los nodos, estas líneas fueron usadas como referencia para estimar la posible distancia entre los nodos.

En la definición de los espacios de interés se contemplaron los salones individualmente; mientras que espacios más grandes como el centro de cómputo o la zona de cubículos se toman como uno solo, para así reducir la complejidad del análisis. En el caso de la zona de cubículos, teniendo énfasis que uno de los objetivos del proyecto es el respeto a la privacidad, agrupar toda la zona como un solo punto de interés ayuda a preservar la privacidad de los profesores en sus espacios de trabajo.

5. Resultados

En el ambiente de simulación, la estimación de cantidad de personas en los espacios de interés serán números arbitrarios que nos permitan visualizar la utilidad de la implementación. Mientras que en el ambiente real se busca poner a prueba el trabajo de la toma de decisiones con información real de por medio, simulando que los sensores están en funcionamiento.

Para la experimentación se tendrá como base iniciar el recorrido en el nodo 0, y se decidirá generar otro plan de búsqueda definiendo un nodo inicial distinto, dando a entender que el robot se encuentra en ese momento cerca de ese nodo en el momento de la planificación. Con esto se busca mostrar los diferentes cambios de prioridades del plan de búsqueda de acuerdo a su ubicación.

Para cada mapa se muestra una tabla, la cual se compone de dos secciones: lista de ganancias y ruta de nodos. En la sección lista de ganancias se muestra una lista que hace alusión a las ganancias de cada nodo, simulando que esta sea la cantidad de personas que hay en la habitación.

Esta lista está ordenada representando la ocupación desde el nodo 0. Por otro lado, en la sección “Ruta de nodos”se muestra una lista que da el orden de visita de los nodos sugerido por el algoritmo. La numeración a la que se hace referencia en la ruta de nodos está dada por la representación del espacio a visitar.

Tabla 1. Simulación en mapa 2a.

Lista de ganancias	Ruta de nodos
[11 3 14 11 6 6 5 4 8 5 7 8 9]	0, 2, 3, 12, 11, 10, 9, 8, 5, 4, 6, 7, 1
[11 3 14 11 6 6 5 4 8 5 7 8 9]	10, 2, 0, 3, 12, 11, 8, 5, 4, 6, 7, 9, 1
[3 9 5 4 3 14 6 5 8 12 2 5 4]	0, 5, 7, 9, 1, 8, 6, 3, 2, 11, 12, 4, 10
[3 9 5 4 3 14 6 5 8 12 2 5 4]	6, 5, 9, 11, 1, 12, 8, 7, 2, 3, 0, 10, 4
[3 9 5 4 3 14 6 5 8 12 2 5 4]	1, 9, 8, 5, 6, 7, 2, 11, 12, 3, 0, 4, 10

Tabla 2. Simulación en mapa 2b.

Lista de ganancias	Ruta de nodos
[1 9 0 9 2 8 9 8 4 5 1 4 4 2 5 4 14 2 6 4 2]	0, 16, 18, 14, 5, 3, 8, 1, 7, 6, 9, 11, 19, 12, 15, 17, 20, 4, 13, 10, 2
[1 9 0 9 2 8 9 8 4 5 1 4 4 2 5 4 14 2 6 4 2]	11, 16, 14, 1, 7, 6, 3, 9, 5, 4, 18, 15, 12, 19, 10, 8, 20, 0, 17, 13, 2
[1 14 6 11 13 9 11 10 7 10 13 12 13 14 2 14 9 11 12 11 9]	0, 1, 4, 3, 6, 5, 10, 11, 20, 15, 12, 9, 19, 18, 17, 13, 16, 7, 8, 2, 14
[1 14 6 11 13 9 11 10 7 10 13 12 13 14 2 14 9 11 12 11 9]	16, 15, 17, 19, 11, 10, 20, 18, 12, 13, 1, 5, 4, 3, 6, 7, 9, 8, 2, 14, 0
[1 14 6 11 13 9 11 10 7 10 13 12 13 14 2 14 9 11 12 11 9]	2, 3, 4, 1, 7, 10, 11, 12, 15, 18, 17, 13, 19, 20, 16, 9, 5, 8, 6, 14, 0

Para las simulaciones se tienen las siguientes configuraciones:

- Número de hormigas: Igual a la cantidad de nodos de cada mapa.
- Número total de iteraciones: 1000
- α: 2
- β: 3
- ρ: 0.7

Estos parámetros de ACO han dado, de acuerdo a diversas pruebas, un balance entre dar prioridad a los lugares con más cantidad de personas y buscar rutas más cortas para explorar todos los espacios. Los parámetros tanto en los entornos de simulación como en el escenario real se mantienen iguales.

En la figura 4 se pueden apreciar algunos ejemplos de las simulaciones hechas en Gazebo, en la que se tiene el modelo del robot recorriendo los objetivos. La figura 4a tiene como mapa una representación del ambiente real, mientras que la figura 4b es una representación del mapa 2b.

5.1. Simulaciones

La tabla 1 da referencia al mapa de la figura 2a, la tabla 2 al mapa de la figura 2b y la tabla 3 al mapa de la figura 2c. Como se puede notar, la ruta propuesta da prioridad a ir a los nodos con mayor ganancia, aun si el recorrido pueda no ser óptimo.

Los parámetros definidos para el ACO tienen un balance prioritario para buscar los nodos con mayor ganancia, pero sin dejar atrás el factor de la distancia.

Tabla 3. Simulación en mapa 2c.

Lista de ganancias	Ruta de nodos
[11 13 10 7 11 14 13 0 0 12 3 6 14 9 8 8 11 11 1]	0, 9, 5, 6, 1, 4, 12, 14, 16, 17, 13, 15, 2, 3, 11, 10, 18, 7, 8
[11 13 10 7 11 14 13 0 0 12 3 6 14 9 8 8 11 11 1]	7, 1, 5, 6, 16, 14, 12, 17, 0, 9, 4, 2, 3, 13, 15, 11, 10, 18, 8
[4 1 0 4 11 9 13 6 13 13 4 10 8 5 14 10 6 5 1]	0, 14, 8, 9, 6, 4, 7, 5, 11, 12, 15, 16, 13, 17, 10, 3, 1, 18, 2
[4 1 0 4 11 9 13 6 13 13 4 10 8 5 14 10 6 5 1]	5, 6, 8, 9, 14, 15, 11, 12, 4, 7, 16, 13, 0, 17, 10, 3, 1, 18, 2
[4 1 0 4 11 9 13 6 13 13 4 10 8 5 14 10 6 5 1]	13, 14, 9, 8, 6, 4, 5, 11, 12, 15, 16, 17, 7, 0, 10, 3, 1, 18, 2

Tabla 4. Resultados del ambiente real.

Lista de ganancias	Ruta de nodos
[0 7 10 1 1 3 6 1 10]	0, 1, 2, 8, 6, 5, 4, 3, 7
[10 8 1 11 12 6 13 2 13]	0, 4, 3, 6, 8, 1, 5, 7, 2
[10 8 1 11 12 6 13 2 13]	5, 4, 3, 6, 8, 0, 1, 7, 2
[5 13 9 5 11 4 1 14 5]	0, 1, 7, 4, 3, 2, 8, 6, 5
[5 13 9 5 11 4 1 14 5]	2, 7, 1, 4, 3, 5, 0, 8, 6
[5 13 9 5 11 4 1 14 5]	3, 4, 0, 1, 7, 2, 6, 8, 5

Es por eso que, generalmente, si el nodo con la mayor ganancia está relativamente cerca, el algoritmo buscará ir a ese nodo desde el principio.

Sin embargo, como se puede observar en la tabla 3 con el caso 4, después del nodo inicial 5 el algoritmo procuro buscar en nodos más cercanos con una buena cantidad de ganancias en lugar de buscar en el nodo 14, el cual es el que tiene la mayor ganancia. Este mismo fenómeno se presentan en distintos casos mostrados.

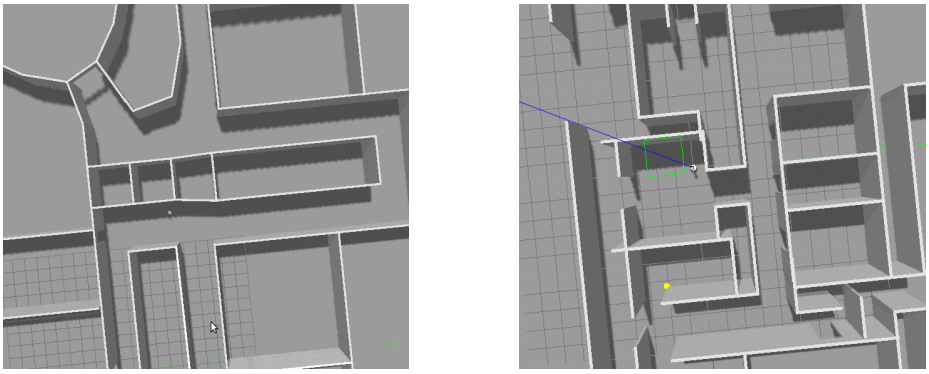
Aun con la observación previa, en todos los casos existe un cierto nivel de coincidencia entre los primeros nodos que el algoritmo toma como prioritarios y los nodos que se dejan hasta el final del plan. Es más notoria esta coincidencia con los nodos que están al fondo de la lista.

5.2. Ambiente real

Para tener el entorno listo se genera una lista con la descripción de la cantidad de personas que hay en cada nodo, haciendo la simulación del funcionamiento de la red de sensores.

Con la lista hecha se genera el plan de búsqueda y se estará mandando los puntos objetivo al robot para que visite los espacios, simulando así una búsqueda. La tabla 4 muestra los planes de búsqueda referentes al ambiente real, representado en la figura 3.

Tanto en los entornos de simulación como en el ambiente real se observan fenómenos similares en los planes de búsqueda. Independientemente del diseño



(a) (b)

Fig. 4. Simulaciones en Gazebo.

distintivo de cada espacio, se logra ver que el plan de búsqueda prioriza los espacios con mayor cantidad de personas.

6. Conclusiones y trabajos futuros

La implementación del algoritmo ACO para resolver el problema del agente viajero con ganancias resulta útil para generar una planificación de rutas inteligente, en la que se priorice buscar en los espacios donde haya la mayor cantidad de personas procurando una distancia de recorrido baja.

Tener una red heterogénea de sensores mediante técnicas alternativas a la visión por computadora podría permitir que la implementación se adapte a los diferentes tipos de espacios y lidiar con sus complejidades, además de evitar invadir la privacidad de las personas. Este desarrollo puede ser ideal para el manejo de situaciones de atención temprana de una crisis, o para complementar la tarea de rescatistas mediante la información del entorno.

Como trabajo futuro se plantean diversas mejoras. Una de ellas es referente a evaluar nuevos métodos de conteo de personas y hacer una combinación de las mismas para mejorar la precisión.

Otro punto de mejora es referente a una replanificación reactiva de acuerdo a las mediciones de los sensores y las condiciones que se tengan, para así actuar con mayor inmediatez a escenarios reales. Igualmente, será relevante tomar en consideración diferentes características de los espacios para ayudar a determinar su prioridad en la búsqueda. Con respecto al robot, se propone agregar técnicas de exploración e interacción con personas para su atención y auxilio en las situaciones de crisis.

Referencias

1. Baxter, A. E., Wilborn Lagerman, H. E., Keskinocak, P.: Quantitative modeling in disaster management: A literature review. IBM Journal of Research and Development, vol. 64, no. 1/2, pp. 3:1–3:13 (2020) doi: 10.1147/JRD.2019.2960356

2. Bock, A., Kleiner, A., Lundberg, J., Ropinski, T.: An interactive visualization system for urban search and rescue mission planning. In: IEEE International Symposium on Safety, Security, and Rescue Robotics, pp. 1–7 (2014) doi: 10.1109/SSRR.2014.7017652
3. Dimou, A., Kogias, D. G., Trakadas, P., Perossini, F., Weller, M., Balet, O., Patrikakis, C. Z., Zahariadis, T., Daras, P.: FASTER: First responder advanced technologies for safe and efficient emergency response. Technology Development for Security Practitioners, pp. 447–460 (2021) doi: 10.1007/978-3-030-69460-9_26
4. Dorigo, M., Birattari, M., Stutzle, T.: Ant colony optimization. IEEE Computational Intelligence Magazine, vol. 1, no. 4, pp. 28–39 (2006) doi: 10.1109/MCI.2006.329691
5. Guitouni, A., Masri, H.: An orienteering model for the search and rescue problem. Computational Management Science, vol. 11, no. 4, pp. 459–473 (2014) doi: 10.1007/s10287-013-0179-1
6. Kohlbrecher, S., Meyer, J., Graber, T., Petersen, K., Klingauf, U., von Stryk, O.: Hector open source modules for autonomous mapping and navigation with rescue robots. In: RoboCup 2013: Robot World Cup XVII. vol. 8371, pp. 624–631 (2014) doi: 10.1007/978-3-662-44468-9_58
7. Kouyoumdjieva, S. T., Danielis, P., Karlsson, G.: Survey of non-image-based approaches for counting people. IEEE Communications Surveys & Tutorials, vol. 22, no. 2, pp. 1305–1336 (2020) doi: 10.1109/COMST.2019.2902824
8. Leech, C., Raykov, Y. P., Ozer, E., Merrett, G. V.: Real-time room occupancy estimation with Bayesian machine learning using a single PIR sensor and microcontroller. In: 2017 IEEE Sensors Applications Symposium (SAS), pp. 1–6 (2017) doi: 10.1109/SAS.2017.7894091
9. Mohamed, S. C., Rajaratnam, S., Hong, S. T., Nejat, G.: Person finding: An autonomous robot search method for finding multiple dynamic users in human-centered environments. IEEE Transactions on Automation Science and Engineering, vol. 17, no. 1, pp. 433–449 (2020) doi: 10.1109/TASE.2019.2928774
10. O'Flynn, B., Brahmi, I., Oudenhoven, J., Nackaerts, A., Pereira, E., Agrawal, P., Fuchs, T., Braun, T., Lang, K. D., Dils, C., Walsh, M.: First responders occupancy, activity and vital signs monitoring - SAFESENS. International Journal on Advances in Networks and Services, vol. 11, no. 1-2, pp. 22–32 (2018)
11. Russell, S., Norvig, P.: Artificial intelligence, a modern approach. Pearson Education (2021)
12. Siciliano, B., Khatib, O.: Springer handbook of robotics. Springer (2008)
13. Vansteenwegen, P., Gunawan, A.: Orienteering problems: Models and algorithms for vehicle routing problems with profits. Springer (2019)
14. Vela, A., Alvarado-Uribe, J., Davila, M., Hernandez-Gress, N., Ceballos, H. G.: Estimating occupancy levels in enclosed spaces using environmental variables: A fitness gym and living room as evaluation scenarios. Sensors, vol. 20 (2020) doi: 10.3390/s20226579
15. Wahl, F., Milenkovic, M., Amft, O.: A distributed PIR-based approach for estimating people count in office environments. In: 2012 IEEE 15th International Conference on Computational Science and Engineering, pp. 640–647 (2012) doi: 10.1109/ICCSE.2012.92

ISSN 1870-4069

Reconstrucción de los parámetros de la calidad del agua en el río Fuerte, Sinaloa, implementando técnicas de aprendizaje máquina

José Luis Medina-Jiménez, Héctor Rodríguez-Rangel,
Leonel Ernesto Amábilis-Sosa, Kimberly Mendivil-García,
Juan Carlos Gonzalez-Nava, Daniel Antonio Nieblas-Fuentes

Tecnólogico Nacional de México Campus Culiacán,
División de Posgrado,
México

{jose.mj, hector.rr, leonel.as}@culiacan.tecnm.mx,
{kimberly.mendivil, 19171353, 19170633}@itculiacan.edu.mx

Resumen. La comisión nacional del agua (CONAGUA), a través de los años, mediante una red de monitoreo (RENAMECA), ha generado conjuntos de datos de parámetros que miden la calidad del agua, donde el objetivo es realizar tomas de muestras en distintos puntos estratégicos de los cuerpos de agua. Por diversos factores, como falla de captura, la dificultad de entrar a la zona de muestreo, falta de personal capacitado, tiempos de análisis de laboratorios, el registro de estos parámetros de calidad de agua no se realizan. De tal manera que, el conjunto de datos generado por la RENAMECA cuenta con una de las problemáticas más comunes dentro de la ciencia de datos, los valores perdidos dentro de cada uno de los parámetros en distintos niveles de porcentaje de perdida. Históricamente, el avance de la estadística y de la inteligencia artificial, han generado distintas técnicas capaces de recuperar estos valores faltantes, llamándolas métodos de imputación. En el artículo se describe la implementación de tres distintos métodos de imputación en un conjunto de datos de CONAGUA del río Fuerte en Sinaloa de 38 parámetros, donde la reconstrucción se realiza en siete parámetros. La validación de la reconstrucción se hizo mediante la simulación de distintos porcentajes de valores perdidos de 10, 20 y 30 %, obteniendo distintos resultados en valores de MSE y RMSE.

Palabras clave: Imputación, reconstrucción de datos, vecinos más cercanos K, regresión lineal.

Reconstruction of Water Quality Parameters in the Fuerte River, Sinaloa, Using Machine Learning Techniques

Abstract. The National Water Commission (CONAGUA), over the years, has generated datasets of parameters that measure water quality through a monitoring network (RENAMECA), where the objective is to take samples at strategic points in bodies of water. Due to various factors, such as capture failure,

difficulty accessing the sampling area, lack of trained personnel, and laboratory analysis times, the recording of these water quality parameters is not always possible. As a result, the dataset generated by RENAMECA has one of the most common problems in data science, missing values within each parameter at different levels of percentage loss. Historically, the advancement of statistics and artificial intelligence has generated different techniques capable of recovering these missing values, calling them imputation methods. This article describes the implementation of three different imputation methods on a CONAGUA dataset from the Fuerte River in Sinaloa with 38 parameters, where reconstruction is carried out on seven parameters. The validation of the reconstruction was done by simulating different percentages of missing values of 10, 20, and 30%, obtaining different results in MSE and RMSE values.

Keywords: Imputation, data reconstruction, K nearest neighbors, linear regression.

1. Introducción

En el campo de la ciencia y análisis de datos, la calidad de los conjuntos de datos es un tema central de gran importancia. Actualmente, diversos campos como las ciencias de la salud, finanzas, medio ambiente, etc., generan conjuntos de datos para mantener registro de los comportamientos de objetos de estudio en sus áreas y así realizar análisis estadísticos, inferencias y modelados [1, 2].

Los conjuntos de datos son colecciones de información almacenadas y organizadas en filas y columnas. Los datos pueden ser de cualquier tipo, como números, texto, imágenes, audio, etc. Los conjunto de datos son generados de diversas maneras, como la recopilación manual de datos mediante encuestas, entrevistas, observaciones, cuestionarios.

Por otro lado, se puede implementar la extracción de datos en las bases de datos mediante consultas,y por último existen los sensores, que son capaces de medir y recopilar información en tiempo real sobre el entorno, como temperatura, presión, humedad, entre otros. En el área de la inteligencia artificial la utilización de conjuntos de datos es esencial, ya que gracias a estos son capaces de generar modelos que puedan estimar o clasificar los comportamientos que se desea analizar.

Es así que, la cantidad y calidad de datos con las que se cuenta en un conjunto de datos son cruciales por el hecho de que estos determinan la precisión y la capacidad de generalizar modelos de aprendizaje [3]. Una de las ventajas de contar con conjuntos de datos diverso, es la ayuda para generar modelos eficaces donde se evitan problemáticas comunes como el sobreajuste y subajuste.

En México existe un organismo denominado CONAGUA el cual se encarga, de medir la calidad en los diversos cuerpos de agua con los que cuenta. En él se implementa una red nacional de monitoreo (RENAMECA), en el cual se colocaron puntos de muestreo estratégicamente para la medición de distintos parámetros de la calidad del agua [4].

Tabla 1. Parámetros de la calidad del agua y sus porcentajes de valores perdidos.

Parámetro	Porcentaje de datos perdidos
N_NO3	0.3205
N_TOT	0.3205
TOX_V_15_UT	0.3205
OD_mg/L	0.3205
TURBIEDAD	0.3205
AS_TOT	0.3205
CD_TOT	0.3205
CR_TOT	0.3205
COT	0.6410
PB_TOT	0.6410
TEMP_AMB	0.6410
TEMP_AGUA	0.6410
HG_TOT	0.9615
NI_TOT	0.9615
SDT	1.9231
CONDUC_CAMPO	1.9231
ABS_UV	2.5641
COT_SOL	3.2051
DUR_TOT	3.2051
E_COLI	3.5256
DBO_TOT	3.8462
DQO_TOT	4.1667
TOX_D_48_UT	4.1667
COLI_TOT	5.7692
DBO_SOL	6.4103
DQO_SOL	6.7308
CN_TOT	16.0256
CAUDAL	22.4359

El conjunto de datos generado por la RENAMECA de los parámetros de la calidad del agua en todo México, por diversos factores como error de captura de datos, traslado al punto de muestreo, tiempo de obtención de resultados, presenta una de las problemáticas más usuales dentro de la ciencia de datos, el cual es la falta de información o la perdida de información en diversos parámetros y en distintos niveles de porcentaje de faltantes [1].

De tal manera que la implementación de modelos estadísticos y de aprendizaje máquina en conjuntos de datos con valores faltantes no son robustos para el manejo de esta problemática, generando así modelos de estimación pocos eficientes [5].

En el Fuerte, Sinaloa, la minería es una de las actividades más intensas en desarrollo, la cual es causa principal de los cambios de concentración de metales pesados en cuerpos de agua.

Los parámetros del conjunto de datos de la calidad del agua en el río Fuerte obtenidos por la RENAMECA presentan datos perdidos, donde es indispensable la imputación de todos esos valores por los pocos registros para la implementación de estimación de los metales pesados.

Actualmente, existen técnicas para el manejo de datos perdidos, donde se hace uso de la eliminación directa de los registros, hasta la imputación de los valores, con la finalidad de obtener más datos de entrada para el modelo a generar de estimación [6].

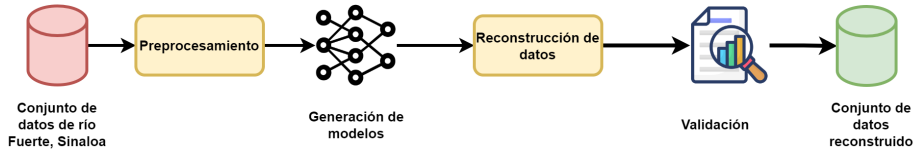


Fig. 1. Diagrama de imputación de conjunto de datos.

Entre las técnicas de imputación, existen las estadísticas como la media, la moda y la regresión lineal, las cuales fueron las primeras en implementarse. Por otra parte, existe la técnica capaz de imputar de manera múltiple los valores faltantes de diversos parámetros dentro del conjunto de datos denominado, imputación multivariada por ecuaciones encadenadas (MICE, por sus siglas en inglés).

Actualmente, la implementación de vecinos más cercanos K(KNN, por sus siglas en inglés), redes neuronales, árboles de decisiones y el aprendizaje profundo son los algoritmos más ampliamente utilizados [7].

Ante estas problemáticas, este artículo propone la implementación de tres modelos basados en vecinos más cercanos K y una red neuronal combinada con redes convolucionales para la reconstrucción individual de cada parámetro y en comparativa con la imputación múltiple mediante imputación multivariada por ecuaciones encadenadas (MICE).

Realizando la imputación de tres modelos aplicados en siete parámetros distintos de la calidad del agua, donde se simuló el borrado artificial de 10, 20 y 30 % para obtener mediante las métricas de MSE y RMSE el desempeño de estos modelos al comparar el conjunto de datos completo con el reconstruido. El siguiente artículo está organizado de la siguiente manera: Estado del arte, Materiales y Métodos, Resultados, Conclusiones y Trabajos Futuros.

2. Estado del arte

La imputación de los datos faltantes llevan casi 100 años de estudio, donde los principales métodos de reemplazo de valores perdidos fue la implementación de la media y moda, donde Milks (1932) se le atribuye la idea de la implementación de ellos.

Conforme fueron avanzando los años, surgieron diversas técnicas de imputación como imputación múltiple o métodos de regresión lineal y múltiple ([8]).

