# Avances recientes en procesamiento de lenguaje natural y otras áreas afines

Mireya Tovar Vidal
Ana Laura Lezama Sánchez
Meliza Contreras González

Editores

Dirección General de Publicaciones

# Avances recientes en procesamiento de lenguaje natural y otras áreas afines

# Avances recientes en procesamiento de lenguaje natural y otras áreas afines

Mireya Tovar Vidal
Ana Laura Lezama Sánchez
Meliza Contreras González
**Coordinadores**


Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación
2024

Benemérita Universidad Autónoma de Puebla
Rectora: Ma. Lilia Cedillo Ramírez
Secretario General: José Manuel Alonso Orozco
Vicerrector de Extensión y Difusión de la Cultura: José Carlos Bernal Suárez
Director General de Publicaciones: Luis Antonio Lucio Venegas

Primera edición **2024**
ISBN: 978-607-5914-56-5


4 Sur 104, Col. Centro Histórico, Puebla, Pue. CP 72000
Teléfono: 222 229 55 00
www.buap.mx


2 Norte 1404, Centro Histórico, Puebla, Pue., CP 72000
Tels.:01 (222) 246 85 59 y 01 (222) 55 00, ext. 5768
www.dgp.buap.mx | libros.dgp@correo.buap.mx
www.publicaciones.buap.mx

Diseño de portada: Mireya Tovar Vidal

Hecho en México
*Made in Mexico*

# Prólogo

En el presente libro titulado *"Avances recientes en procesamiento de lenguaje natural y otras áreas afines"* se presentan diferentes áreas de investigación de procesamiento del lenguaje natural y propuestas que utilizan aprendizaje automático para su solución que son abordadas por distintos grupos de investigación al interior del país.

La obra incluye once capítulos de investigación divididos en áreas de procesamiento de lenguaje natural, en aplicaciones de inteligencia artificial, en aprendizaje automático, en ciencia de datos, en ontologías, entre otras.

Los capítulos que forman parte de esta obra fueron revisados mediante el sistema de doble par ciego y aprobados para su publicación por expertos en el área de conocimiento, lo que permitió asegurar la calidad científica en las áreas de estudio así como la obra en su totalidad. A continuación se menciona la aportación de cada uno de ellos.

En el Capítulo 1 se aborda el problema de identificación de mensajes suicidas en inglés utilizando un enfoque de clasificación a partir de las clases suicidio y no suicidio. Los algoritmos de aprendizaje automático utilizados en este trabajo son *Naive Bayes* y *Gradient Boost Machine*.

En el Capítulo 2 se presenta un estudio centrado en la detección de fraude digital. En la investigación se revisan varias metaheurísticas como Recocido Multiarranque (MSA), Búsqueda Dispersa (SS), Búsqueda en Vecindades Variables (VNS), Algoritmo Genético (GA) y Optimización por Enjambre de Partículas (PSO) para calibrar hiperparámetos y así mejorar los resultados de las métricas de evaluación de la tarea de detección de noticias falsas en corpus en español.

En el Capítulo 3 se presentan experimentos con modelos de lenguaje de gran tamaño para el análisis de sentimientos aplicados a corporas en español obtenidos de X sobre COVID-19 de los años 2020 y 2021. Realizan varios experimentos y obtienen resultados del 97 % de exactitud con el modelo *beto-setiment-analysis* sin pre-procesamiento de datos.

En el Capítulo 4 se presenta un sistema de clasificación de noticias criminales en español obtenidas por medio de web scrapping de un sitio oficial de periodismo. También se ofrece una visión comparativa de resultados de clasificación basada en los algoritmos: *Random Forest, Support Vector Machine, Naive Bayes* y *Gradient Boosting*. Los datos utilizados son preprocesados por medio de *stemming* y lematización.

En el Capítulo 5 se presenta una red neuronal convolucional basada en grafos, para la detección de acoso en mensajes en español, logrando resultados alentadores. El mensaje es clasificado en tres categorías: presencia de acoso, ausencia de acoso o mensajes ambiguos.

En el Capítulo 6 se presenta una comparación entre diversos algoritmos de aprendizaje automático para la clasificación de textos que exhiben conductas depresivas y suicidas. Se emplean corpus en inglés que son traducidos al español

utilizando *Google Translator*. La investigación utiliza 4 algoritmos de aprendizaje automático y la métrica de exactitud.