Actualmente, en las últimas décadas, el aprendizaje automático ha tomado gran relevancia en la imputación de datos faltantes, donde la implementación de redes neuronales y métodos de regresión basados en bosques aleatorios son ampliamente utilizados, permitiendo analizar conjuntos de datos más grandes y de mayor complejidad con mayor precisión y eficiencia.

Menéndez et al. [9] proponen la implementación de imputación controlada, es decir, imputan valores conocidos de un conjunto de datos de manera aleatoria para posteriormente compararlos con los valores reales conocidos. El conjunto de datos implementado son sobre contaminantes en una región de Polonia, implementando técnicas como imputación lineal, bosques aleatorios y vecinos más cercanos K.

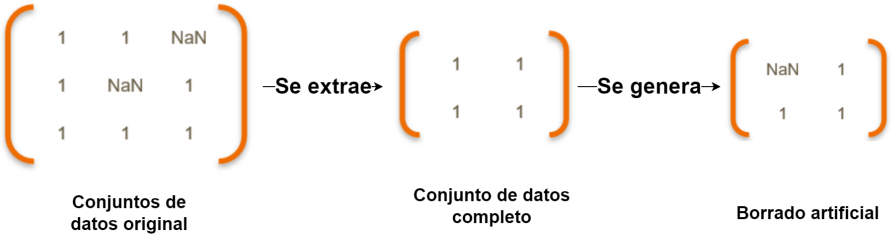


Fig. 2. Diagrama de borrado artificial.

Existen otros trabajos como el de Lin et al. [10], donde se hace la implementación de redes neuronales profundas, como el perceptrón multicapa y las redes de creencia profunda. En dicho trabajo, realizan la comparación con distintas técnicas como la media, KNN, árbol de clasificación y regresión (CART, por sus siglas en inglés) y máquina de soporte vectoriales (SVM, por sus siglas en inglés), llegando a la conclusión de que las redes de creencia profunda tienen mejor desempeño.

Camino et al. [11] proponen una solución de imputación de datos perdidos basada en modelos generativos profundos, donde se implementan autocodificadores de variación con distintas variantes implementados en conjuntos de datos con distintas características, donde la metodología para obtener los resultados fue la eliminación de valores al azar y reconstruirlos con varias técnicas.

Otras técnicas, aparte de las ya mencionadas para la imputación de datos faltantes, es el modelo bayesiano, el cual se propone en el trabajo de Zhai & Gutman [12], el cual se basa en los componentes de descomposición del valor singular de una matriz.

Por otra parte, en el desarrollo de Ratolojanahary et al. [13] realizan la combinación de técnicas de imputación multivariadas por ecuaciones encadenadas con métodos de aprendizaje automático para tratar la relación entre 200 parámetros de la calidad del agua. Los modelos con los que se combinaron fue, bosques aleatorios, árboles de regresión, vecinos más cercanos () y regresión vectorial de soporte.

La combinación de técnicas en distintos enfoques ha dado como resultado buenos algoritmos, en el caso de la imputación de datos faltantes, en el trabajo de Silva & Cabrera [14] se desarrolló un sistema de iferencia neuro-difusa co-activo llamado CANFIS-ART en el cual se automatiza el procedimiento de imputación de datos.

El modelo se construye a partir de las capacidades adaptativas de la red neuronal multicapa y el enfoque cualitativo de la lógica difusa utilizando el algoritmo fuzzy-ART. En el trabajo implementan la reconstrucción en 18 bases de datos, llegando al resultado que el enfoque propuesto por los autores superan totalmente los métodos de vanguardia y adicionalmente demuestran un mayor nivel de capacidad de generalización.

3. Metodología

Para la implementación de reconstrucción de datos faltantes, se hizo uso de un conjunto de datos proporcionado por CONAGUA a partir del río Fuerte de Sinaloa, el cual tienen 38 parámetros de la calidad del agua, en donde 28 de ellos cuentan con valores perdidos, los cuales se muestran en la Tabla 1.

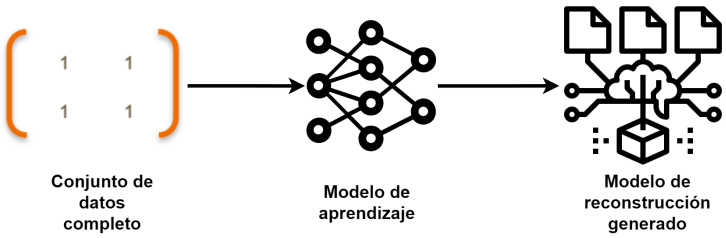


Fig. 3. Diagrama de generación de modelos.

3.1. Imputación de datos perdidos

La imputación de datos perdidos es una tarea que requiere distintos procesos para obtener una buena reconstrucción de los valores. La metodología planteada para el desarrollo de dicha tarea, es el generar un modelo por cada parámetro con datos faltantes y posteriormente realizar la imputación múltiple, con la finalidad de realizar una comparativa general, de cuál método resulta más eficiente.

El proceso de imputación de los valores de la calidad del agua está basado en el diagrama mostrado en la Figura 1:

El proceso de imputación será descrito por cada etapa para conocer a detalle que sucede en cada una de ellas.

Etapa de preprocesamiento

Para generar un modelo de aprendizaje es necesario ingresar los datos de tal manera que el método a implementar sea capaz de entender los datos ingresados.

Es decir, proveer los datos que sean útiles para que el modelo aprenda de ellos. Antes de ingresar los datos para generar los modelos, el conjunto de datos fue limpiada y normalizada por columna, para que tengan la mejor calidad posible de entrada,proceso de filtrado del conjunto de datos se muestra en la Figura 2.

La etapa de preprocesamiento está dividido en distintos procesos, donde se parte del conjunto de datos original del río Fuerte de Sinaloa y posteriormente generar un conjunto de datos completos y posteriormente realizar el borrado artificial con distintos porcentajes para su reconstrucción.

- **Conjunto de datos original:**
  El conjunto de datos original, son los registros del río Fuerte, en Sinaloa, de 38 parámetros en total, con alrededor de 357 registros de ellos. Cada parámetro tiene un porcentaje distinto de datos faltantes, es decir, valores que no fueron registrados en el conjunto de datos.
- **Generación de conjunto de datos completo:**
  La generación de conjunto de datos completo, consiste en que partiendo del conjunto de datos original, generar un conjunto de datos donde se realice la eliminación de aquellas columnas o también conocido como parámetros, con porcentajes altos de datos faltantes y posteriormente eliminar aquellas filas que tengan faltante de registros en alguno de los parámetros.

Tabla 2. Arquitectura de red convolucional con red neuronal artificial.

Capas	Kernel	Filtro	Dimensión de salida
Conv1	2x2	32	37x1
Flatten	-	-	37
Dense1	-	-	32
Dense2	-	-	1

Por otra parte, otras de las condiciones implementadas para la eliminación de columnas, parte de eliminar aquellos parámetros que tengan un comportamiento constante, debido a que no aportan nada al generar modelos donde se requiere reconstruir o estimar.

De tal manera que, se obtuvo un conjunto de datos sin valores faltantes, con el cual se redujo el de 357 registros a 273 y de 38 parámetros se redujo a 32 en total. A partir de este conjunto de datos completo, se implementó la normalización de datos por cada columna y tener estandarizado los formatos de los datos entrada para mayor eficiencia de procesamiento y mejores resultados.

– **Borrado artificial:**
Para realizar validaciones de la reconstrucción de datos faltantes, es necesario generar, a partir del conjunto de datos completo, un segundo conjunto de datos con la finalidad de eliminar o generar de manera aleatoria distintos porcentajes de pérdidas en los parámetros que se desea evaluar su reconstrucción.

De esta manera, se cuenta con dos conjuntos de datos, una con los datos completos y la segunda con el conjunto de datos con borrado artificial de manera aleatoria, para posteriormente realizar la reconstrucción en ella y compararla con el primer conjunto de datos y así obtener la métrica de evaluación.

Generación de modelos

Durante esta etapa se determinó el conjunto de parámetros a reconstruir, implementando aquellos que tuvieran un porcentaje de como cerca del 10 % y posteriormente implementar dos modelos de reconstrucción de manera individual por cada parámetro y realizar la reconstrucción múltiple de estos. La Figura 3 muestra el proceso que se siguió desde la entrada de los datos completos hasta la generación de un modelo de imputación o reconstrucción.

– **Modelo de aprendizaje:**
Se hizo uso de distintos modelos de aprendizaje para realizar la reconstrucción de datos, con la finalidad de tener una comparativa entre los diversos métodos que existen para generar la imputación de valores faltantes.

Los modelos implementados para la imputación fue el uso de vecinos más cercarnos K (KNN por sus siglas en inglés), debido a que es uno de los métodos más ampliamente utilizado para problemas de regresión, donde el valor de K solamente se exploró el 3.

Por otra parte, con la finalidad de explorar técnicas de aprendizaje profundo, se implementó las redes convolucionales combinada con red neuronal artificial,

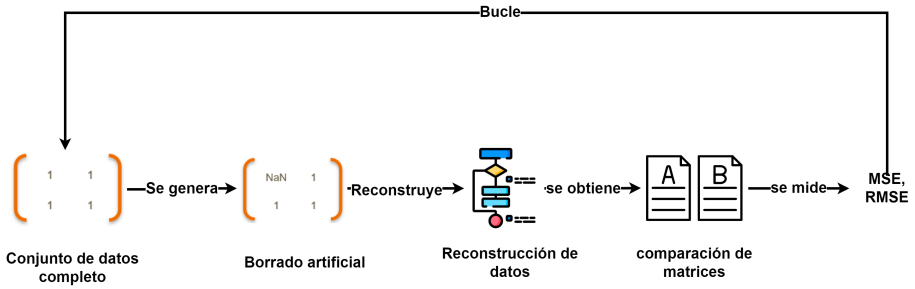


Fig. 4. Diagrama de bucle de obtención de resultados.

la configuración esta dada por la Tabla 2, se hizo uso de una de las técnicas de imputación múltiple denominada imputación múltiple por ecuación en cadena (MICE por sus siglas en inglés), esto con la finalidad de comparar la imputación de todos los parámetros a la vez y la reconstrucción individual por parámetro.

– **Modelo de reconstrucción generado:**
Los modelos de reconstrucción de cada método de aprendizaje generados por cada parámetro se implementaron en el conjunto de datos con borrado artificial, con la finalidad de realizar la comparación de ambas matrices, lo cual lleva a la etapa de obtención de resultados.

Validación:

La etapa de validación, se enfoca en realizar repetitivamente la generación de un conjunto de datos con borrado artificial en distintos porcentajes y como paso siguiente realizar la reconstrucción con los distintos modelos generados.

De tal manera que, para las pruebas, se implementó el borrado artificial en los parámetros en porcentajes de 10 %, 20 % y 30 % para posteriormente realizar la reconstrucción en el conjunto de datos con borrado artificial.

Una vez realizada la reconstrucción, se realiza una comparación entre la matriz de datos completa y la reconstruida para así calcular las métricas del error cuadrático medio (MSE por sus siglas en inglés) y raíz cuadrada del error cuadrático medio (RMSE por sus siglas en inglés).

Las métricas de MSE y RMSE son implementadas ampliamente para la evaluación de modelos de regresión, donde ambas describen el rendimiento de los modelos generados y a su vez realizar comparación diferentes modelos de distintas técnicas implementadas.

El MSE está definido por la ecuación 1, la cual describe la diferencia promedio al cuadrado que existe entre los valores estimados del parámetro reconstruido de la calidad del agua y el valor real, en donde, entre más cercano sea el valor a 0, indica una menor diferencia entre ambos valores:

$$MSE = \frac{1}{n}\sum_{i=1}^{n} \langle y_i + \hat{y} \rangle^2 . \tag{1}$$

Tabla 3. Resultados de imputación con valores faltantes de 10 %.

Parámetro	Modelo	MSE	RMSE
COLI_TOT	KNN	0.0536	0.2314
	MICE	0.0524	0.2288
	ANN + CNN	0.0608	0.2465
E_COLI	KNN	0.0109	0.1042
	MICE	0.0102	0.1008
	ANN + CNN	0.0072	0.0851
DBO_SOL	KNN	0.0144	0.1200
	MICE	0.0113	0.1061
	ANN + CNN	0.0127	0.1129
DBO_TOT	KNN	0.0207	0.1439
	MICE	0.0185	0.1359
	ANN + CNN	0.0231	0.1520
DQO_SOL	KNN	0.0013	0.0361
	MICE	0.0012	0.0340
	ANN + CNN	0.0023	0.0047
DQO_TOT	KNN	0.0081	0.0898
	MICE	0.0061	0.0783
	ANN + CNN	0.0060	0.0775
DUR_TOT	KNN	0.0099	0.0993
	MICE	0.0051	0.0717
	ANN + CNN	0.0040	0.0634

Por otra parte, el RMSE está definido por la ecuación 2, el cual es la obtención de la raíz cuadrada de MSE e indica el error promedio de estimación en relación con la variable que se estima. En este caso particular, al ser normalizados los datos entre 0 y 1, el RMSE nos indica en un porcentaje cuál es la tasa de error entre el resultado obtenido con el real:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\langle y_i + \hat{y}\rangle^2}. \tag{2}$$

Ahora bien, el proceso de reconstrucción se iteró 10 veces para generar un promedio entre los resultados del MSE y RMSE, con la finalidad de tener un valor más acertado al momento de realizar la reconstrucción de parámetros. La Figura 4 muestra el proceso que se sigue para la obtención de resultados.

4. Resultados

Los resultados obtenidos están enfocados en la aplicación de tres modelos distintos para la imputación de datos con distintos porcentajes para siete parámetros en total, donde estos parámetros se seleccionaron con la condición de tener un máximo de 10 % de faltantes de manera real.

Como primera parte, la Tabla 3 presenta los resultados de la reconstrucción de los siete parámetros con el borrado artificial de 10 % faltante, donde, las métricas de MSE y RMSE indican el desempeño de la imputación con las condiciones dadas.

Tabla 4. Resultados de imputación con valores faltantes de 20 %.

Parámetro	Modelo	MSE	RMSE
COLI_TOT	KNN	0.0603	0.2456
	MICE	0.0563	0.2372
	ANN + CNN	0.0704	0.2653
E_COLI	KNN	0.0080	0.0895
	MICE	0.0073	0.0853
	ANN + CNN	0.0037	0.0606
DBO_SOL	KNN	0.0144	0.1199
	MICE	0.0107	0.1032
	ANN + CNN	0.0088	0.0937
DBO_TOT	KNN	0.0205	0.1431
	MICE	0.0188	0.1371
	ANN + CNN	0.0198	0.1406
DQO_SOL	KNN	0.0023	0.0475
	MICE	0.0020	0.0449
	ANN + CNN	0.0019	0.0433
DQO_TOT	KNN	0.0053	0.0727
	MICE	0.0045	0.0674
	ANN + CNN	0.0060	0.0776
DUR_TOT	KNN	0.0044	0.0664
	MICE	0.0039	0.0627
	ANN + CNN	0.0047	0.0689

El análisis de los resultados de MSE en los distintos parámetros reconstruidos a un 10 % de faltantes, existe muy poca diferencia en el resultado por cada modelo generado, obteniendo MSE muy cercanos a 0, indicando que la diferencia entre los valores reales y los imputados son muy similares.

Por otra parte, al observar el RMSE indica el porcentaje de error con el que tiende a equivocarse entre estos parámetros, donde en la mayoría de los parámetros el error radica entre un ± 10 %.

La Tabla 4 presenta los resultados obtenidos de la imputación de valores en un 20 %. Aunque, se aumentó el número de valores faltantes, los valores de RMSE y MSE entre los modelos generados y cada uno de los parámetros se mantienen con una ligera diferencia, manteniendo seis de los parámetros con una diferencia cerca del 10 % a excepción de Coliformes Totales que presenta un RMSE relativamente alto entre un 25 %.

Por último, los resultados obtenidos para la imputación del 30 % se muestran en la Tabla 5, donde los modelos y parámetros mostraron un comportamiento similar de error en el RMSE, que en los anteriores porcentajes de datos faltantes.

Obteniendo así, las pruebas de tres distintos modelos como Vecinos más cercanos, combinación de una red neuronal artificial con redes convolucionales para la imputación individual de cada parámetro y posteriormente el MICE, que representa la imputación múltiple de las siete variables a la vez.

Tabla 5. Resultados de imputación con valores faltantes de 30 %.

Parámetro	Modelo	MSE	RMSE
COLI_TOT	KNN	0.0660	0.2569
	MICE	0.0558	0.2362
	ANN + CNN	0.0690	0.2626
E_COLI	KNN	0.0064	0.0799
	MICE	0.0058	0.0764
	ANN + CNN	0.0052	0.0724
DBO_SOL	KNN	0.0110	0.1047
	MICE	0.0086	0.0930
	ANN + CNN	0.0105	0.1025
DBO_TOT	KNN	0.0258	0.1606
	MICE	0.0231	0.1521
	ANN + CNN	0.0232	0.1522
DQO_SOL	KNN	0.0017	0.0416
	MICE	0.0020	0.0450
	ANN + CNN	0.0038	0.0615
DQO_TOT	KNN	0.0078	0.0884
	MICE	0.0055	0.0741
	ANN + CNN	0.0056	0.0746
DUR_TOT	KNN	0.0039	0.0628
	MICE	0.0034	0.0579
	ANN + CNN	0.0065	0.0806

Comparando los resultados de los tres distintos borrados artificiales para su reconstrucción, se puede observar que no hay una diferencia relevante entre los distintos porcentajes de faltantes, es decir, aunque se tenga 10, 20 o 30 % de datos perdidos, la imputación de dichas variables suele tener un margen de error muy similar entre los distintos modelos de cada parámetro.

De tal manera que, los distintos porcentajes de datos faltantes no influyen en que tan preciso va a ser la imputación, sino la correlación que existe entre los parámetros de entrada para tomar en cuenta la reconstrucción.

El contexto de la información aportada por la base de datos es la que determina que tan buena imputación va a realizar al momento de implementar cualquiera de las técnicas.

5. Conclusiones

El determinar qué método es el mejor para la imputación de valores perdidos en cualquier tipo de variable es difícil. Se necesita explorar todos los métodos existentes de imputación.

Ahora bien, entre los métodos explorados, no existe uno mejor o peor. Debido a los resultados obtenidos, se observa que los resultados no se relacionan con el porcentaje de valores perdidos, sino del hecho de las correlaciones que existen entre los valores a imputar con respecto a las variables de entrada.

Uno de los puntos más importantes dentro de las tareas de aprendizaje máquina, es contar con buenos conjuntos de datos que sean capaces de describir las características de una problemática dada, el hecho de contar con conjunto de datos son valores faltantes se vuelve un reto, ya que no pueden ser descartados simplemente, sino que es necesario encontrar aquellas técnicas que ayuden aportar más información de entrada a las técnicas de aprendizaje máquina y así obtener resultados favorables.

6. Trabajo futuro

Con los resultados obtenidos, se planea explorar otras técnicas del estado del arte, como modelos bayesianos e implementación de bosques aleatorios, para así determinar si existe una mejora en la imputación de datos perdidos mediante estas técnicas.

Posteriormente, es necesario terminar la reconstrucción de todos los parámetros con datos faltantes de la calidad del agua para así realizar una comparativa de predicción de metales pesados en el río fuerte de Sinaloa, con un conjunto de datos reconstruido y sin reconstruir, para determinar si existe una gran mejora con los valores imputados.

Una vez obtenido los resultados, la misma metodología de estimación de metales, se va a extrapolar a la predicción de otros parámetros con las entradas reconstruidas, como pH, turbidez, coliformes fecales.

Referencias

1. Jäger, S., Allhorn, A., Bießmann, F.: A benchmark for data imputation methods. Frontiers In Big Data, vol. 4 (2021) doi: 10.3389/fdata.2021.693674
2. Jain, A., Patel, H., Nagalapatti, L., Gupta, N., Mehta, S., Guttula, S., Mujumdar, S., Afzal, S., Sharma-Mittal, R., Munigala, V.: Overview and importance of data quality for machine learning tasks. In: Proceedings Of The 26th ACM SIGKDD International Conference On Knowledge Discovery and Data Mining, pp. 3561–3562 (2020) doi: 10.1145/3394486.3406477
3. Gudivada, V., Apon, A., Ding, J.: Data quality considerations for big data and machine learning: Going beyond data cleaning and transformations. International Journal On Advances In Software, vol. 10, no. 1-2 (2017)
4. Comisión Nacional del Agua: Calidad del agua en México (2023) https://www.gob.mx/conagua/articulos/calidad-del-agua
5. Jadhav, A., Pramod, D., Ramanathan, K.: Comparison of performance of data imputation methods for numeric dataset. Applied Artificial Intelligence, vol. 33, no. 10, pp. 913–933 (2019) doi: 10.1080/08839514.2019.1637138
6. Lin, W., Tsai, C.: Missing value imputation: a review and analysis of the literature (2006–2017). Artificial Intelligence Review, vol. 53, pp. 1487–1509 (2020) doi: 10.1007/s10462-019-09709-4
7. Enders, C. K.: Applied missing data analysis. Guilford Publications (2022)
8. McKnight, P., McKnight, K., Sidani, S., Figueredo, A. J.: Missing data: A gentle introduction. Guilford Press (2007)
9. Menéndez-García, L., Menéndez-Fernández, M., Sokoła-Szewioła, V., Prado, L., Ortiz-Marqués, A., Fernández-López, D., Sánchez, A. B.: A method of pruning and random replacing of known values for comparing missing data imputation models for incomplete air quality time series. Applied Sciences, vol. 12, no. 6465 (2022) doi: 10.3390/app12136465

10. Lin, W., Tsai, C., Zhong, J.: Deep learning for missing value imputation of continuous data and the effect of data discretization. Knowledge-Based Systems, vol.239, no. 108079 (2022) doi: 10.1016/j.knosys.2021.108079
11. Camino, R., Hammerschmidt, C., State, R.: Improving missing data imputation with deep generative models. (2019)
12. Zhai, R., Gutman, R.: A Bayesian singular value decomposition procedure for missing data imputation. Journal Of Computational And Graphical Statistics, vol. 32, no. 2, pp. 470–482 (2022) doi: 10.1080/10618600.2022.2107534
13. Ratolojanahary, R., Ngouna, R., Medjaher, K., Junca-Bourié, J., Dauriac, F., Sebilo, M.: Model selection to improve multiple imputation for handling high rate missingness in a water quality dataset. Expert Systems With Applications, vol. 131, pp. 299–307 (2019) doi: 10.1016/j.eswa.2019.04.049
14. Silva-Ramírez, E., Cabrera-Sánchez, J. F.: Co-active neuro-fuzzy inference system model as single imputation approach for non-monotone pattern of missing data. Neural Computing And Applications, vol. 33, pp. 8981–9004 (2021) doi: 10.1007/s00521-020-05661-5

ISSN 1870-4069

Detector de ondas gravitacionales fenomenológicas de supernovas basado en aprendizaje supervisado

César Eduardo Tiznado Alonso[1], Claudia Moreno[2], Manuel D. Morales[2], Mauricio Antelis[1]

[1] Instituto Tecnológico y de Estudios Superiores de Monterrey, Escuela de Ciencias e Ingenierías, México

[2] Universidad de Guadalajara, Centro Universitario de Ciencias Exactas e Ingeniería, México

{A00838226, mauricio.antelis}@tec.mx, claudia.moreno@academico.udg.mx, manueld.morales@academicos.udg.mx

Resumen. Debido a que la detección de Ondas Gravitacionales (OG) provenientes de supernovas es una tarea que no se ha llegado a concretar dada a la falta de modelos para su caracterización gracias a la naturaleza estocástica de la señal, se propone una alternativa, un modelo de Machine Learning de aprendizaje supervisado, el cual, utilizando señales sintéticas generadas por un modelo fenomenológico de supernovas y Ruido proveniente de los detectores de OG, generar un clasificador binario que nos indique la presencia de una OG. Para la generación del set de datos de entrenamiento y prueba de nuestro modelo, se procedió a la inyección se las señales sintéticas sobre una muestra de $4096s$ del ruido obtenido de los detectores de OG. Al separar los datos inyectados en ventanas de tiempo, y caracterizar la presencia de una OG o solo ruido, se procedió a la extracción de las características principales del modelo, siendo el rango tiempo-frecuencia de la señal, la cual nos caracterizará la clasificación de nuestro modelo. Con el set de datos preparado, se procede a utilizar un algoritmo de K-vecinos más cercanos, debido a la simplicidad del modelo de aprendizaje. Al hacer las predicciones, obtenemos que el modelo es capaz de distinguir entre el ruido y la señal gravitacional.

Palabras clave: Onda gravitacional, supernova, fenomenológica, KNN, métricas

Detector of Phenomenological Gravitational Waves from Supernovae based on Supervised Learning

Abstract. Due to the difficulty in detecting Gravitational Waves (GW) from supernovae due to the lack of models for their characterization, given the stochastic nature of the signal, an alternative is proposed: a supervised

Machine Learning model. This model uses synthetic signals generated by a phenomenological model of supernovae and noise from GW detectors to create a binary classifier that indicates the presence of a GW. To generate the training and testing dataset for our model, the synthetic signals were injected into a $4096s$ sample of noise obtained from the GW detectors. By separating the injected data into time windows and characterizing the presence of a GW or just noise, the main features of the model were extracted, focusing on the time-frequency range of the signal, which characterizes the classification of our model. With the prepared dataset, a K-nearest neighbors algorithm was used due to the simplicity of the learning model. The predictions show that the model is capable of distinguishing between noise and gravitational signal.

Keywords: Gravitational wave, supernova, phenomenological, KNN, metrics.

1. Introducción

Las ondas gravitacionales son perturbaciones en el espacio-tiempo producidas por masas en movimiento, los productores astrofísicos de estas ondas abarcan desde: La colisión de agujeros negros, fusiones de estrellas de neutrones, Cuásares, Pulsares y Supernovas.

La importancia de la detección de este tipo de señales astronómicas es que nos permite compilar información desde otra perspectiva de los fenómenos astrofísicos que suceden en el universo, los cuales no son posibles de obtener por métodos convencionales, siendo la astronomía de Ondas Gravitacionales una parte fundamental de la llamada Astronomía de Multimensajeros.

El campo nuevo de Astronomía de Multimensajeros apunta hacia el estudio de las fuentes astronómicas mediante diferentes tipos de "Mensajeros": Fotones, Neutrinos, Rayos Gamma y Ondas Gravitacionales [14].

La detección de este tipo de OG ha sido nuevo para el campo de la Física, esto debido a que a pesar de que son una consecuencia directa de la Teoría de la Relatividad General de Einstein publicada en 1915 [6], no fue hasta la primera detección de este tipo de onda el 14 de mayo del 2015 [2], confirmando su existencia.

Uno de los problemas al hacer este tipo de detecciones recae en el tipo de emisor que genera la OG, siendo las colisiones de objetos compactos(CBC's): Agujeros negros y estrellas de neutrones, las que se encuentran modeladas por métodos numéricos y postnewtonianos los cuales nos permiten tener una gran cantidad de modelos con los cuales podemos detectar una señal mediante matched filtering, pero, ¿Qué es lo que pasa si nosotros deseamos obtener un modelo para emisores como lo son las supernovas?

Las Supernovas se caracterizan por ser procesos estocásticos a diferencia de las CBC's que son deterministas, siendo bastante complicadas de modelar mediante relatividad numérica, que, para conseguir el modelo de una señal multidimensional, pueden tardar en un procesamiento cerca de 3×10^5 horas-núcleo (core-hours) [15] debido a la complejidad de procesos físicos que ocurren dentro y sobre las inmediaciones de la explosión de la estrella, así como la falta de certitud en la evolución y desarrollo de los mismos.

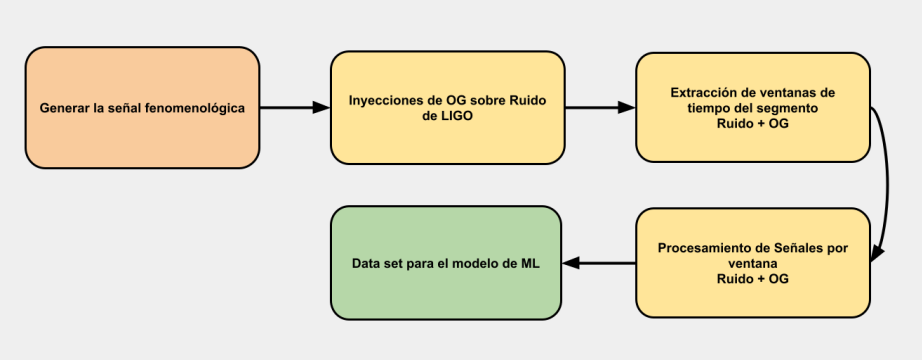


Fig. 1. Diagrama de la estructura de las tareas a realizar para la construcción del dataset para el entrenamiento del modelo de aprendizaje automático. Se inicia con la generación de las señales fenomenológicas, se prosigue con las inyecciones de las señales en ruido de los detectores, se continúa con la extracción de las ventanas imbuidas en ruido y se procesan las ventanas para obtener las características importantes para así obtener el dataset.

En este trabajo se propone utilizar un algoritmo de aprendizaje automático, para reconocer o discriminar entre una señal gravitacional emitida por una supernova imbuida en ruido y ruido de los detectores.

A partir de aquí, se propone utilizar un modelo fenomenológico, que aprovechando la morfología de las OG producidas por supernovas se pueden generar un gran número de formas de onda, teniendo así disponibles la capacidad para crear un set de datos que se podrá utilizar para la construcción de un modelo de aprendizaje supervisado para generar un detector de Ondas Gravitacionales producidas por supernovas.

La importancia de este trabajo radica en el entrenamiento de un clasificador de OG con señales fenomenológicas de supernovas, el cual puede ser utilizado para próximas detecciones de OG provenientes de supernovas.

La estructura del artículo consta de una introducción a la problemática en la primera sección, la segunda sección consta de la generación del dataset a partir del preprocesamiento de las señales fenomenológicas y el ruido de los detectores de OG, la tercera sección se centra en la clasificación llevada a cabo mediante el modelo de aprendizaje, la cuarta sección en los resultados obtenidos por el clasificador y las métricas obtenidas, y por último, en la quinta sección se hacen las disertaciones necesarias a partir de los resultados obtenidos del proceso de aprendizaje y se contemplan las perspectivas futuras a seguir en próximos trabajos.

2. Construcción del dataset

Las modelaciones multidimensionales de este tipo de señales producidas por resolución de ecuaciones de estado [3], tienden a tener grandes tiempos de procesamiento para solo generar una sola señal de onda, es por eso por lo que los datos que se utilizarán provienen desde una aproximación fenomenológica. En esta sección nos centraremos en el preprocesamiento de datos para la generación de nuestro dataset, siguiendo el diagrama de la Fig. 1.

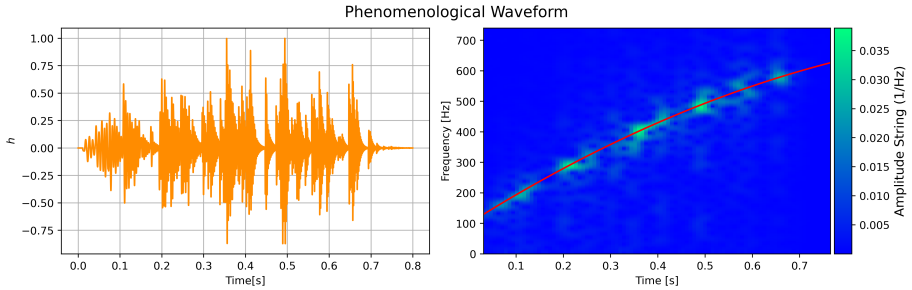


Fig. 2. Ejemplo de una forma de onda Fenomenológica generado con el modelo sintético. En la izquierda podemos observar la señal gravitacional correspondiente con una amplitud normalizada, mientras que en la parte derecha observamos la misma señal, pero descrita en tiempo-frecuencia, en donde se observa el modo de oscilación que será la característica principal para la clasificación de estas señales estocásticas.

2.1. Generación de ondas gravitacionales fenomenológicas

Desarrollando un modelo fenomenológico, el cual genera una forma de onda, la cual imite a los modelos recuperados de las simulaciones multidimensionales ([16, 13, 12, 9, 4]), intentando que tome el mayor número de características, todo para no estar lejos del rigor científico que requieren este tipo de detecciones.

Dentro de este tipo de simulaciones es posible recopilar uno de los rasgos principales en la evolución de las supernovas, modos de oscilación. Dentro de estos procesos astrofísicos, los modos de oscilación tienden a ser caracterizados por tener un aumento monotónico en una ventana de tiempo comenzado a frecuencias de $100Hz$ [5].

Partiendo de este punto, se propone usar un modelo físico que imite a las perturbaciones esperadas, por lo cual se opta por el uso de un oscilador armónico agregando un forzamiento aleatorio, el cual será responsable de la dinámica estocástica de la forma de onda, este forzamiento aleatorio está definido como $s(t)$.

Debido a la necesidad de un factor de amortiguamiento, y siendo este modelo una aproximación fenomenológica de la problemática, se opta por utilizar factor de calidad Q para controlar que tipo de amortiguamiento va a gobernar nuestro oscilador. Utilizando la relación del *factor-Q* y el factor de amortiguamiento β, tenemos la relación final de nuestro oscilador armónico sub amortiguado:

$$\frac{dh(t)}{dt^2} + \frac{\omega(t)}{Q}\frac{dh(t)}{dt} + \omega(t)^2 h(t) = s(t). \tag{1}$$

Se da paso a su resolución aplicando el método de Euler [7]. Se define un arreglo para la función $h(t)$, la cual tiene la misma longitud que la ventana de tiempo t y cada intervalo está definido por una diferencia finita de tiempo, la cual se define como el inverso de la Frecuencia de muestreo (Fm).

Con el fin de tener la mayor cantidad posible de información producida por nuestro modelo, se opta por utilizar una $Fm = 200KHz$. La resolución de esta perturbación puede ser escrita mediante la relación en nodos computacionales, tal que satisfacen:

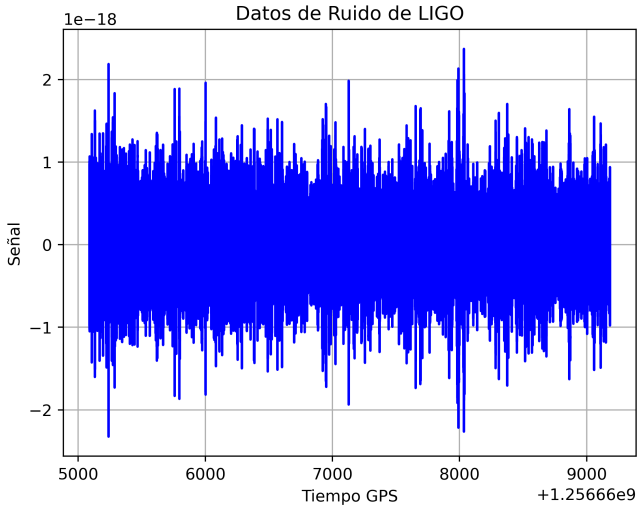


Fig. 3. Ruido proveniente del Detector H1 de LIGO, con una duración de $4096s$ y una frecuencia de muestreo de $16Khz$.

$$\begin{aligned} F[i] &= s[i] - (w[i]/Q)(dh[i]) - w[i]^2, \\ dh[i] &= dh[i-1] + (F[i-1])dt, \\ h[i] &= h[i-1] + (dh[i-1])dt. \end{aligned} \tag{2}$$

Ya con las formas de onda producidas, se procede a un preprocesamiento de las señales con un remuestreo de la señal a una frecuencia de muestreo de $16Khz$, con el fin de tener la señal dentro de los rangos de muestreo utilizados por los detectores de ondas gravitacionales para así tener un modelo escalable.

Ya con la señal fenomenológica dentro de las características adecuadas (Fig.2), es necesario para la construcción del data set que se utilizará para entrenar el modelo, la consideración de que este tipo de señales en la naturaleza están acompañadas de ruido cuando son percibidas por los detectores, debido a esto, se necesita "inyectar"nuestra señal fenomenológica en ruido el cual, es percibido dentro de los detectores bajo condiciones normales (cuando no existe ninguna detección de OG).

2.2. Descripción del ruido

Una de las partes importantes para la identificación y caracterización de las ondas gravitacionales por los grupos de investigación es el uso de los datos obtenidos por los detectores de Ondas Gravitacionales. Estos observatorios son denominados LIGO (Observatorio de Ondas Gravitacionales con Interferómetro Láser) por sus siglas en inglés.

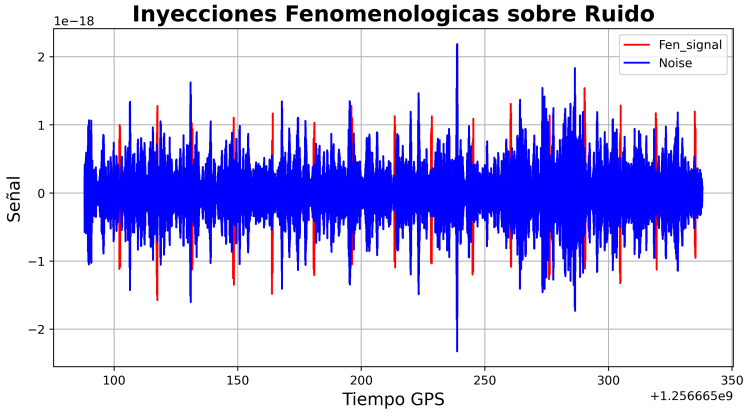


Fig. 4. Inyección de la señal fenomenológica en una muestra de 250s, en la cual podemos observar en la parte roja las 16 inyecciones realizadas, mientras que en la parte azul se encuentra la señal de ruido correspondiente al detector H1 de LIGO.

Los detectores de ondas gravitacionales de escala de varios kilómetros de LIGO utilizan interferometría láser para medir las ondas diminutas en el espacio-tiempo causadas por el paso de ondas gravitacionales de eventos cósmicos cataclísmicos.

LIGO consta de dos interferómetros ampliamente separados dentro de los Estados Unidos, uno en Hanford, Washington y el otro en Livingston, Louisiana, que funcionan al unísono para detectar ondas gravitacionales [10].

Muchas señales no gravitacionales pueden mover los espejos o afectar la luz láser de una manera que puede imitar o enmascarar una señal de onda gravitatoria. Estos ruidos son generados por muchos fenómenos físicos diferentes.

Los detectores han sido diseñados para que estos ruidos sean extremadamente silenciosos, pero las señales de ondas gravitacionales son igualmente pequeñas, e incluso las fuerzas más imperceptibles en los espejos son suficientes para estropear la medición [1].

Dentro de las fuentes se pueden encontrar: *Ruido sísmico, ruido térmico, Ruido por cargas eléctricas, así como el ruido provocado por los láseres, entre otros. Sobre esta muestra de ruido es donde realizaremos las inyecciones de nuestras señales fenomenológicas.*

Tomando estos aspectos en cuenta, es predecible que en los datos recabados en los sets de observaciones se encuentren señales que son atribuidas a todas estas fuentes de ruido, por lo cual es necesario que las señales fenomenológicas estén concatenadas en este tipo de ruido para así poder generar un set de datos de entrenamiento que se encuentre en condiciones naturales del fenómeno.

El set de datos de ruido proveniente de LIGO utilizados en este trabajo (Fig. 3) son recopilados desde el *Gravitational Waves Open Science Center* (GWOSC) y sus características corresponden a una frecuencia de muestreo de $16Khz$, duración de $4096s$, perteneciente al detector $Livingston$ y integra al set de datos $03b$.

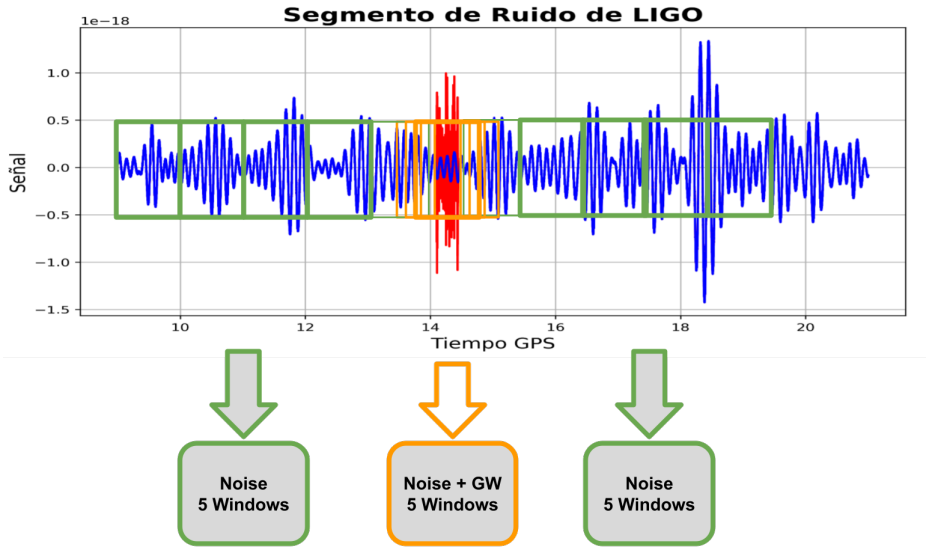


Fig. 5. Representación gráfica de la segmentación de los datos, en la cual la señal roja representa la inyección de la señal sintética, mientras en azul el ruido del detector. Se puede observar que se toman 5 ventanas de tiempo antes y después de la señal que corresponden a información correspondiente a solo el ruido, mientras que sobre la señal se toman 5 ventanas diferentes que corresponden a la señal sintética imbuida en el ruido.

2.3. Inyecciones

Con los datos del ruido sobre los cuales se trabajará, se procede a las inyecciones de las señales sintéticas. Este proceso es usualmente llevado a cabo mediante software especializado, pero uno de los problemas por los cuales no es posible utilizar herramientas ya construidas para este proceso recae en que las señales deben contener información de la distancia, los cuales reflejan la intensidad de la señal y la cantidad de energía que pueden emitir estos fenómenos.

Para este trabajo, en el cual se usa una señal fenomenológica que cuenta con la ventaja de generación a un bajo coste, pero que su amplitud está normalizada para poder usarse para los parámetros que sean convenientes, se procederá a utilizar la relación señal a ruido (SNR), el cual al seleccionar un parámetro obtendremos una relación directa entre la señal inyectada y el ruido que se está utilizando, dentro de estos valores es posible calcular cuál es el factor para que la amplitud de la señal seleccionada este bajo las condiciones deseadas.

Se procede a inyectar la señal varias veces la señal con una respectiva ventana de tiempo de $16\,segundos$, marcando el inicio de la inyección en el tiempo t_i pero que se encontrara variado por la inclusión de un jitter con valores entre (-2,2)s de la señal.

Un ejemplo sería tener el tiempo $t_1 = 16s$ del ruido sobre el cual se efectuara la primera inyección, pero con un jitter de $-1s$, el cual nos daría nuestra inyección en el tiempo $t_1 = 15s$.

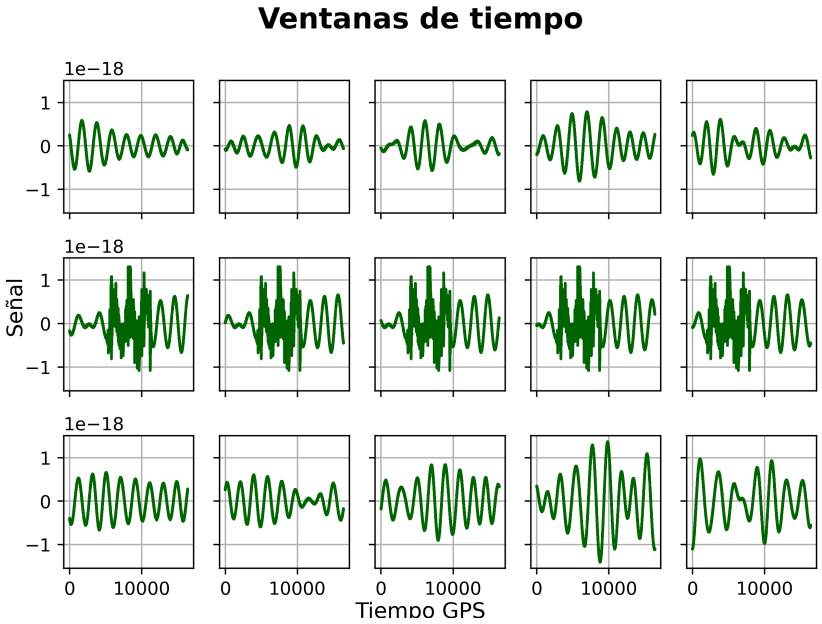


Fig. 6. Ventanas de tiempo tomadas para un segmento de las inyecciones. (Superior) Ventanas previas a la inyección con solo ruido, (Central) Ventanas las cuales contienen la inyección de la señal sintética, (Inferior) Ventanas posteriores a la inyección, solo contienen ruido.

Ya con los parámetros de cada ventana de tiempo y un jitter que va cambiando entre sus rangos predefinidos, procedemos a efectuar las inyecciones sobre la señal de ruido, como lo mostramos en la Fig. 4, en los cuales tomamos solo un rango de $250s$ de la señal para mostrar las inyecciones en rojo, mientras en azul se encuentra solo la señal inherente al ruido. Al final se logran obtener alrededor de 255 inyecciones sobre toda la ventana de tiempo perteneciente al ruido.

2.4. Extracción de ventanas de tiempo

Con nuestra señal de tiempo ya inyectada, uno de los procesos seguidos para la generación de nuestro data set, es segmentar esta información a partir de pequeñas ventanas de tiempo [11] (Fig. 5), sobre las cuales será mucho más fácil manejar la información debido a la gran extensión de recursos que se requiere para procesar cerca de 6×10^6 segmentos de datos.

Teniendo esto en cuenta, haremos una segmentación alrededor de la señal que consta de $5s$ antes de la inyección y $5s$ posteriores al terminar la inyección de la forma de onda.

Si consideramos que las ventanas que tomaremos tiene la longitud de $1s$ y procederemos a tomar 15 ventanas del problema, 10 correspondientes a obtener muestras del ruido, mientras 5 se centraran en la obtención de la señal inyectada sobre el ruido, una muestra de esto es la Fig. 6.

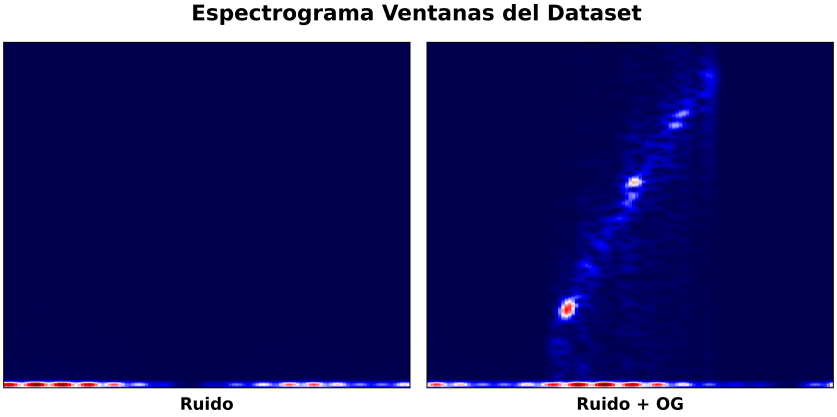


Fig. 7. Espectrogramas de las ventanas de tiempo, (Izquierda) Corresponde a la relación tiempo-frecuencia de la señal perturbativa del detector de OG, (Derecha) Relación tiempo frecuencia de la inyección de la señal fenomenológica en el ruido del detector, mostrando la característica principal de esta señal, el modo de oscilación de la OG de supernovas.

2.5. Procesamiento de datos

Al extraer las ventanas de las 255 inyecciones, se logran obtener alrededor de 3825 señales, las cuales constan del $67\,\%$ pertenecientes a ruido de los detectores y el otro $33\,\%$ corresponden a una señal gravitacional imbuida en ruido.

Debido a que nuestro interés recae sobre la clasificación de estas señales utilizando un modo de oscilación, se necesita ejecutar de un análisis de tiempo-frecuencia de cada ventana de tiempo, por lo tanto calculando el espectrograma para cada una de las ventanas obtenidas, se adquieren las ventanas necesarias para la creación de nuestro data set, el cual alimentara nuestro modelo de aprendizaje (Fig. 7).