En el Capítulo 7 se evalua la eficiencia de algunos algoritmos de aprendizaje automático como máquinas de soporte vectorial, árboles de decisión, $k$-vecinos más cercanos y perceptrón para identificar entidades y relaciones en conjuntos de datos desbalanceados de textos médicos en español.

En el Capítulo 8 se describe una representación de conocimiento u ontología de dominio sobre la arquitectura de control de un robot SCARA y su relación con tecnologías como IoT y MatLAB.

En el Capítulo 9 se caracteriza el comportamiento muscular para identificar las condiciones para maximizar la hipertrofia muscular que ayuda a los deportistas a tener mejoras en su entrenamiento, a través de algoritmos de aprendizaje por refuerzo como *Q-learning* y *Deep Q learning*.

En el Capítulo 10 se presentan algoritmos de aprendizaje supervisado y no supervisado para realizar un análisis del comportamiento de los dispositivos en la red WI-FI pública de la ciudad de Puebla de Zaragoza, así como la aglomeración de los puntos de acceso que conforman esa red. Exploran la complejidad y tráfico de la red en un periodo de tiempo dado para modelar el flujo de datos de ésta y concluir con el grado de confiabilidad que tiene la red WI-FI como servicio público a la comunidad.

Por último, en el Capítulo 11 se implementó una metaheurística del procedimiento de búsqueda adaptativa aleatoria codiciosa (GRASP) para buscar sus soluciones a partir de 29 instancias. La cuál arrojó resultados importantes ya que se enfoca en la adaptación de las estructuras de un vecindario en la fase de post-procesamiento de GRASP para comparar el mejor valor encontrado contra el resultado del mejor valor conocido encontrado.

Finalmente, queremos agradecer a cada uno de los autores por su contribución, al comité revisor por su valiosa labor y cuidado en el proceso de revisión ciega, a la Facultad de Ciencias de la Computación de la Benemérita Universidad Autónoma de Puebla y a todos aquellos cuya participación contribuyó para la publicación de este libro.

Los editores,
Mireya Tovar Vidal
Ana Laura Lezama Sánchez
Meliza Contreras González

# Índice general

# Capítulo 1

## Detección de Suicidio con Naive Bayes y GBM

Ricardo Ulises Caballero Hidalgo[1], Mireya Tovar Vidal[1], Meliza Contreras González [1]

[1] Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación, Puebla, Pue., México
ch223470316@alm.buap.mx, mireya.tovar@correo.buap.mx,
meliza.contreras@correo.buap.mx

**Resumen.** El mundo actual se está viendo afectado por un creciente incremento de enfermedades de salud mental como la depresión y la ansiedad, lo que en muchos casos conlleva al suicidio. Muchas personas expresan ideas de este tipo antes de cometer el triste acto de acabar con la vida. En las redes sociales se observa que las personas plasman sus ideas y pensamientos por lo tanto el aprendizaje automático a través del análisis de textos en redes sociales contribuiría a prevenir en muchos casos este tipo de situación. A pesar de ser un fenómeno potencialmente prevenible, predecir el riesgo de suicidio sigue siendo un desafío. Este estudio tuvo como objetivo desarrollar clasificadores de aprendizaje automático para detectar indicios de suicidio en textos, utilizando el conjunto de datos *Suicide and Depression Detection* de la plataforma Kaggle (Komati, N. *Suicide Watch*). Se implementaron dos modelos: Naive Bayes Multinomial *y Gradient Boost Machine* (GBM). Se obtuvieron métricas favorables, con una exactitud del 0.91 para Naive Bayes y del 0.92 para GBM. En cuanto a la precisión, Naive Bayes alcanzó un 0.89, mientras que GBM obtuvo un 0.91. El puntaje $F_1$ para ambos modelos fue de 0.90 y 0.91, respectivamente. Estos resultados reafirman la eficacia de Naive Bayes y GBM en el análisis de textos mediante modelos de aprendizaje automático.

**Palabras Clave:** Suicidio, Aprendizaje automático, Naive Bayes, Gradient Boost Machine (GBM).

## 1 Introducción

El suicidio se define como el acto deliberado de poner fin a la propia vida. Aunque las razones detrás de este acto pueden ser diversas y complejas, su impacto es innegablemente devastador. Las estadísticas muestran que el suicidio es una de las principales causas de muerte en todo el mundo, lo que subraya la urgente necesidad de abordar este problema de manera integral.