Para poder continuar con el trabajo, se necesitan extraer las componentes principales que se encuentran en cada espectrograma obtenido, por lo cual, adoptando una estrategia de extracción de la componente espectral de cada una de las imágenes, se lleva a cabo mediante los valores pertenecientes a cada pixel de cada una de estas gráficas de relación tiempo-frecuencia.

Además, si reducimos la información que proviene de un formato 3-dimensional como lo es él RGB a una interpretación 1-dimensional B/N lograremos atenuar la cantidad de datos a procesar y complementando podemos reducir la cantidad de píxeles con los que contará cada una de las ventanas, logrando una reducción importante de datos sobre los cuales ninguna característica principal recibirá una potencial penalización por el modelo. Para cada una de las imágenes que será utilizada para este trabajo consta de una dimensión de 28×28 píxeles (Fig. 8).

Debido a la estrategia de clasificación que utilizaremos, se requiere el procesamiento de los datos, el cual consta de una vectorización de los datos de entrada, por lo tanto, realizando un ajuste dimensional de los datos obtenidos por cada imagen, reducimos de una matriz de 28×28 a un vector de 1×784 tendremos un vector de 784 valores por cada ventana, creando así el set de datos para el entrenamiento y test de nuestro modelo de aprendizaje.

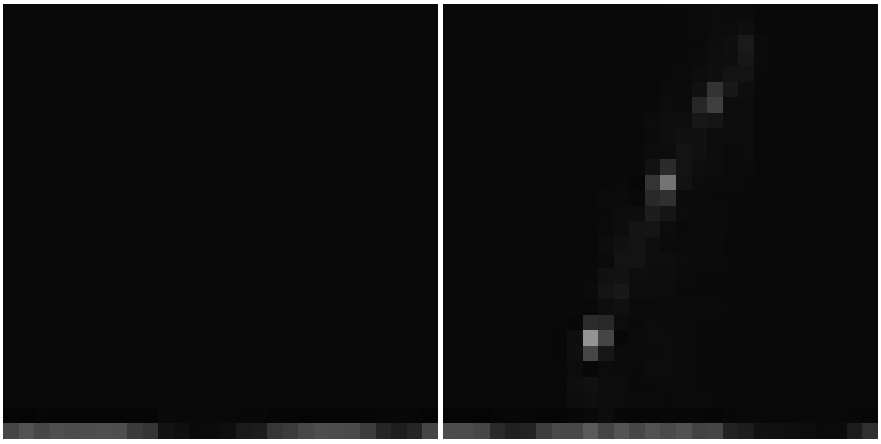


Fig. 8. Comparación de los elementos en el set de entrenamiento. Al extraer las componentes principales de las ventanas de tiempo (espectrograma), corresponde a una extracción de la información de los píxeles de estas imágenes. (Izquierda) ventana de 28×28 píxeles correspondiente a ruido del detector, (Derecha) ventana de 28×28 píxeles correspondiente a la señal sintética inyectada.

3. Clasificación

Debido a que el objetivo de nuestro trabajo es diseñar un clasificador el cual nos muestre si existe una señal gravitacional dentro del ruido de los detectores, es imperante utilizar un algoritmo de Machine Learning en este caso, uno de aprendizaje supervisado el cual consta de una clasificación binaria: Existe o no la presencia de una onda gravitacional.

Por lo tanto, se opta por un algoritmo, el cual sea de fácil implementación y que no necesite una gran cantidad de hiperparámetros, es natural la selección del clasificador por K-Vecinos-más Cercanos (KNN).

3.1. KNN: K vecinos más cercanos

El algoritmo de k vecinos más cercanos, también conocido como KNN o $k-NN$, es un clasificador de aprendizaje supervisado no paramétrico, que utiliza la proximidad para hacer clasificaciones o predicciones sobre la agrupación de un punto de datos individual.

Para los problemas de clasificación, se asigna una etiqueta de clase sobre la base de un voto mayoritario, es decir, se utiliza la etiqueta que se representa con más frecuencia alrededor de un punto de datos determinado. Si bien esto técnicamente se considera “voto por mayoría”, el término “voto por mayoría”se usa más comúnmente en la literatura.

El objetivo del algoritmo del vecino más cercano es identificar los vecinos más cercanos de un punto de consulta dado, de modo que podamos asignar una etiqueta de clase a ese punto.

Tabla 1. Métricas del modelo de aprendizaje.

Métrica	Valor
Accuracy	0.970
Precisión	0.993
Recuperación	0.911

Para determinar qué puntos de datos están más cerca de un punto de consulta determinado, será necesario calcular la distancia entre el punto de consulta y los otros puntos de datos. Estas métricas de distancia ayudan a formar límites de decisión, que dividen los puntos de consulta en diferentes regiones [8].

Para este caso se utiliza la distancia euclidiana:

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(y_i - x_i)^2}. \tag{3}$$

En donde la distancia entre los puntos x_i y y_i está denotada por $d(x, y)$.

Tomando esto en cuenta, primero se divide el set de datos en dos partes, entrenamiento y prueba, los cuales con $70\,\%$ y $30\,\%$ correspondiente del set de datos. Después se complementa con un set de etiquetas las cuales corresponden a cada una de las señales y nos indican la presencia de la OG y se dividen de misma manera.

Ya con el set de datos de entrenamiento comenzamos con el desarrollo del modelo, el cual utilizando de IDE: Python = 3.8.5 y las paqueterías especializadas de Scikit-learn procedemos a entrenar nuestro modelo con los datos de entrenamiento utilizando un hiperparámetro de vecinos $k = 3$. Con el modelo ya entrenado, se procede a efectuar una predicción con los datos de prueba.

4. Resultados

Las predicciones efectuadas por el modelo son evaluadas con el set de etiquetas de prueba, los resultados de las predicciones pueden ser observadas a través de una matriz de confusión la cual nos muestra las señales que fueron clasificas satisfactoriamente, así como las que no a través de una comparativa entre señales reales y las predichas, esto puede observarse en la Fig. 9.

Así mismo, para la evaluación del modelo de aprendizaje es necesario utilizar las métricas utilizadas para valorar el rendimiento de las predicciones que puede generar.

Las utilizadas, las podemos observar en la Tab. 1 en la que obtenemos los valores de Exactitud (Accuracy), el cual representa el porcentaje total de los valores correctamente clasificados, la precisión, la cual determina el porcentaje de valores clasificados como correctos, la recuperación (Recall), la cual nos muestra el porcentaje de los valores que han sido clasificados como verdaderos positivos. Para el caso de la clasificación de la clase: Ruido, se identificaron satisfactoriamente 783, de las 785 ventanas.

Es decir, se clasificaron correctamente el $99\,\%$ de las ventanas de ruido, mientras que $1\,\%$ de las ventanas de ruido resultan en una clasificación errónea.

Para el caso de la clasificación de la señal sintética de OG se identificaron correctamente 331, de 363 ventanas de señales de OG.

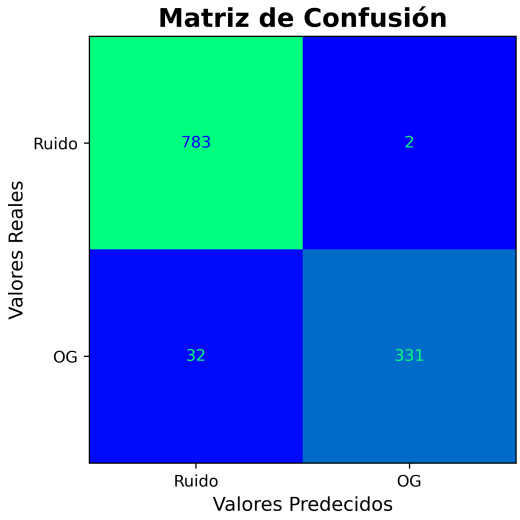


Fig. 9. Matriz de Confusión del modelo de aprendizaje. El rendimiento de clasificación contó con $97\,\%$ de predicciones y un $3\,\%$ a clasificaciones erróneas.

Es decir, se clasifican correctamente el $91\,\%$ de las ventanas de señal fenomenológica, mientras que él $9\,\%$ resulta en una clasificación errónea.

5. Conclusiones

Dentro de las clasificaciones efectuadas por el modelo de aprendizaje obtenidas mediante las métricas, se logra apreciar que las predicciones son satisfactorias, casi generando una clasificación perfecta, este fenómeno nos muestra que se cumple el objetivo del clasificador, lograr distinguir entre una OG y el ruido.

Es sumamente importante el notar que, aunque las clasificaciones suelen ser muy buenas, estas son atribuidas en partes porque el modelo de aprendizaje clasifica mejor el ruido que la señal gravitacional.

Una de las razones por las cuales se puede atribuir este comportamiento se puede observar la Fig. 7, los datos de la OG sintética, que son inyectados en el ruido, son totalmente distinguibles en el ruido, cosas que no ocurre en la naturaleza.

La razón por la cual la amplitud de estas señales sintéticas es debido a que el valor del SNR dado para cada una es bastante grande.

La realización de este detector nos muestra una alternativa para la detección de este tipo de señales, que a pesar de que no han sido detectadas, ya se pueden emplear estrategias para que las próximas detecciones sean exitosas.

En trabajos futuros se planteará la utilización de modelos de Deep learning como redes neuronales convolucionales para realizar un detector más preciso y que se adapte a las condiciones de los detectores de OG.

Referencias

1. La sensibilidad de los detectores LIGO avanzado en los albores de la astronomía de ondas gravitacionales (2023)
2. Abbott, B. P., Abbott, R., Abbott, T. D., Abernathy, M. R., Acernese, F., Ackley, K., Adams, C., Adams, T., Addesso, P., Adhikari, R. X., Adya, V. B., Affeldt, C., Agathos, M., Agatsuma, K., Aggarwal, N., Aguiar, O. D., Aiello, L., Ain, A., Ajith, P., Allen, B., Allocca, A., et al.: Observation of gravitational waves from a binary black hole merger. Physical Review Letters, vol. 116, no. 6, pp. 1–16 (2016) doi: 10.1103/PhysRevLett.116.061102
3. Andersen, O. E., Zha, S., da Silva Schneider, A., Betranhandy, A., Couch, S. M., O'Connor, E. P.: Equation-of-state dependence of gravitational waves in core-collapse supernovae. The Astrophysical Journal, vol. 923, no. 2, pp. 1–18 (2021) doi: 10.3847/1538-4357/ac294c
4. Andresen, H., Müller, B., Müller, E., Janka, H. T.: Gravitational wave signals from 3D neutrino hydrodynamics simulations of core-collapse supernovae. Monthly Notices of the Royal Astronomical Society, vol. 468, no. 2, pp. 2032–2051 (2017) doi: 10.1093/mnras/stx618
5. Astone, P., Cerdá-Durán, P., Di Palma, I., Drago, M., Muciaccia, F., Palomba, C., Ricci, F.: New method to observe gravitational waves emitted by core collapse supernovae. Physical Review D, vol. 98, no. 122002, pp. 1–11 (2018) doi: 10.1103/PhysRevD.98.122002
6. Einstein, A.: Kosmologische Betrachtungen zur allgemeinen Relativitäts-theorie (1922)
7. Greenspan, D.: Numerical solution of ordinary differential equations: For classical, relativistic and nano systems. John Wiley and Sons (2008)
8. IBM: Algoritmo de K vecinos más cercanos (2023) https://www.ibm.com/es-es/topics/knn
9. Kuroda, T., Kotake, K., Takiwaki, T.: A new gravitational-wave signature from standing accretion shock instability in supernovae. The Astrophysical Journal Letters, vol. 829, no. 1, pp. 1–6 (2016) doi: 10.3847/2041-8205/829/1/L14
10. Laser Interferometer Gravitational-Wave Observatory: About (2023) https://www.ligo.caltech.edu/page/about?highlight=LIGO\%27s+multi-kilometer-scale+gravitational+wave+detectors+use+laser+interferometry
11. Morales, M. D., Antelis, J. M., Moreno, C., Nesterov, A. I.: Deep learning for gravitational-wave data analysis: A resampling white-box approach. Sensors, vol. 21, no. 3174, pp. 1–38 (2021) doi: 10.3390/s21093174
12. Müller, B., Janka, H. T., Marek, A.: A new multi-dimensional general relativistic neutrino hydrodynamics code of core-collapse supernovae. III. Gravitational wave signals from supernova explosion models. The Astrophysical Journal, vol. 766, no. 43, pp. 1–21 (2013) doi: 10.1088/0004-637X/766/1/43
13. Murphy, J. W., Ott, C. D., Burrows, A.: A model for gravitational wave emission from neutrino-driven core-collapse supernovae. The Astrophysical Journal, vol. 707, no. 2, pp. 1173–1190 (2009) doi: 10.1088/0004-637X/707/2/1173
14. Neronov, A.: Introduction to multi-messenger astronomy. In: Journal of Physics: Conference Series, vol. 1263, pp. 1–44 (2019) doi: 10.1088/1742-6596/1263/1/012001
15. Reichert, M., Obergaulinger, M., Aloy, M. Á., Gabler, M., Arcones, A., Thielemann, F. K.: Magnetorotational supernovae: A nucleosynthetic analysis of sophisticated 3D models. Monthly Notices of the Royal Astronomical Society, vol. 518, no. 1, pp. 1557–1583 (2023) doi: 10.1093/mnras/stac3185
16. Yakunin, K. N., Mezzacappa, A., Marronetti, P., Yoshida, S., Bruenn, S. W., Hix, W. R., Lentz, E. J., Bronson Messer, O. E., Harris, J. A., Endeve, E., Blondin, J. M., Lingerfelt, E. J.: Gravitational wave signatures of ab initio two-dimensional core collapse supernova explosion models for 12-25 m stars. Physical Review D, vol. 92, no. 084040, pp. 1–13 (2015) doi: 10.1103/PhysRevD.92.084040

ISSN 1870-4069

Detección de deterioros superficiales en pavimentos flexibles basada en segmentación semántica y redes transformer

Mario Alberto Roman-Garay[1], Luis Alberto Morales-Rosales[2], Héctor Rodríguez-Rangel[1], Sofía Isabel Fernández-Gregorio[3]

[1] Tecnólogico Nacional de México Campus Culiacán,
División de Posgrado,
México

[2] Universidad Michoacana de San Nicolás de Hidalgo,
Facultad de Ingeniería Civil,
México

[3] Universidad Veracruzana,
Facultad de Estadística e Informática,
Doctorado en Ciencias de la Computación,
México

{mario.rg, hector.rr}@culiacan.tecnm.mx,
lamorales@conacyt.mx, zS21000291@estudiantes.uv.mx

Resumen. La detección temprana y precisa de deterioros en carreteras es esencial para garantizar la seguridad vial y prevenir accidentes de tráfico. Además, la identificación temprana de agrietamientos y baches ayuda a reducir los costos de mantenimiento de las carreteras a largo plazo, ya que permite realizar reparaciones menores en lugar de costosas renovaciones de carreteras completas. El presente artículo se enfoca en la detección y segmentación automática de agrietamientos y baches en pavimentos por medio de redes Transformer en situaciones no controladas. Para abordar este problema, se utiliza una arquitectura de red llamada Segformer, que se encarga de la extracción de características de las imágenes. Además, se crea un conjunto de datos compuesto por 245 imágenes, en el cual se aplicaron técnicas de procesamiento digital de imágenes y aumento de datos para mejorar la precisión del modelo, por lo que el conjunto de imágenes se extendió a 2052. En el conjunto de datos se contemplaron distintos ambientes obtenidos en puntos situados en la ciudad de Culiacán, México, considerando cambios de iluminación y sombras, lo que permitió evaluar la robustez del modelo obtenido en condiciones similares a las de la vida real donde el ruido ambiental está presente. Nuestro modelo obtuvo métricas de *Precision* de 82.35 %, *F1 Score* de 88.89 % y *Recall* de 96.55 % en la detección de agrietamientos y baches en diferentes condiciones ambientales.

Palabras clave: Segmentación semántica, agrietamientos, baches, pavimento flexible, redes transformer.

Mario Alberto Roman-Garay, Luis Alberto Morales-Rosales, Héctor Rodríguez-Rangel, et al.

Detection of Surface Deterioration in Flexible Pavements based on Semantic Segmentation and Transformer Networks

Abstract. Early and accurate detection of road deterioration is essential to ensure road safety and prevent traffic accidents. In addition, early identification of cracks and potholes helps to reduce long-term road maintenance costs by allowing minor repairs instead of costly complete road renovations. This paper focuses on automatically detecting and segmenting pavement cracks and potholes by Transformer networks in uncontrolled situations. To address this problem, a network architecture called Segformer is used, responsible for feature extraction from the images. In addition, a dataset composed of 245 images was created, extending this dataset to 2052 images with digital image processing and data augmentation techniques to improve the model's accuracy. In the dataset, different environments obtained in points located in the city of Culiacan, Mexico, were contemplated, considering changes in illumination and shadows, allowing us to evaluate the robustness of the model obtained in conditions similar to those of real-life where environmental noise is present. Our model obtained a *Precision* of 82.35%, an *F1 Score* of 88.89%, and a *Recall* of 96.55% in detecting cracks and potholes in different environmental conditions.

Keywords: Semantic segmentation, cracking, potholes, flexible pavement, transformer networks

1. Introducción

Las carreteras son una infraestructura fundamental para el desarrollo económico y social de un país, así como para la movilidad terrestre. Sin embargo, muchas veces sufren deterioros que afectan su funcionamiento y seguridad. Entre los principales deterioros se encuentran los baches, agrietamientos o desprendimientos que incrementan el riesgo de accidentes y daños a vehículos. Para evitar estos daños, cada año se asignan recursos económicos por parte del gobierno para el programa de conservación de caminos. Una actividad importante dentro de la conservación es el monitoreo y evaluación del estado de las carreteras, denominada auscultamiento.

El auscultamiento o evaluación del estado de las carreteras es un conjunto de actividades que tienen como finalidad conservar los caminos en condiciones óptimas y seguras para los usuarios. Estas actividades consisten en obtener información de la calidad superficial mediante elementos tecnológicos como sensores, escáneres o cámaras digitales, instalados en vehículos o dispositivos que circulan por las carreteras.

Asimismo, se efectúan inspecciones visuales por parte de personal calificado para esta tarea. Con la información recopilada se diseñan planes de mantenimiento preventivo y correctivo según el grado de los deterioros identificados. Entre los deterioros que comprometen la seguridad y confort de los usuarios destacan los baches, agrietamientos o desprendimientos.

Aunado a ello se obtienen parámetros como la rugosidad para determinar la calidad del pavimento que permita garantizar la movilidad sin comprometer la seguridad vial. Es por ello, que la tarea de auscultamiento implica una gran inversión de tiempo y recursos humanos.

En los últimos años, se ha avanzado en la automatización de diversas tareas en el área de la ingeniería civil mediante el uso de la inteligencia artificial, y el auscultamiento no es una excepción. Existen numerosos trabajos que intentan realizar esta tarea mediante diferentes técnicas, destacando el empleo de la visión por computadora y el aprendizaje profundo [1].

La mayoría los trabajos se han enfocado en la detección y clasificación de deterioros superficiales como baches y agrietamientos utilizando cámaras digitales para obtener muestras [2]. No obstante, aún persiste un problema frecuente: el ruido ambiental que aparece en las imágenes, como sombras, diversidad de materiales en el pavimento o manchas sobre él.

Esto supone un inconveniente debido a que los algoritmos de aprendizaje profundo aprenden características como bordes, cambios de tonalidades en los píxeles o la textura de las superficies. Dado que las cámaras digitales captan la intensidad de la luz que reflejan las superficies, los cambios de tonalidad de elementos como sombras o manchas son semejantes a los cambios de tonalidad de la superficie cuando se presenta un deterioro como grieta o bache. Esto ocasiona que se presenten falsos positivos en los modelos de detección y clasificación al identificar deterioros cuando no existen.

Para evitar los problemas mencionados anteriormente, se han explorado diversos métodos basados en algoritmos de redes neuronales, tales como las redes convolucionales [3], las redes adversarias generativas (GAN) [4] y en los últimos años las redes transformer [5].

Aunado a la exploración de nuevas técnicas de detección y clasificación de deterioros, un desafío importante en la detección y clasificación de deterioros en carreteras es la falta de un conjunto de datos representativo que incluya distintos escenarios y condiciones de carreteras con y sin deterioros superficiales. Un conjunto de datos robusto hará que los modelos aprendan a reconocer patrones y características en una amplia variedad de escenarios.

El objetivo de un conjunto de datos robusto es contribuir a la mejora de la precisión y la capacidad de los modelos en la detección y clasificación de deterioros ante situaciones no previstas que se encuentran en campo. Además, se requiere distinguir entre el ruido ambiental y los deterioros. Por lo tanto, en este artículo se presenta una propuesta para la detección de grietas y baches en pavimentos flexibles por medio de segmentación semántica y la implementación de una arquitectura de redes transformer.

Se contribuye con un nuevo conjunto de datos que abarca 245 imágenes de deterioros. Este conjunto está compuesto por 130 imágenes de agrietamientos y 115 imágenes de baches donde se presentan distintos escenarios con presencia de sombras, cambios de iluminación, presencia de hojas u objetos varios.

Además, se genera un aumento de datos, obteniendo un total de 2052 imágenes, las cuales fueron preprocesadas mediante los filtros *Contrast Stretching* y conversión a escala de grises. La detección se efectúa empleando la red transformer SegFormer propuesta en [6].

Los resultados obtenidos muestran métricas de *precision* de 82.35 %, *F1 Score* de 88.89 % y *Recall* de 96.55 %. Estos resultados permitirán realizar de manera eficiente la detección de grietas y baches en pavimentos flexibles en ambientes no controlados, lo que demuestra la robustez del conjunto de datos de entrenamiento.

2. Estado del arte

Se han utilizado diferentes técnicas computacionales para abordar el problema de la detección automática de deterioros en pavimentos, como el procesamiento de imágenes por computadora y el aprendizaje automático. A continuación se presentan algunos trabajos que pertenecen a estas dos categorías y que ofrecen soluciones a diversas problemáticas en la detección automática de daños en carretera.

2.1. Procesamiento de imágenes

El procesamiento digital de imágenes ha sido pionero en la detección automática de deterioros en pavimentos. Uno de los primeros enfoques era inferir que los agrietamientos y baches presentaban una tonalidad más oscura con respecto al área sana del pavimento. Tomando esto en cuenta se utilizaron métodos de umbralización como Otsu [7] para segmentar las zonas más oscuras del pavimento y detectar agrietamientos o baches. Sin embargo, este enfoque presentaba detección de falsos positivos cuando existían cambios de tonalidad en el pavimento provocados por manchas, sombras u otros elementos ajenos al pavimento.

Otro enfoque aplicado a la detección de deterioros en pavimentos es el uso de métodos de detección de bordes como Canny o Sobel [8]. Con estos métodos se infiere que los bordes detectados pertenecen a las orillas de las grietas y los baches. Sin embargo, se presenta el mismo problema de falsos positivos en la presencia de entidades externas a la carretera.

En [9, 10] utilizan preprocesamiento de imágenes para mejorar la detección de grietas en pavimentos. El primero utiliza un método basado en la fusión de imágenes multiescala para reducir el ruido y mejorar la calidad de las imágenes antes de aplicar el algoritmo de detección de grietas. En el segundo se emplea un procedimiento de preprocesamiento para eliminar el ruido espurio y rectificar los datos originales del pavimento. Ambos realizan la segmentación semántica de grietas.

2.2. Aprendizaje profundo

Redes convolucionales

Las redes convolucionales han sido las más utilizadas para la detección automática de grietas en imágenes digitales. Estas han sido utilizadas para identificar y clasificar los daños en distintas categorías.

En [3, 11] los autores presentan su propio conjunto de datos con imágenes de distintos tipos de deterioros, como grietas transversales, longitudinales, piel de cocodrilo o baches. Estos conjuntos de datos son utilizados para entrenar redes convolucionales como *SqueezeNet* y *U-net*.

Se han combinado técnicas como el procesamiento de imágenes y las redes convolucionales. En [18] utiliza preprocesamiento para eliminar reflejos de los vidrios y extraer regiones del pavimento debido a la posición de la cámara. Además, utiliza capas de convoluciones para la extracción de características, utilizando un total de 527 imágenes para entrenamiento.

Redes transformer

En los últimos años, las redes transformer han experimentado un gran auge debido al excelente rendimiento que han demostrado en el procesamiento del lenguaje natural. Estas redes han sido adaptadas para el procesamiento de imágenes, así como para tareas de reconocimiento y clasificación de objetos. Arquitecturas como ViT (Vision Transformer) [12], DeiT (Data-efficient Image Transformer) [13] y DETR (Detection TRansformer) han logrado resultados competitivos o incluso superiores a los obtenidos por las redes convolucionales en tareas tanto de reconocimiento como en la clasificación de objetos.

Desde hace muchos años la detección automática de deterioros en pavimentos es un tema que se ha desarrollado y se ha adaptado a las nuevas tecnologías. Se ha visto beneficiada tanto con dispositivos nuevos para la recolección de información como con algoritmos computacionales que entrenan modelos de inteligencia artificial y aprendizaje profundo que identifican automáticamente grietas. Sin embargo, aún existen algunas problemáticas abiertas que deben ser solucionadas, tales como la detección de falsos positivos ante la presencia de ruido ambiental o la evaluación de distintos tipos de deterioros.

3. Materiales y métodos

En esta sección se describe el diseño de la toma de muestras de imágenes, de grietas y baches en pavimentos flexibles, y la implementación de una red Transformer (Segformer) entrenada para la segmentación semántica de grietas y baches. Para el proceso de adquisición de imágenes se empleó una cámara de acción compacta GoPro 8 Black. La elección de la GoPro 8 Black se debe a la alta calidad de imagen, así como a su capacidad para grabar en alta definición, lo que la hace ideal para capturar imágenes de superficies de pavimento de forma rápida y sencilla.

A continuación, se describe la instalación y el montaje de la cámara para la captura de imágenes. Además, se detalla la configuración del hardware utilizado, la especificación técnica de la cámara y la computadora empleada para el proceso de entrenamiento, así como el software utilizado para el procesamiento de las imágenes.

3.1. Hardware

El uso de distintos dispositivos es indispensable para llevar a cabo las tareas de auscultación con precisión. Para la toma de muestras se utilizó una cámara de acción GoPro 8 black con una resolución de 4000x3000 píxeles, la cual se colocó a 1.20 metros de altura con apoyo de un tripie. Además, se utilizó un equipo de cómputo con las características que se muestran en la Tabla 1.

Tabla 1. Características del servidor.

Nombre	Características
Sistema Operativo	Ubuntu 20.04
RAM	16 GB
Almacenamiento	500 GB SSD
Procesador	Ryzen 7 3700
Tarjeta Gráfica	RTX 3060 Ti

3.2. Metodología

La metodología que se ha utilizado en este trabajo se basa en los pasos genéricos para la obtención de modelos de aprendizaje profundo presentada por [14] y [2]. Sin embargo, se ha añadido una etapa adicional de pre-preprocesamiento para mejorar la calidad de los datos de entrada, así como una etapa de pruebas de los resultados.

La Figura 1 muestra el proceso de la metodología utilizada durante este trabajo. Comienza con la recolección de datos, donde se recopilan imágenes relevantes que muestren el objeto en diferentes contextos.

Luego sigue el preprocesamiento, donde se considera si se aplican o no distintos filtros para resaltar características como bordes, texturas y tonalidades de colores. También se busca reducir el ruido presente en los datos. Una vez definidas las imágenes, se construye el conjunto de datos necesario para entrenar el modelo clasificador.

Dependiendo de la arquitectura de redes neuronales utilizada se determina si es o no necesaria una etapa de etiquetado de imágenes. Tras el preprocesamiento y la construcción del conjunto de datos, se procede al entrenamiento de las redes neuronales y una validación de los modelos resultantes. Esta validación proporciona métricas para evaluar el rendimiento del modelo final, las cuales son *Mean IOU, Presicion, Recall y F1 Score*.

3.3. Recolección de datos

Con el propósito de obtener imágenes de deterioros superficiales en pavimentos, se realizaron recorridos por diversas vías de la ciudad de Culiacán, Sinaloa. Se enfatizó en el registro de pavimentos flexibles.

Para la captura de las imágenes se empleó una cámara de acción GoPro 8 black y un tripié con una inclinación de 20 grados respecto a la superficie del pavimento y a una altura de 1.20 metros, de tal manera que la imagen capturada mostrara el ancho total del carril, el cual tiene una anchura de tres metros en promedio.

3.4. Anotación de imágenes

El etiquetado o anotación de imágenes para el entrenamiento de redes neuronales para segmentación, es un proceso donde se indica el área que representa un objeto en una imagen [15]. Esta anotación se lleva a cabo a nivel de píxel, y existen herramientas con las cuales esto es posible.

En este trabajo se llevó a cabo la anotación con Roboflow [16], esta herramienta permite a los usuarios realizar anotaciones de distintos tipos, aplicar técnicas de

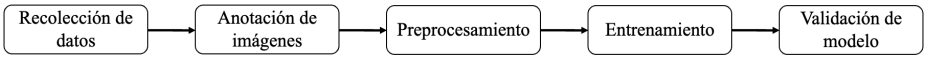


Fig. 1. Metodología implementada.

pre-procesamiento y hacer aumento de datos para crear distintas versiones del conjunto de datos.

La anotación utilizada fue la segmentación semántica, en la cual se debe señalar el contorno de los objetos de interés. En el caso de las grietas y baches se indicaron el contorno en todas las imágenes y se indicaba que el área pertenecía a alguno de los dos deterioros. De esta forma, al momento de descargar el conjunto de datos, la herramienta Roboflow nos proporciona la imagen original y una máscara, la cual es una imagen que contiene el área señalada durante la anotación.

En la segmentación semántica, la forma en que se diferencia un objeto de otro, está dado por el valor de los píxeles del área que se señala durante las anotaciones. Estos valores van desde 0 hasta 255, que son los posibles valores de un píxel en una imagen en escala de grises.

En la imagen mascara, los píxeles que representan el área de una grieta tienen un valor de uno y los que representan el área de un bache tienen el valor dos. Debido a que estos valores son muy cercanos a cero, el cual es la representación de un píxel totalmente negro, por lo que la mascara se aprecia como una imagen completamente oscura.

3.5. Pre-procesamiento

Actualmente, los modelos de aprendizaje profundo tienen la capacidad de ser entrenados sin necesidad de tener un preprocesamiento de datos; sin embargo, añadir esta etapa antes del entrenamiento mejora la calidad de los datos y aumenta la precisión del modelo [17].

Las imágenes crudas llegan a contener ruido como sombras o iluminación inconsistente que interfieren con la capacidad del modelo para identificar y segmentar los objetos de interés. Al aplicar técnicas de preprocesamiento como el estiramiento por contraste o el cambio a escala de grises, se mejora la calidad de las imágenes y facilita la identificación de los objetos de interés.

Estiramiento por contraste

El *Contrast Stretching* (estiramiento de contraste) es una técnica de procesamiento de imágenes que se utiliza para mejorar la calidad visual de las imágenes. Consiste en expandir el rango de valores de píxeles de una imagen de manera que los valores más bajos se mapeen a un valor mínimo y los valores más altos se mapeen a un valor máximo.

De esta manera, se amplía el rango de valores de la imagen para obtener una imagen con mayor contraste y detalle. El *Contrast Stretching* es una técnica común utilizada en el preprocesamiento de imágenes antes de aplicar algoritmos de segmentación o clasificación en tareas de procesamiento de imágenes. Con esta técnica se busca reducir la afección de las sombras y los cambios de iluminación en las imágenes.

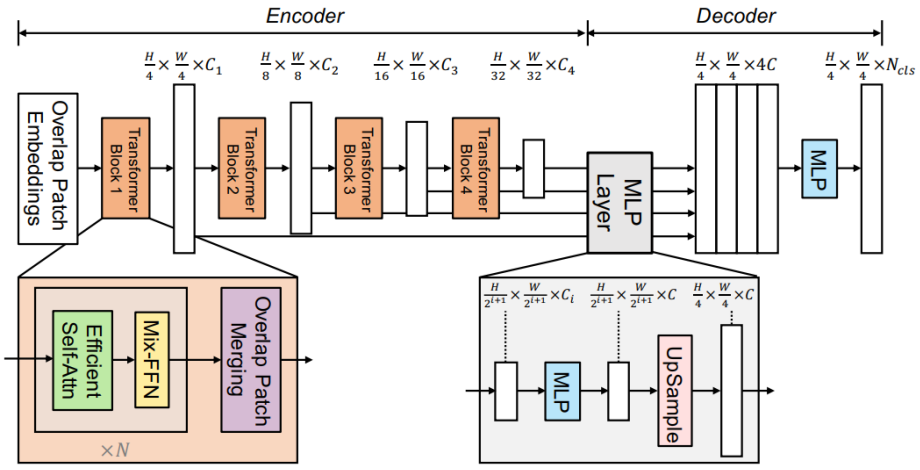


Fig. 2. Arquitectura de red Segformer [6].

Escala de grises

Es importante convertir las imágenes a escala de grises para mejorar la eficiencia del procesamiento de la información y reducir la cantidad de datos que se deben procesar en la red neuronal.

Al eliminar la información de color, se reduce el tamaño de la imagen y se elimina la redundancia de datos, lo que acelera el entrenamiento de la red y reduce la complejidad del modelo.

Redimensión de imágenes

El proceso de redimensión de imágenes cambia el tamaño y la resolución de las muestras para adaptarlas a un formato que permita ser introducidas a las redes neuronales. El tamaño original de las imágenes era de 4000x3000 píxeles, pero para introducirlas en las redes se redimensionaron a 512x512 píxeles.

Esta reducción se debe a que la red transformer (SegFormer) empleada tiene este requerimiento para el tamaño de imagen como dato de entrada, ya que cada píxel es introducido a una neurona como parte de la red.

Aumento de datos

Para aumentar la variedad y el número de muestras disponibles para el entrenamiento del modelo clasificador, se aplicaron técnicas de aumento de datos conocidas como *Tiling* y rotación.

– **Rotación:** Esta consiste en girar las imágenes originales un cierto ángulo alrededor de su centro. En este caso, se rotaron las imágenes 90 grados en sentido horario y se repitió el proceso hasta completar un giro completo de 270 grados. De esta manera, se obtuvieron tres imágenes rotadas por cada imagen original.

Tabla 2. Cantidad de imágenes obtenidas durante la toma de muestras.

Clase	Total
Agrietamiento	130
Baches	115
Total	**245**

– ***Tiling:*** consiste en dividir una imagen en varias partes más pequeñas y utilizar cada parte como una imagen independiente en el conjunto de datos. Esto se hace para aumentar la cantidad de datos de entrenamiento y evitar problemas de memoria cuando las imágenes son demasiado grandes para ser procesadas por la red neuronal en su totalidad.

3.6. Entrenamiento

En esta etapa, se utilizó la red SegFormer [6], la cual se basa en la arquitectura Transformer.

Esta red tiene dos características principales: el uso de un codificador para extraer características finas y gruesas de las imágenes, y un decodificador de perceptrones multicapa para fusionar estas características y predecir la máscara de segmentación. En la Figura 2 se muestra la arquitectura de la Red Segformer donde se observa gráficamente como están estructurados cada uno de los módulos que se utilizan, así como el codificador y el decodificador.

3.7. Validación de modelo

A continuación se muestran las métricas que se utilizan para llevar a cabo la evaluación del modelo de segmentación.

Mean Intersection Over Union

El mIoU (Intersección sobre la Unión Promedio) es una medida de la superposición entre la máscara de segmentación (también conocida como máscara de verdad terrenal) y la máscara de predicción producida por el modelo de segmentación. Se calcula como la relación entre el área de la intersección entre estas dos máscaras y el área de la unión entre ellas, para cada clase de objeto. Luego, se calcula el promedio de las mIoU de todas las clases para obtener una medida general del rendimiento del modelo. Para el cálculo de mIoU se consideran la ecuaciones 1 y 2 descritas a continuación:

$$IoU_c = \frac{TP_c}{TP_c + FP_c + FN_c}, \tag{1}$$

$$meanIoU = \frac{1}{c}\sum_{c} IoU_c. \tag{2}$$

TP= *Verdaderos positivos*, FP=*Falsos positivos*, FN= *Falsos negativos*

Precision, Recall, F1 Score

Las métricas *Precision, Recall y F1 Score* se obtuvieron al comparar la información real de las anotaciones realizadas con las predicciones generadas por el modelo entrenado.

La métrica *presicion* o precisión, en español, se define como la proporción de verdaderos positivos (TP, por sus siglas en inglés), es decir, los casos positivos correctamente clasificados, sobre el total de casos clasificados como positivos. En otras palabras, la precisión mide la capacidad del modelo para identificar correctamente los casos positivos. La Ecuación 3 muestra la fórmula de la precisión:

$$Precision = \frac{TP}{TP + FP}. \tag{3}$$

La recuperación, también conocida como *recall*, se define como la proporción de verdaderos positivos sobre el total de casos positivos (TP + Falsos negativos). En otras palabras, la recuperación mide la capacidad del modelo para identificar todos los casos positivos. La Ecuación 4 muestra la fórmula de la Recuperación (Recall):

$$Recall = \frac{TP}{TP + FN}. \tag{4}$$

La medida *F1 Score* es una combinación de precisión y recuperación, y se utiliza como una medida general de rendimiento del modelo. Se calcula como el promedio armónico de precisión y recuperación. La Ecuación 5 muestra la fórmula de la medida *F1 Score*:

$$F1Score = \frac{2 \times Precision \times Recall}{Precision + Recall}. \tag{5}$$

4. Resultados

Durante la etapa de recolección de datos se llevaron a cabo distintos recorridos para obtener muestras de deterioros superficiales en pavimentos. Se recorrieron las calles Cerro Monte Largo, y el bulevar Luciernilla, además del circuito interior del Tecnológico Nacional de México Campus Culiacán, todo esto en la ciudad de Culiacán, Sinaloa. A continuación se muestra el recorrido total que se realizó en las calles mencionadas:

- Calle Juan de Dios Batiz con un total de 832 metros,
- Calle Cerro Monte Largo con un total de 903 metros,
- Calle Luciernilla con un total de 1650 metros.

En la Tabla 2 se muestran los resultados de las imágenes que se obtuvieron durante los recorridos realizados. En ella se observa que en total se lograron recolectar 245 imágenes.

Posteriormente, se llevó a cabo el proceso de anotación de imágenes. La Figura 3 muestra un conjunto de imágenes normales de pavimentos junto a sus respectivas máscaras de segmentación. En las máscaras se han delimitado con precisión las áreas

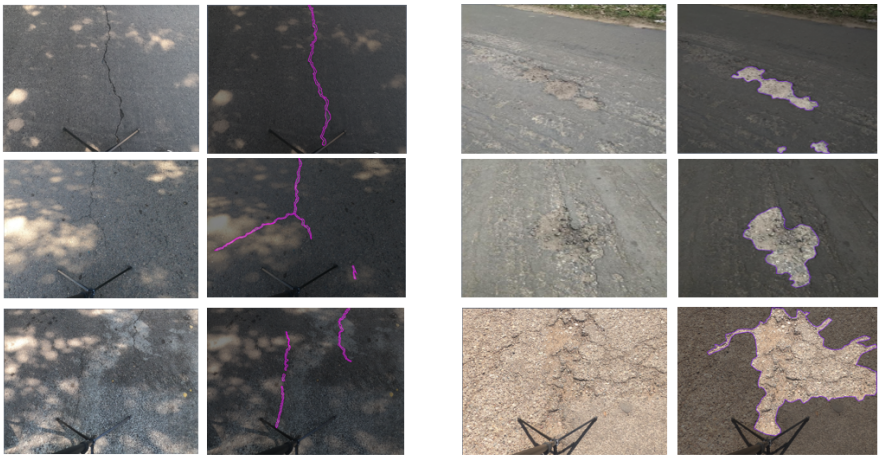


Fig. 3. a) Imagen normal. b) Imagen con anotación a mano.

que presentan agrietamientos y baches, utilizando técnicas manuales de segmentación. Estas máscaras de segmentación sirven de referencia durante el entrenamiento para indicar el área de interés que se requiere analizar.

Los ejemplos presentados son solo una muestra de las 245 imágenes que se etiquetaron manualmente durante el proceso de anotación. Es importante mencionar que este proceso es el que consume más tiempo. La anotación de cada imagen toma alrededor de 10 a 15 minutos, dependiendo de la complejidad de los deterioros. Por lo tanto, el tiempo total necesario para completar estas anotaciones fue de aproximadamente 60 horas.

Resultados de preprocesamiento

Después de realizar las anotaciones en las imágenes, se aplican técnicas de preprocesamiento para resaltar ciertas características, como los bordes, o para reducir el impacto de la iluminación o las sombras en las imágenes.

En este caso, además de las técnicas mencionadas anteriormente, se aplicaron técnicas como *Contrast Stretching* y conversión a escala de grises. También se realizaron cambios en las dimensiones de las imágenes.

Una vez aplicadas las técnicas de Tiling y rotación de imágenes, se produjo un aumento significativo en la cantidad de imágenes. De las 245 imágenes originales, se obtuvo un conjunto de datos de, 2940 imágenes. Sin embargo, debido a una restricción en la versión gratuita de Roboflow en cuanto a la cantidad de imágenes que se pueden descargar, el conjunto de datos final resultó en, 2052 imágenes.

Se realizó un recorte en la cantidad de imágenes totales en los conjuntos de validación y pruebas para lograr esto. La distribución de imágenes en los conjuntos de entrenamiento, validación y pruebas fue realizada de la siguiente manera 1780, 184 y 88 respectivamente.

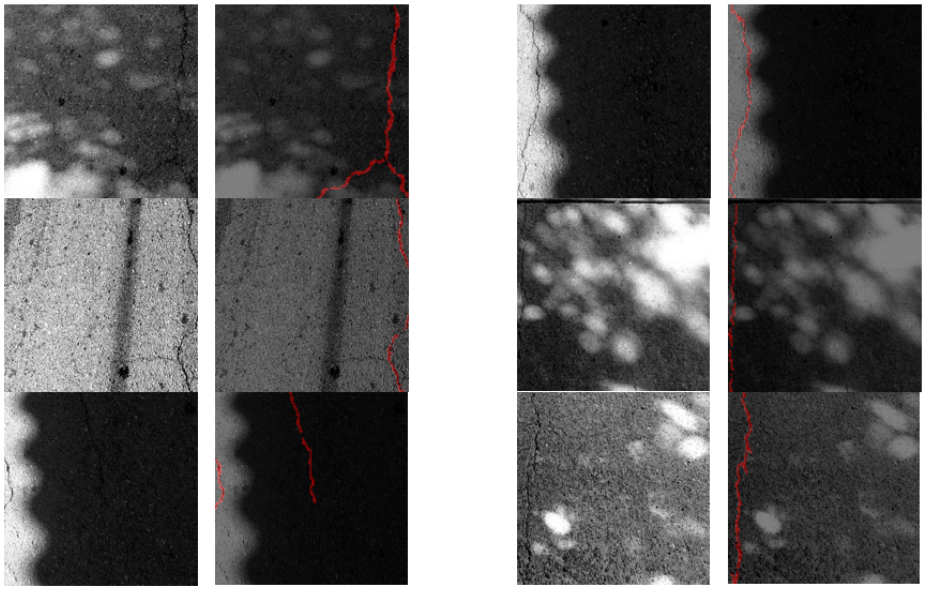


Fig. 4. a) Imagen normal. b) Mascara de predicción.

Resultados del entrenamiento del modelo de segmentación

En la Figura 4 se muestran algunos resultados obtenidos con el conjunto de pruebas una vez realizado el entrenamiento del modelo de predicción. Se observa que esté es capaz de identificar correctamente los agrietamientos que aparecen en el pavimento, incluso con la presencia de cambios de iluminación y sombras.

En la Tabla 3 se muestran los resultados de las métricas que se utilizaron para evaluar el entrenamiento del modelo de segmentación, así como una comparación con trabajos de otros autores.

Los resultados mostrados de nuestra solución en la Tabla 3 son las métricas obtenidas en el conjunto de pruebas. En la Tabla 3 los valores de nuestras métricas de evaluación son superiores al 80 %, excepto Mean IoU. Se obtuvo un *Recall* del 96.55 %, lo que indica que el modelo detecta correctamente la mayoría de las grietas y baches con una cantidad pequeña de falsos positivos, superando a los otros modelos presentados en la tabla comparativa.

En general, los resultados obtenidos son prometedores y sugieren que el modelo de detección de objetos es eficaz para la tarea que fue entrenado en presencia de ruido ambiental, cambios de iluminación y sombras, como se observa en la Figura 4.

No obstante, es importante tener en cuenta que siempre hay margen de mejora en cualquier modelo de aprendizaje automático y se pueden explorar técnicas adicionales para mejorar aún más la precisión y el rendimiento del modelo en futuros trabajos.

Tabla 3. Comparativa de las métricas de evaluación con el estado del arte.

Autor	Técnica	Imágenes	*Mean IOU*	*Presicion*	*Recall*	*F1 Score*
Li et al. [9]	Procesamiento de imágenes	No requiere Entrenamiento	No indica	88.38 %	93.15 %	90.68 %
Li et al. [10]	Procesamiento de imágenes	200 de prueba	No indica	89.90 %	89.47 %	88.04 %
Bang et al.[14]	Redes Convolucionales	527	No indica	93.57 %	84.90 %	89.03 %
Solución propuesta	Redes transformer	2940	67.00 %	82.35 %	96.55 %	88.89 %

5. Conclusiones

En este trabajo se ha presentado una propuesta para la detección y segmentación semántica de deterioros superficiales en pavimentos mediante el uso de técnicas de visión artificial y aprendizaje profundo. Además, se presentó la construcción de un conjunto de datos de deterioros superficiales de 245 imágenes, a las cuales se les aplicaron técnicas de aumento de datos de Tiling y rotación, obteniendo un total de 2052 imágenes.

A pesar de no contar con una gran cantidad de imágenes, se logró llevar a cabo la segmentación semántica que permite identificar la presencia de baches y grietas, incluso en presencia de ruido como sombras, cambios de iluminación, manchas en el pavimento y objetos ajenos al pavimento.

Esto demuestra que el modelo entrenado tiene la capacidad de detectar deterioros en ambientes no controlados y diferentes condiciones. Los factores que contribuyeron a ello fue la inclusión de técnicas de aumento de datos, preprocesamiento, la inclusión de datos con distinto ruido externo y la arquitectura de la red Segformer.

Es importante considerar la calidad de la segmentación semántica que se lleva a cabo manualmente al construir el conjunto de datos. Esta calidad es crucial para el desempeño del modelo, ya que las delimitaciones precisas del área de interés permiten que la red neuronal se enfoque en las características que se desean identificar.

En trabajos futuros, se aumentará la cantidad de imágenes anotadas para llevar a cabo una clasificación de los diferentes tipos de agrietamientos y baches. Además, se obtendrán medidas de longitud, anchura y profundidad para evaluar la severidad del daño que representa cada deterioro para la carretera, con el fin de desarrollar un algoritmo que permita la evaluación de los deterioros de manera automática.

Referencias

1. Du, Z., Yuan, J., Xiao, F., Hettiarachchi, C.: Application of image technology on pavement distress detection: A review. Measurement, vol. 184, no. 109900 (2021) doi: 10.1016/j.measurement.2021.109900
2. Hou, Y., Li, Q., Zhang, C., Lu, G., Ye, Z., Chen, Y., Wang, L., Cao, D.: The state-of-the-art review on applications of intrusive sensing, image processing techniques, and machine learning methods in pavement monitoring and analysis. Engineering, vol. 7, no. 6, pp. 845–856 (2021) doi: 10.1016/j.eng.2020.07.030

3. Ha, J., Kim, D., Kim, M.: Assessing severity of road cracks using deep learning-based segmentation and detection. The Journal of Supercomputing, vol. 78, pp. 17721–17735 (2022) doi: 10.1007/s11227-022-04560-x
4. Zhang, K., Zhang, Y., Cheng, H. D.: CrackGAN: Pavement crack detection using partially accurate ground truths based on generative adversarial learning. In: IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 2, pp. 1306–1319 (2021) doi: 10.1109/TITS.2020.2990703
5. Xiao, S., Shang, K., Lin, K., Wu, Q., Gu, H., Zhang, Z.: Pavement crack detection with hybrid-window attentive vision transformers. International Journal of Applied Earth Observation and Geoinformation, vol. 116 (2023) doi: 10.1016/j.jag.2022.103172
6. Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., Luo, P.: SegFormer: Simple and efficient design for semantic segmentation with transformers. Advances in Neural Information Processing Systems, vol. 34, pp. 12077–12090 (2021)
7. Ha, J., Kim, D., Kim, M.: Assessing severity of road cracks using deep learning-based segmentation and detection. The Journal of Supercomputing, vol. 78, pp. 17721-–17735 (2022) doi: 10.1007/s11227-022-04560-x
8. Ahmed-Talab, A. M., Huang, Z., Xi, F., HaiMing, L.: Detection crack in image using Otsu method and multiple filtering in image processing techniques. Optik, vol. 127, no. 3, pp. 1030–1033 (2016) doi: 10.1016/j.ijleo.2015.09.147
9. Li, H., Song, D., Liu, Y., Li, B.: Automatic pavement crack detection by multi-scale image fusion. IEEE Transactions on Intelligent Transportation Systems, vol. 20, no. 6, pp. 2025–2036 (2019) doi: 10.1109/TITS.2018.2856928
10. Li, B., Wang, K. C., Zhang, A., Fei, Y., Sollazzo, G.: Automatic segmentation and enhancement of pavement cracks based on 3D pavement images. Journal of Advanced Transportation, vol. 2019 (2019) doi: 10.1155/2019/1813763
11. Eisenbach, M., Stricker, R., Seichter, D., Amende, K., Debes, K., Sesselmann, M., Ebersbach, D., Stoeckert, U., Gross, H. M.: How to get pavement distress detection ready for deep learning? A systematic approach. In: 2017 International Joint Conference on Neural Networks (IJCNN), pp. 2039–2047 (2017) doi: 10.1109/IJCNN.2017.7966101
12. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N.: An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations (2021) doi: 10.48550/arXiv.2010.11929
13. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jégou, H.: Training data-efficient image transformers and distillation through attention. In: International conference on machine learning, vol. 139, pp. 10347–10357 (2021)
14. Hamishebahar, Y., Guan, H., So, S., Jo, J.: A comprehensive review of deep learning-based crack detection approaches. Applied Sciences, vol. 12, no. 3 (2022) doi: 10.3390/app12031374
15. Guo, Y., Liu, Y., Georgiou, T., Lew, M. S.: A review of semantic segmentation using deep neural networks. International Journal of Multimedia Information Retrieval, vol. 7, pp. 87–93 (2018) doi: 10.1007/s13735-017-0141-z
16. Roboflow: Roboflow universe: Open source computer vision community (2023) https://universe.roboflow.com/
17. Ranganathan, G.: A study to find facts behind preprocessing on deep learning algorithms. Journal of Innovative Image Processing (JIIP), vol. 3, no. 1, pp. 66–74 (2021) doi: 10.36548/jiip.2021.1.006
18. Bang, S., Park, S., Kim, H., Kim, H.: Encoder–decoder network for pixel-level road crack detection in black-box images. Computer-Aided Civil and Infrastructure Engineering, vol. 34, no. 8, pp. 713–727 (2019) doi: 10.1111/mice.12440

ISSN 1870-4069

Inducción de árboles de decisión oblicuos utilizando dos variantes de evolución diferencial adaptiva

Miguel Angel Morales-Hernández[1], Rafael Rivera-López[2], Efrén Mezura-Montes[3]

[1] Laboratorio Nacional de Informática Avanzada, México

[2] Tecnológico Nacional de México, Instituto Tecnológico de Veracruz, México

[3] Instituto de Investigaciones en Inteligencia Artificial, Universidad Veracruzana, México

mangelmhdez@gmail.com, rafael.rl@veracruz.tecnm.mx, emezura@uv.mx

Resumen. Este artículo presenta un análisis comparativo del comportamiento de tres variantes del algoritmo de Evolución Diferencial para inducir árboles de decisión oblicuos. Tomando como base los resultados de una versión clásica del algoritmo, se implementaron dos de sus variantes adaptivas: JADE y SHADE, con el ánimo de observar su comportamiento de búsqueda en un espacio no continuo como es el de los árboles de decisión. Los resultados obtenidos sobre un conjunto de datos de prueba, indica que los algoritmos tienen un comportamiento similar, pero SHADE se comporta mejor construyendo árboles oblicuos para datasets con muchas etiquetas de clase.

Palabras clave: IA, aprendizaje automático, evolución diferencial, árboles de decisión, árboles de decisión oblicuos.

Oblique Decision Trees Induction Using Two Adaptive Differential Evolution Algorithms

Abstract. This article presents a comparative analysis of the behavior of three variants of the Differential Evolution algorithm for inducing oblique decision trees. Based on the results of a classic version of Differential Evolution, two self-adaptive algorithms were implemented: JADE and SHADE, with the aim of observing the behavior of these algorithms in non-continuos search spaces such as the space of decision trees. The results obtained on a test dataset indicate that the

algorithms have a similar behavior, but SHADE performs better in constructing oblique trees for datasets with many class labels.

Keywords: AI, machine learning, diferential evolution, decision trees, oblique decision trees.

1. Introducción

La Inteligencia Artificial (IA) se enfoca en el desarrollo de sistemas que pueden realizar tareas que requieren inteligencia humana, como el reconocimiento de voz, el procesamiento del lenguaje natural, la toma de decisiones, así como la resolución de problemas complejos [7]. La IA trata de emular las capacidades humanas, siendo el aprendizaje un tema de creciente interés.

El Aprendizaje Automático es la rama de la IA centrada en el desarrollo de algoritmos y modelos matemáticos que pueden aprender de los datos, sin ser programados explícitamente. Esta área cuenta con diferentes modelos de aprendizaje, destacándose aquellos de aprendizaje supervisado, que parten de un conjunto establecido de datos etiquetados (datasets) para generar un modelo que permita encontrar patrones y realizar tareas de clasificación o regresión [13].

En este contexto se puede decir que el desarrollo de análisis de datos es un tema importante porque con la información recolectada se pueden tomar mejores decisiones, así como obtener resultados más precisos, particularmente con la implementación y uso de diferentes técnicas de la minería de datos [17], con el objetivo de descubrir patrones en ellos.

1.1. Árboles de decisión

Los árboles de decisión (ADs) son modelos generados a partir de datos que cuentan con una estructura jerárquica similar a la de un árbol biológico, compuesto por ramas y hojas, entre otras características [6].

Ellos particionan un dataset "*de arriba hacia abajo*"distribuyendo los datos en los nodos internos del árbol, iniciando con el nodo raíz y ramificándose hacia los nodos hoja, dividiendo así el espacio de datos usando hiperplanos. En los ADs más populares, conocidos como paralelos a los ejes, cada nodo interno incluye la evaluación de un solo atributo del dataset.

Un aspecto importante al generar ADs es el inducir modelos compactos, ya que permiten tener una mejor interpretación y precisión con los resultados.

Los algoritmos tradicionales para inducir ADs como C4.5 [11] y CART [2] pueden producir árboles con muchas particiones, lo que los hace difíciles de interpretar, por lo que otros tipos de particiones se han estudiado, como aquellos donde se utiliza una combinación lineal de varios atributos en la condición evaluada por un nodo interno.

Estas combinaciones de atributos producen particiones con hiperplanos oblicuos, definidos como sigue:

$$\sum_{i=1}^{d} w_i x_i \leq \theta, \tag{1}$$

donde w_i es un coeficiente numérico aplicado al i-ésimo atributo x_i de un dataset con d atributos, y θ es el término independiente del hiperplano. Los árboles creados usando este enfoque, conocidos como árboles oblicuos, generalmente son más compactos y mas precisos que los tradicionales.

La inducción de este tipo de ADs suele hacerse con buscadores voraces, como el algoritmo CART con combinaciones lineales, y los métodos Linear Machine Decision Trees (LMDT) y Simulated Annealing of Decision Trees (SADT) [10].

1.2. Evolución diferencial

Evolución Diferencial (ED) es uno de los algoritmos más recientes dentro del campo de la computación evolutiva, que ha demostrado ser altamente competitivo para resolver problemas de búsqueda complejos [4], distinguiéndose por la simplicidad en su implementación y su habilidad para encontrar soluciones cercanas al óptimo.

Los elementos de ED provienen del esquema clásico de un algoritmo poblacional: inicialización, mutación, recombinación y selección. Todo empieza con un conjunto inicial de soluciones candidatas, también llamadas *individuos* o vectores, que se generan de forma aleatoria con una distribución uniforme.

Posteriormente se aplica una mutación diferencial y una cruza a cada solución (padre) para generar un nuevo individuo (descendiente), el cual permanecerá para la siguiente generación (iteración) si es mejor que su padre.

Todo lo anterior se repite hasta alcanzar una condición de paro, asociada usualmente a un número máximo de generaciones. La selección de los individuos se basa en la comparación de su aptitud, representada por la función objetivo del problema.

La variante más popular de ED es conocida como DE/rand/1 [15] , cuyo operador de mutación es el siguiente:

$$\mathbf{v}_{i,g} = \mathbf{x}_{r_0,g} + F \cdot (\mathbf{x}_{r_1,g} - \mathbf{x}_{r_2,g}), \tag{2}$$

donde g es la generación actual, $\mathbf{x}_{r_0,g}$, $\mathbf{x}_{r_1,g}$ y $\mathbf{x}_{r_2,g}$ son tres soluciones diferentes entre sí y diferentes al padre, escogidas al azar de la población actual, y $\mathbf{v}_i$ es el vector o solución mutante. $\mathbf{v}_i$ se recombina con la solución padre mediante una cruza discreta que puede ser binomial (bin) o exponencial (exp), para generar la solución descendiente.

Esta nueva solución compite contra su padre y el mejor de ellos, con base en aptitud, sobrevive para la siguiente generación y el otro es eliminado. ED usa tres parámetros: NP que indica el tamaño de la población, F que representa un factor de escala para el vector mutante, y CR, que se utiliza en la recombinación.

Existen otras variantes de ED, como la DE/best/1, que usa el siguiente operador de mutación:

$$\mathbf{v}_{i,g} = \mathbf{x}_{best,g} + F \cdot (\mathbf{x}_{r_1,g} - \mathbf{x}_{r_2,g}), \tag{3}$$

donde la diferencia es que el primer vector usado en la combinación, $\mathbf{x}_{best,g}$, es la mejor solución de la población actual.

1.3. Evolución diferencial para inducir árboles oblicuos

El utilizar algoritmos de búsqueda poblacional como los algoritmos evolutivos es de gran interés, pues la expectativa es que se pueden encontrar modelos (árboles en este caso) de alta calidad gracias a una búsqueda global en contraste con una búsqueda de trayectoria como la de los algoritmos tradicionales [12].

En la actualidad las versiones adaptivas de ED (aquellas que incluyen un autoajuste de los parámetros del algoritmo) han arrojado mejores resultados comparándolos con otros algoritmos evolutivos para la resolución de problemas de optimización numérica [3], ya que pueden adaptar automáticamente las estrategias de aprendizaje y la configuración de parámetros durante la evolución.

En la revisión de la literatura no se ha encontrado un estudio que haya probado la inducción de árboles de decisión oblicuos en un espacio de búsqueda continuo como el de los algoritmos de evolución diferencial adaptiva. Dado a lo anterior también es de suma importancia estudiar el comportamiento de los algoritmos con espacios de búsqueda que no son de naturaleza continua, tal es el caso de los AD oblicuos.

En este contexto se propone trabajar con los árboles de decisión oblicuos, ya que estos suelen mostrar un mejor desempeño y son más compactos que los árboles paralelos a los ejes [14]. El generar árboles oblicuos mediante algoritmos evolutivos adaptivos, particularmente las variantes de ED que adaptan los valores de sus parámetros, representa una oportunidad de desarrollar resultados con mejor precisión e interpretación.

De acuerdo con lo anterior, se plantea inducir AD oblicuos mediante los algoritmos de evolución diferencial adaptivos, y comparar su desempeño contra la versión rand/1/bin. Se propone utilizar los algoritmos adaptivos JADE y SHADE, que han demostrado tener un desempeño destacado al resolver problemas de optimización numérica, comparando los resultados con base en la precisión de clasificación.

El resto del trabajo se organiza de la siguiente manera. La sección 2 presenta de forma detallada la descripción de los algoritmos, las técnicas tradicionales que inducen los AD oblicuos y su comportamiento. La sección 3 muestra los experimentos y resultados obtenidos de cada AD generado, así como las técnicas y algoritmos implementados y una discusión de los resultados obtenidos. Por último en la sección 4 se presentan las conclusiones de los resultados obtenidos y el trabajo futuro.

2. Propuesta

En esta investigación, para que ED pueda inducir árboles de decisión, los AD se representan usando vectores numéricos. En los siguientes párrafos se describe la estrategia de representación y los algoritmos usados en la experimentación.

2.1. Esquema de codificación

En la Fig. 1 se muestran las etapas para la transformación de un vector de parámetros numéricos a un árbol de decisión oblicuo [13]. Este esquema se propone en un trabajo previo donde se utilizó la versión DE/rand/1/bin para inducir árboles oblicuos [14].

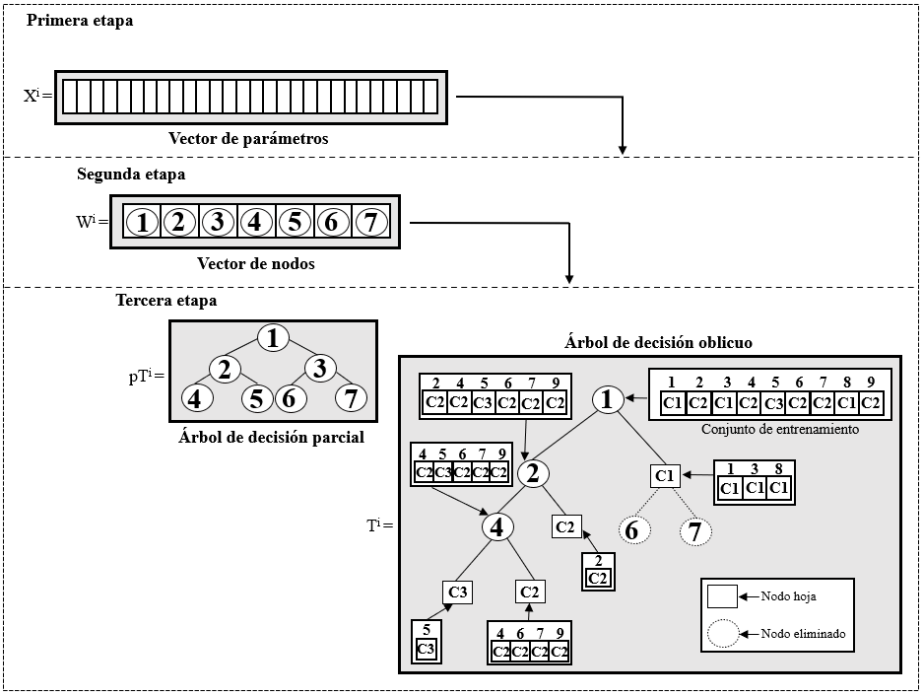


Fig. 1. Proceso de mapeo para la creación de un AD oblicuo.

Tamaño del individuo: La primera etapa consiste en determinar el tamaño del vector de parámetros, usado para crear los nodos internos del AD, mediante las fórmulas que estiman la profundidad de un árbol binario a partir del número de nodos internos y nodos hojas. En este caso se utiliza el número de atributos y el número de etiquetas de clase como un estimado de esos nodos:

$$H_i = \lceil log_2(d+1) \rceil, \tag{4}$$

$$H_l = \lceil log_2(s) \rceil, \tag{5}$$

donde H_i y H_l representan la profundidad del árbol, d es el número de atributos y s es el número de etiquetas de clase.

Posteriormente se aplica la siguiente fórmula para obtener el número estimado de nodos internos (n_e):

$$n_e = 2^{max(H_i, H_l)-1} - 1. \tag{6}$$

Finalmente, el tamaño del vector de parámetros (n) que representa la secuencia de hiperplanos usados en los nodos internos del árbol se obtiene usando la siguiente fórmula:

$$n = n_e(d+1). \tag{7}$$

Vector de nodos internos: En la segunda etapa se crea el vector de nodos internos que se usarán en el árbol. Cada nodo representa un hiperplano usando $d + 1$ valores del vector de parámetros.

Construcción de árbol: Usando el vector de nodos, se construye un árbol binario con los hiperplanos construidos previamente. Este es un árbol parcial que representa solo nodos internos.

El paso final es usar el dataset para particionar sus datos usando los hiperplanos. Si al evaluar un hiperplano los datos son de una misma clase, este nodo se convierte en un nodo hoja, podando todos los nodos descendientes de dicho nodo. Esto nos permite crear árboles de decisión usando hiperplanos en sus nodos internos.

En este trabajo se propone experimentar con dos algoritmos que son versiones adaptivas de ED conocidas como SHADE y JADE. A continuación se detallan ambos.

2.2. Evolución diferencial adaptiva

JADE: El algoritmo de evolución diferencial adaptativa con archivo externo opcional, nombrado JADE por sus autores [18], implementa la estrategia de mutación *DE/current-to-pbest*, junto al auto-ajuste de los parámetros F y CR.

El parámetro F se modifica de acuerdo a una regla de actualización que utiliza la mejor combinación de parámetros de mutación en la población actual, así como de la iteración anterior, mientras que CR se actualiza de forma similar, utilizando la mejor combinación de parámetros de cruza de ambas poblaciones.

En general ambos valores se ajustan utilizando distribuciones de probabilidad independientes en cada iteración, obteniendo el promedio de los vectores mutados mejor adaptados. La estrategia de mutación implementada en este algoritmo es la base principal para el rendimiento y confiabilidad del mismo.

En cada generación, la probabilidad de cruzamiento CR_i de cada individuo $\mathbf{x}_i$ es generada usando una distribución de probabilidad normal con media μ_{CR} y una desviación estándar de 0.1, como sigue:

$$CR_i = randn_i(\mu_{CR}, 0{,}1). \tag{8}$$

El valor μ_{CR} es ajustado en cada generación de acuerdo al siguiente criterio:

$$\mu_{CR} = (1 - c) \cdot \mu_{CR} + c \cdot mean_A(S_{CR}), \tag{9}$$

donde c es una constante positiva y $mean_A$ es la media aritmética de los valores de los CR_i exitosos en la población, almacenados en S_{CR}.

De forma similar, el factor de mutación F_i de cada individuo $\mathbf{x}_i$ es generado usando una distribución de Cauchy con un parámetro μ_F y un parámetro de escala 0.1, como sigue:

$$F_i = randc_i(\mu_F, 0{,}1). \tag{10}$$

El valor μ_F es ajustado en cada generación de acuerdo al siguiente criterio:

$$\mu_F = (1 - c) \cdot \mu_{CR} + c \cdot mean_L(S_F), \tag{11}$$

donde $mean_L$ es la media de Lehmer de los F_i exitosos en la población, almacenados en S_F.

Adicionalmente al autoajuste de parámetros, JADE introduce un nuevo operador de mutación, denominado *DE/current-to-pbest*, basado en combinar la aleatoriedad de las soluciones candidatas usadas en la mutación, que permite una mejor exploración del espacio de búsqueda, con la posibilidad de utilizar los mejores individuos para guiar la búsqueda en zonas prometedoras. El operador se representa como sigue:

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F_i \cdot (\mathbf{x}^p_{best,g} - \mathbf{x}_{i,g}) + F_i \cdot (\mathbf{x}_{r1,g} - \tilde{\mathbf{x}}_{r2,g}), \tag{12}$$

donde $\mathbf{x}^p_{best,g}$ es seleccionado de entre un subconjunto de los mejores individuos de la población actual, $\mathbf{x}_{r1,g}$ es seleccionado aleatoriamente de la población actual, y $\tilde{\mathbf{x}}_{r2,g}$ es seleccionado de la unión de la población actual y un conjunto de soluciones desechadas previamente, almacenadas en un archivo externo.

La estructura de JADE se describe a continuación:

function $JADE$
  $\mu_{CR} = 0{,}5$; $\mu_F = 0{,}5$; $A = \emptyset$
  Creación de una población inicial aleatoria $\{\mathbf{x}_{i,0} | i = 1, 2, \ldots, NP\}$
  for $g \in \{1, \ldots, G\}$ **do**
    $S_F = \emptyset$; $S_{CR} = \emptyset$;
    for $i \in \{1, \ldots, NP\}$ **do**
      $CR_i = randn_i(\mu_{CR}, 0{,}1), F_i = randc_i(\mu_F, 0{,}1)$
      Selecciona al azar $\mathbf{x}^p_{best,g}$ entre el $100_p\,\%$ de mejores soluciones
      Selecciona al azar $\mathbf{x}_{rl,g} \neq \mathbf{x}_{i,g}$ de la población actual **P**
      Selecciona al azar $\tilde{\mathbf{x}}_{r2,g} \neq \mathbf{x}_{r1,g} \neq \mathbf{x}_{i,g}$ de $\mathbf{P} \cup \mathbf{A}$
      $\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F_i \cdot (\mathbf{x}^p_{best,g} - \mathbf{x}i, g) + F_i \cdot (\mathbf{x}_{r1,g} - \tilde{\mathbf{x}}_{r2,g})$
      $j_{rand} = randint(1, D)$
      for $j \in \{1, \ldots, D\}$ **do**
        if $j = j_{rand} \vee rand(0, 1) < CR_i$ **then**
          $\mathbf{u}_{j,i,g} = \mathbf{v}_{j,i,g}$
        else
          $\mathbf{u}_{j,i,g} = \mathbf{x}_{j,i,g}$
        end if
      end for
      if $f(\mathbf{x}_{i,g}) \leq f(\mathbf{u}_{i,g})$ **then**
        $\mathbf{x}_{i,g+1} = \mathbf{x}_{i,g}$
      else
        $\mathbf{x}_{i,g+1} = \mathbf{u}_{i,g}$; $\mathbf{x}_{i,g} \rightarrow \mathbf{A}$; $CR_i \rightarrow S_{CR}$; $F_i \rightarrow S_F$
      end if
    end for
    Elimina al azar soluciones en **A** tal que $|\mathbf{A}| \leq NP$
    $\mu_{CR} = (1 - c) \cdot \mu_{CR} + c \cdot mean_A(S_{CR})$
    $\mu_F = (1 - c) \cdot \mu_F + c \cdot mean_L(S_F)$
  end for
end function

SHADE: Este algoritmo surge de querer encontrar una mejora a la robustez del algoritmo de JADE, a través de una adaptación de sus parámetros basada en el historial de éxito [16].

SHADE contiene un conjunto de parámetros para guiar la actualizacion de CR y F, a medida que avanza la búsqueda, considerando la historia de cada generación llevada a cabo. Además, los vectores de prueba se generan para ser aplicados en la selección y la memoria histórica será actualizada. Lo anterior se repetirá hasta alcanzar algún criterio de terminación. SHADE se describe a continuación.

function $SHADE$
  $G = 0$;
  Creación de una población inicial aleatoria $\{\mathbf{x}_{i,0} \mid i = 1, 2, \ldots, NP\}$
  $M_{CR} = 0{,}5, M_F = 0{,}5$;
  A = $\emptyset$;
  $k = 1$;
  while No se alcance la condición de paro **do**
    $S_{CR} = \emptyset, S_F = \emptyset$;
    for $i \in \{1, \cdots, N\}$ **do**
      $r_i = randint(1, H)[1, H]$;
      $CR_{i,G} = randn_i(M_{CR,r_i}, 0{,}1)$;
      $F_{i,G} = randc_i(M_{F,r_i}, 0{,}1)$;
      $p_i, G = rand[p_{min}, 0{,}2]$;
      Crear $\mathbf{u}_{i,G}$ usando el operador *current-to-pbest/1/bin*;
    end for
    for $i \in \{1, \ldots, N\}$ **do**
      if $f(\mathbf{u}_{i,G}) \leq f(\mathbf{x}_{i,G})$ **then**
        $\mathbf{x}_{i,G+1} = \mathbf{u}_{i,G}$;
      else
        $\mathbf{x}_{i,G+1} = \mathbf{x}_{i,G}$;
      end if
      if $f(\mathbf{u}_{i,G}) < f(\mathbf{x}_{i,G})$ **then**
        $\mathbf{x}_{i,G} \rightarrow \mathbf{A}$;
        $CR_{i,G} \rightarrow S_{CR}, F_{i,G} \rightarrow S_F$;
      end if
    end for
    Si el tamaño de **A** excede $|\mathbf{A}|$,
    eliminar al azar individuos en **A** tal que $|\mathbf{A}| \leq |\mathbf{P}|$;
    if $S_{CR} \neq \emptyset \wedge S_F \neq \emptyset$ **then**
      Actualiza $M_{CR,k}, M_{F,k}$ usando S_{CR} y S_F;
      $k++$;
      if $k > H$ **then**
        $k = 1$;
      end if
    end if
  end while
end function

Tabla 1. Descripción de los datasets usados en el estudio experimental.

Dataset	Instancias	Atributos	Clases
glass	214	9	7
diabetes	768	8	2
australian	690	14	2
ionosphere	351	34	2
iris	150	4	3
balance-scale	625	4	3
ecoli	336	7	8
heart-startlog	270	13	2
liver-disorders	345	6	2
wine	178	13	3

SHADE depende del tamaño de memoria representado por H. Si este valor es pequeño, los valores tienen un uso frecuente, porque los valores más antiguos se sobrescriben rápidamente, favoreciendo una rápida convergencia de los valores de los parámetros de control. En caso contrario (valor grande), la tasa de control se espera hasta que la convergencia de parámetros disminuya, debido a que éstos seguirán teniendo influencia durante más tiempo.

3. Experimentos y resultados

3.1. Diseño experimental

Para poder evaluar si el uso de algoritmos adaptivos basados en DE ofrecen una ventaja sobre el trabajo existente, es necesario realizar un estudio experimental de su comportamiento. Los algoritmos a comparar en este estudio son DE/rand/1/bin, JADE y SHADE. Estos algoritmos fueron implementados en Java y utilizando la librería JMetal [5].

Primero se seleccionaron una serie de datasets, luego se definen los parámetros utilizados para ajustar las versiones de ED a comparar, y se determina la forma de usar los datos para construir los modelos de aprendizaje.

Datasets seleccionados. Para llevar a cabo el estudio experimental, se seleccionaron diez datasets del repositorio de aprendizaje supervisado (UCI) [9]. Debido a que las particiones que los nodos internos generan al evaluar el dataset representan una combinación lineal de valores de los atributos, todos los datasets utilizados tienen atributos numéricos solamente. La Tabla 1 describe las características de estos datasets.

Parámetros de los algoritmos. Tomando en cuenta lo descrito en la literatura, se han utilizado los siguientes parámetros para cada algoritmo.

- **rand/1/bin**: El factor de cruzamiento CR es 0.9, y el factor de escala es (F) es 0.9.
- **JADE:** La taza de cruzamiento promedio (μ_{CR}) es 0.5, el factor de escala promedio (μ_F) es 0.5, la tasa de mejores soluciones (p) es 0.05, y el valor del parámetro de balance (c) es 0.1.

Tabla 2. Desempeño de los algoritmos DE-ODT, JADE-ODT y SHADE-ODT en varios conjuntos de datos.

Dataset	DE-ODT	JADE-ODT	SHADE-ODT
glass	57.47 (3)	59.48 (2)	**61.02** (1)
diabetes	**74.21** (1)	73.54 (3)	74.14 (2)
australian	**79.71** (1)	75.23 (3)	76.71 (2)
ionosphere	90.79 (2)	89.08 (3)	**90.91** (1)
iris	**96.86** (1)	95.53 (3)	96.53 (2)
balance-scale	**90.41** (1)	90.12 (2)	90.09 (3)
ecoli	75.68 (3)	77.29 (2)	**77.52** (1)
heart-startlog	**83.18** (1)	70.33 (3)	78.25 (2)
liver-disorders	69.18 (3)	**70.00** (1)	69.65 (2)
wine	**89.15** (1)	72.35 (3)	85.73 (2)
Rank promedio	1.7	2.4	1.8

– **SHADE**: El tamaño de la memoria histórica (H) es 100, la proporción del tamaño del archivo externo (A) es 2.0 y la tasa de mejores soluciones (p) es 0.1.

El tamaño de población es de 100 individuos y el número máximo de evaluaciones es de $10,000 * n$, donde n es el tamaño del individuo.

Evaluación de los conjuntos de datos. El proceso evolutivo de cada algoritmo es guiado por la precisión de clasificación de los árboles de decisión como función de aptitud. Para obtener estimaciones fiables del rendimiento predictivo de los algoritmos implementados, se utilizó una validación cruzada (VC) de 10 folds [1]. VC es uno de los métodos de re-muestreo de datos más utilizados para estimar la predicción real [13].

En una validación cruzada de 10 folds, el conjunto de entrenamiento se divide aleatoriamente en diez folds disjuntos aproximadamente iguales. Para cada $k \in \{1, \ldots, 10\}$, se retiene el $k-$ésimo fold (el conjunto de prueba) y los folds restantes se utilizan para inducir un árbol de decisión. Una vez que se ha construido el árbol, el fold retenido se usa para calcular la precisión de la prueba.

Finalmente, cuando todos los folds se han usado en la fase de inducción, se calcula la precisión general de la prueba del modelo. Este esquema se repite diez veces, y al final se reporta la precisión de prueba promedio.

3.2. Resultados

La Tabla 2 muestra la precisión promedio de las diez ejecuciones para cada dataset y cada algoritmo. Los mejores resultados de cada dataset se resaltan en negritas y los números que se encuentran entre paréntesis, se refieren al ranking obtenido por el algoritmo al compararlo con los resultados de los otros algoritmos en un dataset particular.

3.3. Discusión

Como podemos observar en la Tabla 2, DE/rand/1/bin y SHADE obtienen mejores resultados por generar los promedios más bajos, *1.7* y *1.8* respectivamente, a

comparación de JADE que obtuvo *2.4*. En principio eso se esperaría al comparar JADE con SHADE, ya que es conocido que el comportamiento de SHADE en otros problemas es mejor al de JADE.

Sin embargo, se esperaría que SHADE superara en desempeño a las versiones tradicionales de DE, pero en estos resultados iniciales se encuentra lo contrario. Algo interesante que se observa es que SHADE obtiene mejores resultados para datasets con mayor número de clases (como en el caso de los datasets glass y ecoli), pero para dataset con dos clases, DE/rand/1/bin tiene mejor desempeño.

Se puede justificar este comportamiento debido a la forma de comparar los modelos generados, ya que es muy probable que SHADE obtenga mejor desempeño con los datos de entrenamiento, que se usan dentro del proceso evolutivo, y que al evaluar con los datos de prueba, su desempeño se degrade, lo que podría indicar que el modelo entrenado está sobreajustado.

En el caso de SHADE, se utiliza una memoria igual al tamaño de la población, y se observa que los parámetros varían lentamente, lo que sugiere evaluar el uso de una memoria de menor tamaño para acelerar la convergencia de las soluciones.

4. Conclusiones y trabajo futuro

Los resultados reportados en este trabajo son los primeros que se obtuvieron de una investigación en curso, y nos permiten reconocer que es importante seguir analizando el comportamiento de los algoritmos autoadaptables como JADE, SHADE y sus derivados, para obtener modelos de aprendizaje supervisado más precisos.

La métrica comparada inicialmente es la precisión de clasificación, para posteriormente analizar otras métricas como el tamaño del modelo y aquellas basadas en la matriz de confusión de los datos. Existen varios desafíos interesantes que se observan de estos resultados, como el poder generar árboles con menor número de nodos, y que el proceso evolutivo evite el sobre ajuste de los modelos.

También es interesante estudiar el comportamiento de otras variantes de los algoritmos autoadaptables conocidos en la literatura, tales como las versiones actualizadas de SHADE y aplicarlos a conjuntos de datos más robustos con otras características como su número de instancias y distribución de clases.

Agradecimientos. El primer autor agradece el apoyo de CONACyT mediante una beca para la realización de estudios de maestría en el Laboratorio Nacional de Informática Avanzada y la realización de la residencia profesional en el Instituto de Investigaciones en Inteligencia Artificial de la Universidad Veracruzana.

Referencias

1. Berrar, D.: Cross validation. Data Science Laboratory, Institute of Technology (2018) doi: 10.1016/B978-0-12-809633-8.20349-X
2. Breiman, L., Friedman, J. H., Olshen, R., Stone, C. J.: Classification and regression trees. Routledge (1984) doi: 10.1201/9781315139470

3. Brest, J., Bošković, B., Greiner, S., Žumer, V., Maučec, M.: Performance comparison of self-adaptive and adaptive differential evolution algorithms. Soft Computing, vol. 11, pp. 617–629 (2007) doi: 10.1007/s00500-006-0124-0
4. Coello, C.: Computación evolutiva. Academia Mexicana de Computación, Amexcomp (2019)
5. Durillo, J. J., Nebro, A. J.: jMetal: A Java framework for multi-objective optimization. Advances in Engineering Software, vol. 42, no. 10, pp. 760-–771 (2011) doi: 10.1016/j.advengsoft.2011.05.014
6. Bhargava. N., Sharma, G., Bhargava, R., Mathuria, M.: Decision tree analysis on J48 algorithm for data mining. In: Proceedings of International Journal of Advanced Research in Computer Science and Software Engineering, vol. 3, no. 6, pp. 1114–1119 (2013)
7. Grewal, D.: A critical conceptual analysis of definitions of artificial intelligence as applicable to computer engineering. IOSR Journal of Computer Engineering, vol. 16, no. 2, pp. 9–13 (2014) doi: 10.9790/0661-16210913
8. Hernández, S.: Mejoras al algoritmo de evolución diferencial para resolver problemas de optimización con restricciones. Universidad Nacional de la Patagonia Austral (2015)
9. Lichman, M.: UCI Machine Learning Repository, University of California, Irvine, School of Information and Computer Sciences (2013)
10. Murthy, S., Kasif, S., Salzberg, S.: A system for induction of oblique decision trees. Journal of Artificial Intelligence Research, vol. 2, pp. 1–32 (1994) doi: 10.1613/jair.63
11. Quinlan, J. R.: C4.5: Programs for machine learning. Machine Learning, vol. 16, pp. 235–240 (1994) doi: 10.1007/BF00993309
12. Rivera-Lopez, R., Canul-Reich, J., Mezura-Montes, E., Cruz-Chávez, M. A: Induction of decision trees as classification models through metaheuristics. Swarm and Evolutionary Computation, vol. 69 (2022) doi: 10.1016/j.swevo.2021.101006
13. Rivera-Lopez, R., Canul-Reich, J.: Construction of near-optimal axis-parallel decision trees using a differential-evolution-based approach. IEEE Access, vol. 6, pp. 5548-5563 (2018) doi: 10.1109/ACCESS.2017.2788700
14. Rivera-Lopez, R., Canul-Reich, J.: A global search approach for inducing oblique decision trees using differential evolution. In: Canadian Conference on Artificial Intelligence, vol. 10233, pp. 27–38 (2017) doi: 10.1007/978-3-319-57351-9_3
15. Storn, R., Price, K.: Differential evolution-A simple and efficient heuristic for global optimization over continuous spaces. Journal of global optimization, vol. 11, no. 4, pp. 341–359 (1997) doi: 10.1023/A:1008202821328
16. Tanabe, R., Fukunaga, A.: Success-history based parameter adaptation for differential evolution. In: 2013 IEEE Congress on Evolutionary Computation, pp. 71–78 (2013) doi: 10.1109/CEC.2013.6557555
17. Witten, I., Frank, E., Hall, M.: Data mining practical machine learning tools and techniques. The Morgan Kaufmann Series in Data Management Systems (2005) doi: 10.1016/C2009-0-19715-5
18. Zhang. J., Sanderson, A.: JADE: Adaptive differential evolution with optional external archive. EEE Transactions on Evolutionary Computation, vol. 13, no. 5, pp. 945–958 (2009) doi: 10.1109/TEVC.2009.2014613

ISSN 1870-4069

Hacia la predicción de pacientes con diabetes: Comparación de algoritmos de aprendizaje automático

Edgar García-Quezada, Carlos E. Galván-Tejada,
José M. Celaya-Padilla, Irma González-Curiel,
Jorge I. Galván-Tejada

Universidad Autónoma de Zacatecas,
Unidad Académica de Ingeniería Eléctrica,
México

{39207895, ericgalvan, jose.celaya,
irmacuriel, gatejo}@uaz.edu.mx

Resumen. Según la organización mundial de la salud, la diabetes mellitus es una enfermedad crónico degenerativa que aparece cuando el páncreas no secreta insulina suficiente o cuando el organismo no la utiliza de manera adecuada. Para el año 2019, en México, existían 12.8 millones de personas con diabetes. Los síntomas iniciales de la enfermedad se relacionan con la hiperglucemia e incluyen polidipsia, polifagia, poliuria y visión borrosa. El diagnostico de diabetes por parte de un médico puede ser complicado, debido a que intervienen varios factores, entre ellos el examen de sangre, que, aunque es barato, no proporciona información suficiente para realizar el diagnostico, además de ser un método invasivo. El objetivo de este artículo es analizar datos de pacientes diabéticos y personas sanas para hacer un diagnostico temprano por medio de sintomatología de la enfermedad, utilizando para esto los algoritmos: Regresión logística múltiple, Máquina de vectores de soporte y K vecinos más cercanos y evaluando cual de ellos es el mejor utilizando como métrica de evulación el área bajo la curva.

Palabras clave: Diabetes mellitus, algoritmos de aprendizaje automático, diagnóstico, área bajo la curva.

Towards the Prediction of Patients with Diabetes: Comparison of Machine Learning Algorithms

Abstract. According to World Health Organization, diabetes mellitus is a chronic degenerative disease that appears when the pancreas does not secrete enough insulin or even the body does not use it properly. By 2019, In Mexico, there were 12.8 million people with diabetes. The initial symptoms of the disease are related to hyperglycemia and include polydipsia, polyphagia, polyuria, and

blurred vision. Diagnosis of diabetes by a doctor is complicated because several factors are involved, including the blood test, which, although cheap, does not provide enough information to make the diagnosis, in addition to being an invasive method. The objective of this paper is to analyze data from diabetic patients and health people to make an early diagnosis, employing symptoms, of the disease using this the algorithms: Multiple logistic regression, Support Vector Machine and K nearest neighbors and evaluating which of them is more optimal using the area under the curve as the main indicator.

Keywords: Diabetes mellitus, machine learning algorithms, diagnosis, area under the curve.

1. Introducción

La diabetes mellitus es una alteración metabólica que se caracteriza por la presencia crónica de hiperglucemia, acompañada por alteraciones en diferentes niveles del metabolismo de hidratos de carbono, proteínas y lípidos.

Aunque esta enfermedad tiene un origen variado, todos los casos conllevan alteraciones en la secreción de insulina, en la sensibilidad a la acción de la hormona, o en ambas, en algún momento de su historia natural.

Cabe destacar que la sintomatología de la enfermedad puede llevar a dos situaciones importantes. En la primera, los síntomas son evidentes, persistentes y las cifras de glucemia suficientemente elevadas, lo que hace que el diagnóstico sea obvio en la mayoría de las ocasiones. En el segundo caso, el paciente podría ser asintomático y requerir una exploración analítica de rutina [10].

Es debido a lo mencionado anteriormente, junto con las complicaciones específicas y la presencia de otros factores asociados a la diabetes mellitus, que esta enfermedad se ha convertido en un grave problema en la actualidad [10].

En México, la diabetes es una enfermedad común en la población desde el año 2000 [22]. Según el Instituto Nacional de Salud Pública, para el 2010 la enfermedad ya había afectado a 83,000 personas [22]. Para el año 2019, la Federación Internacional de Diabetes reportó que en México había 12.8 millones de personas con diabetes [8].

Aunque el método más común para diagnosticar la enfermedad es por medio de pruebas de sangre baratas [3], este no es suficiente para una valoración completa. Por lo tanto, esta investigación se enfoca en buscar un diagnóstico no invasivo para la diabetes mellitus a través de síntomas que puedan acompañar a la enfermedad.

Para ello, se ha implementado algoritmos de aprendizaje automático en una base de datos de pacientes con diabetes tipo 2 (DT2), elaborada por el Sylhet Diabetes Hospital de Sylhet, Bangladesh [11].

El diagnóstico de la diabetes mellitus puede resultar complicado debido a las características cambiantes de la enfermedad y a los posibles errores humanos al momento de identificarla. Los análisis de laboratorio, aunque son indicadores fuertes de la patología, pueden ser interpretados de forma errónea. Por esta causa, en los últimos años se ha buscado una forma más precisa de identificar a los pacientes que padecen la enfermedad.

Fig. 1. Diferentes etapas y pasos en la implementación de los algoritmos, desde la selección de la base de datos hasta la evaluación de los resultados obtenidos.

En el estudio de Benítez, et al. [4], se habla acerca de la predicción de este padecimiento implementando Máquinas de vectores de soporte en pacientes de Baja California.

Su estudio demostró una exactitud del 99.2% en pacientes mexicanos, utilizando como indicadores el índice de masa corporal y la concentración de glucosa en la sangre. En el artículo escrito por Sisodia, D. S. [24] el objetivo es predecir diabetes utilizando varios métodos (árboles de decisión, máquina de vectores de soporte y Naive Bayes), en sus resultados se muestra que, para sus predicciones, el método Naive Bayes es el mejor, ya que tiene una exactitud del 76.3%.

En la propuesta de Liao-Li, et al. [15], el objetivo es ayudar en la prevención y diagnóstico de la enfermedad, así como poder predecir complicaciones que puedan surgir durante el control del mencionado padecimiento.

Para ello, se utilizó la Regresión Lineal Múltiple, la Regresión Logística, Bosques aleatorios y los K vecinos más cercanos. Su estudio demostró que, aunque el Bosques aleatorios obtuvo la mayor área bajo la curva, al ser un modelo no supervisado, se consideró como mejor modelo los K vecinos más cercanos.

La principal contribución de este artículo es analizar datos de pacientes con diabetes mellitus y los principales síntomas asociados a esta enfermedad, así como pacientes sanos, con el objetivo de realizar un diagnóstico temprano del padecimiento por medio de sintomatología, utilizando para esto los algoritmos de aprendizaje automático.

El presente artículo está organizado de la siguiente manera: la primera sección muestra la introducción y estudios relacionados con esta investigación; en la segunda sección se presentan los datos utilizados y la metodología para su tratamiento con el fin de predecir la diabetes; en la tercera sección se describen los resultados obtenidos, así como las comparaciones entre los métodos utilizados; y la cuarta sección tiene como objetivo presentar las conclusiones del estudio realizado.

Table 1. Descripción de atributos de la base de datos.

Atributos	Valores
Edad	20-90
Sexo	1.Masculino, 2. Femenino
Poliuria	1. Si, 2. No
Polidipsia	1. Si, 2. No
Pérdida de peso repentina	1. Si, 2. No
Debilidad	1. Si, 2. No
Polifagia	1. Si, 2. No
Candidiasis	1. Si, 2. No
Vista borrosa	1. Si, 2. No
Purito	1. Si, 2. No
Irritabilidad	1. Si, 2. No
Curación tardía	1. Si, 2. No
Paresis parcial	1. Si, 2. No
Rigidez muscular	1. Si, 2. No
Alopecia	1. Si, 2. No
Obesidad	1. Si, 2. No

2. Materiales y métodos

En la presente sección se muestra la metodología que se llevó a cabo para cada uno de los algoritmos (Fig. 1), así como la descripción de la base de datos utilizada.

Para la evaluación de los algoritmos seleccionados se utilizaron datos provenientes del repositorio de la Universidad de California, Irvine (UCI) Machine Learning. Este conjunto de datos contiene información sobre pacientes, incluyendo síntomas y signos característicos que influyen en el desarrollo de la diabetes.

Los datos se obtuvieron mediante un cuestionario aplicado a pacientes recién diagnosticados con la enfermedad, así como a personas que presentaban algunos de los síntomas aunque no la padecían. El cuestionario fue aplicado directamente a pacientes del Sylhet Diabetes Hospital de Sylhet, Bangladesh.

El conjunto de datos consta de 16 características que representan los síntomas que podrían tener pacientes con diabetes, así como la clasificación de las personas que padecen (Positivo) o no (Negativo) la enfermedad [11]. La descripción de los atributos se muestra en la Tabla 1.

2.1. Definición de los atributos

- Edad: Es la edad de los pacientes encuestados.
- Sexo: Es el sexo de los pacientes encuestados.
- Poliuria: Es la emisión de un volumen de orina superior al esperado [2].
- Polidipsia: Aumento anormal de sed que lleva a una persona a ingerir grandes cantidades de agua [2].
- Pérdida de peso repentina: Pérdida de peso, en un periodo corto de tiempo, sin explicación o causa aparente [2].

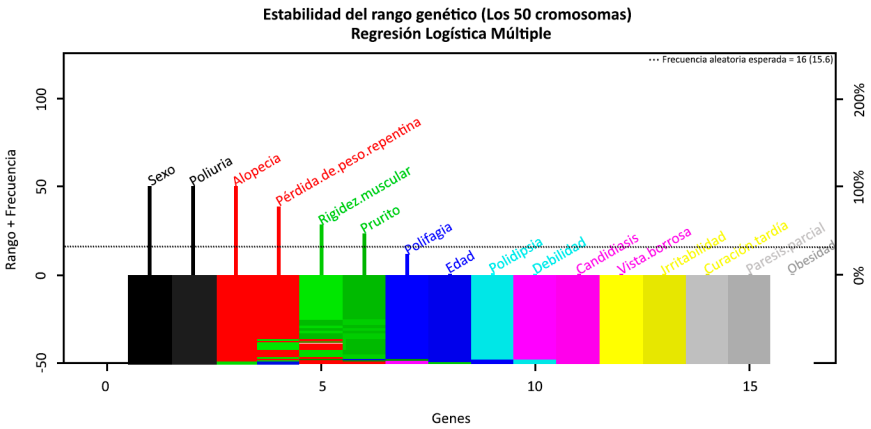


Fig. 2. Gráfica de estabilidad de genes para el modelo de regresión logística múltiple.

- Debilidad: Se refiere a la pérdida de la fuerza muscular, es decir, la persona afectada no puede mover un músculo normalmente a pesar de intentarlo con todas sus fuerzas [14].
- Polifagia: Aumento anormal de la necesidad de querer comer [25].
- Candidiasis: Es un tipo de infección fúngica causada por un hongo denominado cándida [7].
- Vista borrosa: es la pérdida de la agudeza visual, lo que hace que los objetos aparezcan fuera de foco y con opacidad [29].
- Prurito: Sensación incómoda que crea deseo de rascarse [1].
- Irritabilidad: Es un estado emocional en el que una persona tiene un temperamento explosivo y se molesta o enoja fácilmente [28].
- Curación tardía: Es cuando una herida presenta dificultad para cicatrizar o permanecer cerrada [6].
- Paresis parcial: Es la ausencia parcial de movimiento voluntario, la parálisis parcial o suave, descrito generalmente como debilidad del músculo [19].
- Rigidez muscular: Se refiere a músculos tensos o rígidos [21].
- Alopecia: Pérdida anormal de cabello [12].
- Obesidad: Es el exceso de acumulación de grasa en el cuerpo [27].

En cuanto al preprocesamiento de la base de datos, se realizó una verificación de los datos faltantes en las propiedades existentes en la primera etapa. Luego, se procedió a binarizar las 15 características y normalizar la información. Se decidió normalizar la característica Edad, que es un dato continuo, para evitar que afecte el rendimiento de los modelos predictivos. También se evaluó la distribución de los diagnósticos de los pacientes, y se encontró que había 320 casos positivos y 200 negativos, lo que indica una buena distribución entre ellos.

La base de datos se particionó en dos, utilizando el 70% de los datos, seleccionados aleatoriamente, como conjunto de entrenamiento y el 30% restante como conjunto de prueba.

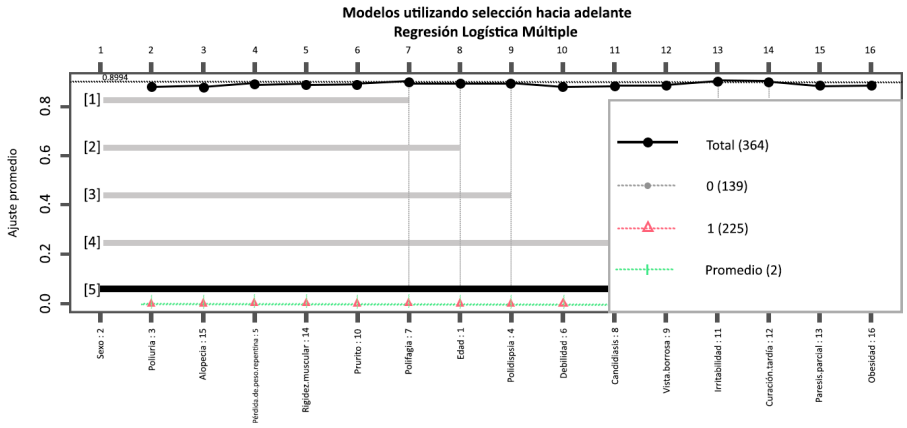


Fig. 3. Gráfica que muestra el mejor modelo usando selección hacia adelante para regresión logística múltiple.

Esto se realizó de esta manera ya que entrenar el modelo con la totalidad de los datos no nos permitiría saber si éste se esta comportando adecuadamente, ya que podría estarse prestando a un sobreajuste, es decir, que el modelo esté describiendo los datos perfectamente bien sin ser realmente adecuado.

Por eso se emplea un número de datos considerablemente grande y aleatorio para asegurarse que se describa lo mejor posible la información y lo restante se utilizará para llevar a cabo las pruebas y que se pueda asegurar que el modelo esté trabajando de forma adecuada [13].

2.2. Método de selección de variables

Para la selección de características, el procedimiento implementado es GALGO, el cual es un paquete que implementa algoritmos genéticos para la resolución de problemas de optimización, estos implican la selección de subconjuntos de variables e incluye una serie de métodos para realizar la clasificación supervisada [26].

Para cada algoritmo se construyó un modelo de selección hacia adelante, esto es un procedimiento de elección de características por pasos en el que las variables se introducen secuencialmente en el modelo.

El primer atributo considerado para la entrada en la ecuación es el que tiene mayor correlación positiva o negativa con la variable dependiente [23].

Por último, se hizo implementó una eliminación hacia atrás, esto es, un procedimiento de selección de variables en el que todas las características se introducen en la ecuación y luego se eliminan secuencialmente. El atributo con la correlación parcial más pequeña respecto a la variable dependiente se considera en primer lugar para la eliminación [20].

Table 2. Selección de características para cada modelo.

Algoritmo	Caracterśticas
Regresíon Logística Múltiple	Sexo, Poliuria, pérdida.de.peso.repentina, debilidad, Prurito, Irritabilidad, Rigidez.muscular, Alopecia.
Máquinas de vectores de soporte	Edad, Sexo, Poliuria, pérdida.de.peso.repentina, Polifagia, Rigidez.muscular, Alopecia.
K vecinos más cercanos	Poliuria, Sexo, curación.tardía, Alopecia, Polidipsia, vista.borrosa.

2.3. Algoritmos

Para desarrollar este estudio, se utilizaron tres algoritmos de aprendizaje automático: Regresión Logística Múltiple, Máquina de Soporte de Vectores y K vecinos más cercanos.

Estos algoritmos fueron seleccionados por sus diferentes enfoques de análisis. La Regresión Logística Múltiple se utiliza para predecir la probabilidad de diferentes resultados posibles de una distribución categórica, dada un conjunto de variables independientes.

La Máquina de Soporte de Vectores correlaciona los datos en un espacio de características de grandes dimensiones, lo que permite categorizar los puntos de datos incluso si no se pueden separar linealmente. Por último, K vecinos más cercanos busca las distancias entre una consulta y todos los ejemplos de los datos, seleccionando los K ejemplos más cercanos a la consulta y votando por la etiqueta más frecuente.

Regresión logística múltiple. La regresión logística múltiple (RLM) es una extensión del modelo de regresión logística simple en el que se predice una respuesta binaria en función de múltiples predictores, que pueden ser tanto continuos como categóricos [5]:

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1+...+\beta_p X_p}}{1 + e^{\beta_0+\beta_1 X_1+...+\beta_p X_p}}, \tag{1}$$

donde $X = X_1, ..., X_p$ son los p predictores [5], como se presenta en la ecuación 1.

Máquinas de vectores de soporte. Las Máquinas de Vectores de Soporte (SVM) permiten encontrar la forma óptima de clasificar entre varias clases. La clasificación óptima se realiza maximizando el margen de separación entre las clases. Los vectores que definen el borde de esta separación son los vectores de soporte [9].

K vecinos más cercanos. El método los k vecinos más cercanos (KNN) es uno de los métodos más importantes de clasificación supervisada. En el proceso de aprendizaje no se hace ninguna suposición acerca de la distribución de las variables predictoras, es por ello que es un método de clasificación no paramétrico, que estima el valor de la función de densidad de probabilidad o directamente la probabilidad posterior de que un elemento pertenezca a la clase va partir de la información proporcionada por el conjunto de entrenamiento [17].

Table 3. Resultados de las métricas de validación para los modelos seleccionados.

Métrica	RLM	SVM	KNN
Precisión	0.8782	0.8526	0.8856
Sensibilidad	0.902	0.8750	0.9462
Especificidad	0.8333	0.8077	0.7937
Área bajo la curva	0.8645	0.8323	0.8902

El grado de cercanía entre dos tuplas $X_1 = (X_{11}...X_{1n})$ y $X_2 = (X_{21}...X_{2n})$ está basado en la distancia Euclidiana. El modelo matemático se describe en la ecuación (2):

$$dist(X_1, X_2) = \sqrt{\sum_{i=1}^{n}(x_{1i} - x_{2i})^2}. \tag{2}$$

2.4. Validación de modelos

Con el objetivo de analizar la capacidad de clasificación de cada modelo, una vez obtenidos, se emplearon indicadores de validación. Las medidas de validación empleadas son las siguientes:

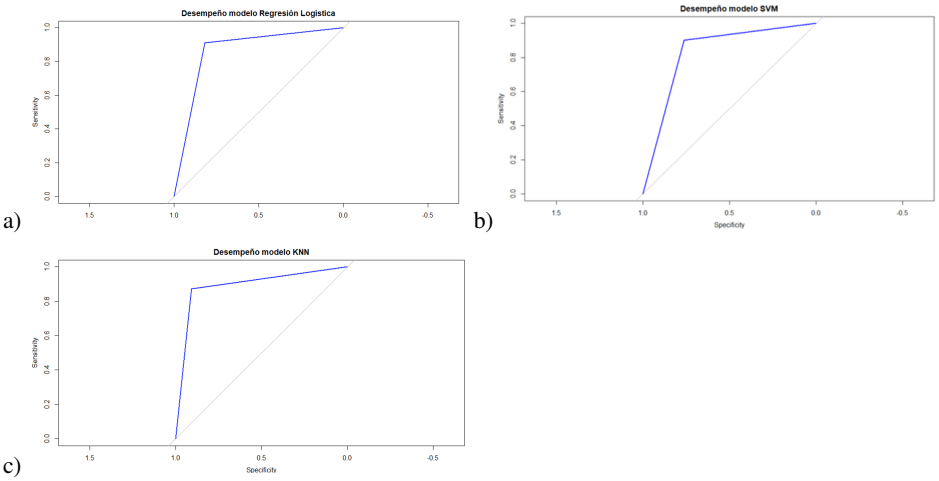
Curvas ROC
El análisis de la curva característica de operación del receptor (ROC) es una herramienta muy utilizada para el análisis del rendimiento del modelo de clasificación, ya que permite evaluar la sensibilidad y la especificidad de un modelo a diferentes umbrales de decisión.

La curva ROC representa la tasa de verdaderos positivos (sensibilidad) en función de la tasa de falsos positivos (1 - especificidad), lo que aporta información suficiente para elegir el mejor umbral y minimizar tanto los falsos positivos como los falsos negativos [16].

Área bajo la curva
El área bajo la curva (AUC), se utiliza como resumen de la rentabilidad del modelo, es decir, cuanto más esté hacia la izquierda la curva, más área habrá contenida bajo ella y, por ende, mejor será el clasificador. El clasificador aleatorio tendría una AUC de 0.5 mientras que el clasificador perfecto tendría el valor de 1 [17].

Sensibilidad y especificidad
La sensibilidad es una métrica que nos permite visualizar el desempeño del modelo, es decir, es la proporción de casos positivos que fueron correctamente identificados por el algoritmo, cuyo modelo se describe en la ecuación (3). Por otra parte, la especificidad se refiere a los casos negativos que se han clasificado correctamente, expresado en la ecuación (4). Además, expresa qué tan bien el modelo puede detectar esa clase [18]:

$$Sensibilidad = \frac{VP}{VP + FN}, \tag{3}$$



a) b) c)

Fig. 4. Resultados de las pruebas realizados para el 30% de los datos restantes utilizando validación cruzada. Para RLM (a), el área bajo la curva es de 0.8645; Para SVM (b), el área es de 0.8323 y para KNN (c) es de 0.8902.

$$Especificidad = \frac{VN}{VN + FP}, \tag{4}$$

donde VP son los verdaderos positivos; VN, los verdaderos negativos; FP, los falsos positivos y FN, los falsos negativos.

3. Resultados

De los resultados obtenidos con el entrenamiento de GALGO, utilizando el 70% de la base de datos, la gráfica 2 muestra las características ordenadas de izquierda a derecha por su grado de importancia en el modelo de predicción. Además, se puede observar la estabilidad de cada característica, lo que significa que una característica no cambiará de color si no cambia su estado (es decir, si se mantiene importante para el modelo mezclándose con otras características). Lo anterior se realiza para todos los algoritmos propuestos y para obtener de cada uno de ellos las características más descriptivas de la variable dependiente, en este caso, pacientes con o sin diabetes. Cabe mencionar que para el algoritmo KNN se utilizó K = 1 y para SVM el kernel seleccionado fue radial.

Después se construyó un modelo hacia adelante utilizando las características que se encontraron con GALGO. En las gráficas del modelo hacia adelante se puede observar cómo se seleccionan las variables más importantes al principio y se van agregando para crear el mejor modelo. Como ejemplo, para Regresión Logística Múltiple, se puede observar dicho comportamiento en la figura 3.

Para finalizar el proceso de selección de características, se implementó una eliminación hacia atrás tomando como base los modelos con mejor rendimiento en la parte de selección hacia adelante, eliminado las variables que menos aportan el resultado final; para cada modelo se muestran en la Tabla 2.

Table 4. Métricas de validación para el método de ensamble por mayoría de votos.

Métrica	Ensamble por mayoría de votos
Precisión	0.8910
Sensibilidad	0.9286
Especificidad	0.8276
ROC	0.8869

Como parte final del proceso se realizaron las pruebas de los modelos seleccionados con las características más significativas y estables arrojadas por GALGO. Las pruebas se realizaron con el 30% restante de la base de datos con una validación cruzada. Los resultados de estas pruebas se pueden observar en la Tabla 3 y las curvas ROC que se muestran en la figura 4.

Como se puede observar en la tabla 3 los modelos se comportan de manera similar, sin embargo el mejor desempeño con base a los objetivos del presente trabajo, es el modelo de los K vecinos más cercanos, pues este tiene la mejor precisión y la mayor sensibilidad, siendo esta última un apartado relevante tratándose de la salud de los pacientes, dado que es de especial importancia para la salud que a una persona que no tenga diabetes, no se le clasifique por error como enferma y que a una que tenga la enfermedad, se le clasifique erróneamente como sana.

Lo siguiente fue utilizar el método de ensamble por mayoría de votos para los resultados de los modelos y así validar la posible mejora del desempeño de estos, resultados que se muestran en la tabla 4 y en la figura 5.

En la tabla 4 se observa que la precisión mejoró aproximadamente un 1% en comparación con el mejor modelo, que fue KNN, mientras que la sensibilidad disminuyó alrededor de un 2%. Por lo tanto, el modelo de los K vecinos más cercanos sigue siendo el mejor método.

4. Conclusiones

Benítez utilizó Máquinas de Vectores de Soporte para predecir la diabetes en pacientes de Baja California, México. Su trabajo concluyó que la exactitud de este método en su base de datos demostró ser del 99.2%, utilizando el índice de masa corporal y la glucosa en sangre como indicadores. Por otro lado, Sisodia, D. S. encontró que el algoritmo Naive Bayes ofrece la mayor exactitud en la predicción de la diabetes, con un 76.3%, tras evaluar varios métodos. En su estudio, Liao-Li propuso diagnosticar la diabetes en pacientes mexicanos utilizando diferentes algoritmos. Llegó a la conclusión de que el algoritmo K vecinos más cercanos proporciona la mayor área bajo la curva y una exactitud del 87.31%.

De acuerdo con los resultados obtenidos por los modelos analizados en este estudio, el modelo K vecinos más cercanos tiene el mejor comportamiento, dada su precisión de 0.8846, una área bajo la curva de 0.8902 y la curva ROC muestra una sensibilidad de 0.9462, siendo una característica importante para este estudio, pues clasifica con exactitud a los pacientes que están enfermos. Logra este rendimiento con las siguientes características: poliuria, sexo, curación tardía, alopecia, polidipsia y vista borrosa. Estas modelan al conjunto de prueba, por lo que es posible hacer el diagnóstico inicial de forma no invasiva, cumpliendo con el objetivo general del estudio desarrollado.

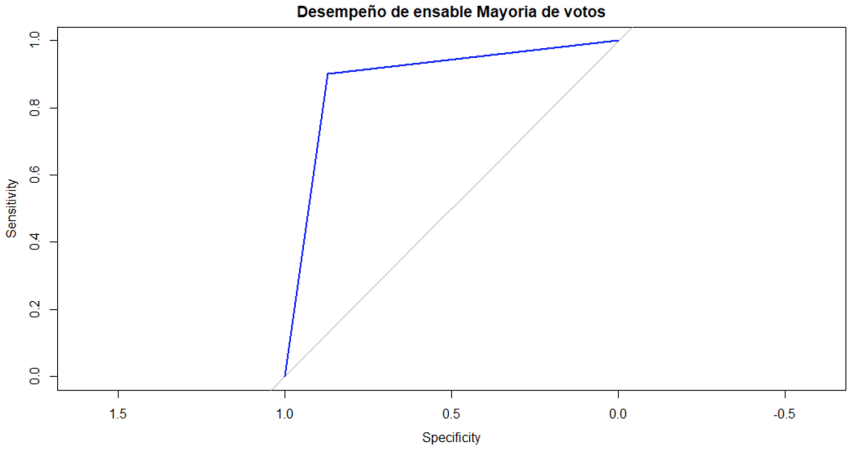


Fig. 5. Curva ROC para el modelo ensamble por mayoría de votos.

Además, de los modelos analizados, el que tiene menor desempeño fue el de máquina de vectores de soporte, pues éste cuenta con una precisión de 0.8526, un área bajo la curva de 0.8323 y la curva ROC muestra una sensibilidad de 0.8750. Sin embargo es importante mencionar que SVM requiere solamente de 7 características para lograr dicho desempeño; una característica menos que las requeridas por la Regresión Logística Múltiple.

La técnica de ensamble mostró una precisión de 0.891, una área bajo la curva de 0.8869 y una curva ROC que muestra una sensibilidad de 0.9286, aunque muestra un pequeño aumento en la precisión, disminuye el rendimiento en otras métricas, por esto no se le considera una técnica adecuada para este caso.

El resultado del modelo de K vecinos más cercanos puede ser utilizado para desarrollar una herramienta auxiliar en el diagnóstico médico, pues permitiría hacer un acercamiento al diagnóstico temprano de diabetes, siendo este corroborado después por un experto de la salud.

Como trabajo a futuro se propone implementar otros algoritmos que aborden el problema de la clasificación de los pacientes de distintas maneras, permitiendo obtener mejores resultados, además de otras técnicas de ensamble que ayuden a mejorar el rendimiento de los algoritmos por sí solos.

También se pueden explorar otras variables que no se incluyeron en este estudio, para ver si tienen un impacto significativo en la clasificación de los pacientes diabéticos. Por ejemplo, se pueden considerar variables relacionadas con el estilo de vida de los pacientes, como su nivel de actividad física, su dieta y su nivel de estrés.

Agradecimientos. Los autores agradecen al Consejo Nacional de Ciencia y Tecnología (CONACyT) y a la Universidad Autónoma de Zacatecas por el apoyo y financiamiento para la realización de este proyecto.

References

1. Alcalá-Pérez, D., Barrera-Pérez, M., Santa-Cruz, F.: Fisiopatología del prurito. Revista del Centro Dermatológico Pascua, vol. 23, no. 1, pp. 6–10 (2014)
2. Almaguer-Herrera, A., Miguel-Soca, P., Reynaldo-Será, C., Mariño-Soler, A. L., Oliveros-Guerra, R.: Actualización sobre diabetes mellitus. Correo Científico Médico, vol. 16, no. 2 (2012)
3. American Diabetes Association: Classification and diagnosis of diabetes. Diabetes Care, vol. 39, no. 1, pp. 13–22 (2016) doi: 10.2337/dc16-S005
4. Benítez, B., Castro, C., Castañeda-Martínez, R. A., Abaroa, A.: Predicción de diagnóstico de diabetes mellitus utilizando máquinas de soporte vectorial en pacientes de Baja California. In: Memorias Del XL Congreso Nacional De Ingeniería Biomédica, vol. 4, no. 1, pp. 415–418 (2017)
5. Berea-Baltierra, R., Rivas-Ruiz, R., Pérez-Rodríguez, M., Palacios-Cruz, L., Moreno, J., Talavera, J. O.: Investigación clínica XX del juicio clínico a la regresión logística múltiple. Revista Médica del Instituto Mexicano del Seguro Social, vol. 52, no. 2, pp. 192–197 (2014)
6. Berlanga-Acosta, J., Valdez-Pérez, C., Savigne-Gutierrez, W., Mendoza-Marí, Y., Franco-Perez, N., Vargas-Machiran, E., Poll-Marrón, N., Alvarez-Duarte, H., Echeverria-Requeijo, H., Perez-Aguilar, R. M.: Particularidades celulares y moleculares del mecanismo de cicatrización en la diabetes. Biotecnología Aplicada, vol. 27, no. 4, pp. 255–261 (2010)
7. Biasoli, M.: Candidiasis (2013) http://www.fbioyf.unr.edu.ar/evirtual/file.php/118/MATERIALES_2013/TEORICOS_20
8. Centro de Investigación en Alimenación y Desarrollo CIAD: La pandemia de diabetes en México (2020) https://www.ciad.mx/notas/item/2450-la-pandemia-de-diabetes-en-mexico
9. Cervantes, J., Garcia-Lamont, F., Rodríguez-Mazahua, L., Lopez, A.: A comprehensive survey on support vector machine classification: Applications, challenges and trends. Neurocomputing, vol. 408, pp. 189–215 (2020) doi: 10.1016/j.neucom.2019.10.118
10. Conget, I.: Diagnóstico, clasificación y patogenia de la diabetes mellitus. Revista Española de Cardiología, vol. 55, no. 5, pp. 528–535 (2002) doi: 10.1016/S0300-8932(02)76646-3
11. Faniqul-Islam, M. M., Ferdousi, R., Rahman, S., Bushra, H. Y.: Likelihood prediction of diabetes at early stage using data mining techniques. Computer Vision and Machine Intelligence in Medical Image Analysis, Advances in Intelligent Systems and Computing, vol. 992, pp. 113–125 (2020) doi: 10.1007/978-981-13-8798-2_12
12. Forouzan, P., Cohen, P. R.: Incipient diabetes mellitus and nascent thyroid disease presenting as beard alopecia areata: case report and treatment review of alopecia areata of the beard. Cureus, vol. 12, no. 7 (2020) doi: 10.7759/cureus.9500
13. Gholamy, A., Kreinovich, V., Kosheleva, O.: Why 70/30 or 80/20 relation between training and testing sets: A pedagogical explanation. Departmental Technical Reports (CS) (2018)
14. Hasbum-Fernández, B., Ampudia-Blasco, F. J., Carmena, R.: Paciente diabético con debilidad muscular generalizada. Endocrinología y Nutrición, vol. 50, no. 5, pp. 169–170 (2003) doi: 10.1016/S1575-0922(03)74521-5
15. Liao-Li, M. K., Aparicio-Montelongo, I., Celaya-Padilla, J. M., Galván-Tejada, C. E., Cruz, M.: Implementación de algoritmos de aprendizaje automático para la predicción de pacientes con diabetes. Investigación Aplicada, un Enfoque en la Tecnología, vol. 6, no. 12, pp. 609–621 (2021)
16. Martínez-Pérez, J. A., Pérez-Martin, P. S.: La curva roc. Medicina de Familia. SEMERGEN, vol. 49, no. 1 (2023) doi: 10.1016/j.semerg.2022.101821
17. Meneses-Villegas, C., Aqueveque, D.: Diagnosis of neuropathies in diabetic patients by applying machine learning. Ingeniare: Revista Chilena de Ingenieria, vol. 29, no. 3, pp. 517–530 (2021) doi: 10.4067/S0718-33052021000300517

18. Pita-Fernández, S., Pértegas-Díaz, S.: Pruebas diagnósticas: Sensibilidad y especificidad. Cad Aten Primaria, vol. 10, no. 1, pp. 120–124 (2003)
19. Rodríguez-Maldonado, A.: Medio para la aplicación de material terapéutico para uso en adultos discapacitados que padecen algún tipo de paresia (debilidad muscular) en miembros superiores. Ph. D. thesis, Universidad de los Andes (2006)
20. Roque-López, J.: Técnicas de selección de variables en regresión lineal múltiple. Master's thesis, Universidad Internacional de Andalucía (2021)
21. Salsich, G. B., Brown, M., Mueller, M. J.: Relationships between plantar flexor muscle stiffness, strength, and range of motion in subjects with diabetes-peripheral neuropathy compared to age-matched controls. Journal of Orthopaedic & Sports Physical Therapy, vol. 30, no. 8, pp. 473–483 (2000) doi: 10.2519/jospt.2000.30.8.473
22. Secretaría de Educación, Ciencia, Tecnología e Innovación (SECTEI): México, segundo país en América Latina con prevalencia de diabetes (2021) https://sectei.cdmx.gob.mx/comunicacion/nota/mexico-segundo-pais-en-america-latina-con-prevalencia-de-diabetes
23. Singh, P., Soni, K., Nair, A. S., Singh, M.: Regression analysis of ventilation coefficient at a semi-arid IGP region using forward selection technique. MAUSAM, vol. 73, no. 3, pp. 617–626 (2022) doi: 10.54302/mausam.v73i3.5933
24. Sisodia, D., Sisodia, D. S.: Prediction of diabetes using classification algorithms. Procedia Computer Science, vol. 132, pp. 1578–1585 (2018) doi: 10.1016/j.procs.2018.05.122
25. Tiwari, D.: Polyphagia can be a side effect of diabetes. African Journal of Diabetes medicine, vol. 30, no. 5 (2022)
26. Treviño, V., Falciani, F.: GALGO: An R package for multivariate variable selection using genetic algorithms. Bioinformatics, vol. 22, no. 9, pp. 1154–1156 (2006) doi: 10.1093/bioinformatics/btl074
27. Vázquez-Morales, E., Calderón-Ramos, Z. G., Arias-Rico, J., Ruvalcaba-Ledezma, J. C., Rivera-Ramírez, L. A., Ramírez-Moreno, E.: Sedentarismo, alimentación, obesidad, consumo de alcohol y tabaco como factores de riesgo para el desarrollo de diabetes tipo 2. Journal of Negative and No Positive Results, vol. 4, no. 10, pp. 1011–1021 (2019) doi: 10.19230/jonnpr.3068
28. Vidal-Ribas, P., Brotman, M. A., Valdivieso, I., Leibenluft, E., Stringaris, A.: The status of irritability in psychiatry: A conceptual and quantitative review. Journal of the American Academy of Child & Adolescent Psychiatry, vol. 55, no. 7, pp. 556–570 (2016) doi: 10.1016/j.jaac.2016.04.014
29. Zhou, S., Carroll, E., Nicholson, S., Vize, C. J.: Blurred vision. BMJ, vol. 368 (2020) doi: 10.1136/bmj.m569

Electronic edition
Available online: http://www.rcs.cic.ipn.mx

http://rcs.cic.ipn.mx

Centro de Investigación
en Computación