El impacto del suicidio se extiende más allá del individuo que lo comete, afectando profundamente a las familias, amigos y comunidades. Las secuelas emocionales y psicológicas son duraderas, dejando a los seres queridos en un estado de dolor y angustia

inimaginables. La sociedad en su conjunto también sufre las consecuencias, enfrentando la pérdida de talento, potencial humano y contribuciones valiosas.

Las causas subyacentes del suicidio son multifacéticas y pueden variar considerablemente de una persona a otra. Factores como trastornos mentales, experiencias traumáticas, problemas de salud, relaciones interpersonales conflictivas, dificultades económicas y falta de apoyo social pueden desempeñar un papel significativo en la decisión de una persona de quitarse la vida. Es crucial abordar estos factores de manera integral y comprensiva para prevenir el suicidio y promover la salud mental, este sirve para conocer como se comportan 2 de los algoritmos más empleados en el análisis de sentimientos aplicando la metodología de preprocesado y limpieza de datos que se utiliza en este trabajo utilizando el set de datos de la plataforma Kaggle *Suicide and Depression* (Komati, N. *Suicide Watch*). En este trabajo se discuten además técnicas de balanceo de clases y sus resultados.

El trabajo está conformado de la siguiente forma, después de la introducción se encuentra una revisión de trabajos preliminares sobre el tema, le sigue la metodología de solución empleada explicada por pasos. Luego se presentan los resultados y a continuación la discusión del trabajo, finalmente se presentan las conclusiones y las propuestas de trabajo futuro.

## 2 Preliminares

A continuación, se presenta un estudio sobre trabajos relacionados con el tema de investigación, así como otros referentes a algoritmos de aprendizaje automático.

En el ámbito de la prevención del suicidio, el análisis de datos y la aplicación de técnicas computacionales han surgido como herramientas poderosas para identificar patrones y factores de riesgo que pueden ayudar a prevenir este trágico acto. Una de las técnicas más utilizadas en este contexto es el clasificador Naive Bayes, así como las técnicas de Ensamble.

Existen numerosos estudios sobre estos clasificadores para el análisis de sentimientos, en (Spasić, I. et al 2012), utilizaron Naive Bayes con Weka para detectar indicios de suicidio en 600 notas, obteniendo un puntaje $F_1$ de 53%, superando una gran mayoría de modelos que habían obtenido un puntaje $F_1$ de 50%. Argumentan además en (Spasić, I. et al 2012) que este tipo de herramienta no necesariamente necesita muchos datos de entrenamiento para lograr buenos resultados, lo que en muchos casos supone un problema a la hora de entrenar modelos de clasificación.

En (Rai, P et al 2024) se plantea que identificar a personas que muestran signos de depresión en las redes sociales es difícil, ya que su estado mental es inherentemente impredecible, en este estudio se compara la precisión de los algoritmos de aprendizaje automático (Naive Bayes, Regresión Logística y algoritmo de Máquinas de Soporte Vectorial) utilizando 3 categorías de sentimientos (positivo, negativo y neutro) donde se obtuvo una exactitud de 89, 89 y 95% respectivamente para cada modelo.

En un estudio realizado en (Rabani, et al, 2021), recopilan texto a partir de la API de Twitter (Ahora X) y Reddit y se utilizaron 12,534 registros de un total de 40,000 recopilados inicialmente. Los posts se clasificaron en 3 categorías las cuales fueron bajo riesgo, medio riesgo y alto riesgo. Otro estudio (Ao, Z et al 2021, September) utiliza el conjunto de datos empleado en este trabajo para comparar varios modelos como LSTM, GRU, LSTM bidireccional, clasificador logístico, SVM, XGBoost y LGBM para la clasificación de sentimientos. Comparando las ventajas y desventajas de cada modelo, y se encontró que LSTM unidireccional alcanzó una exactitud del 92.4%. No se conocen otras métricas empleadas en este estudio.

En (Desmet, B., & Hoste, V. 2018) utilizan algoritmos genéticos para detectar indicios de suicidio en notas en idioma holandés obtenidas de redes online, en este estudio se presta especial énfasis en la selección de características (*feature selection*) y la optimización de hiperparámetros (*hyperparameter tunning*). Los resultados indicaron que el análisis de texto para la detección de posibles casos de suicidio es una técnica viable y prometedora a la hora de detectar estos indicios obteniendo un 93% de puntaje $F_1$ en mensajes relevantes y 70% en mensajes severos. En (Desmet, B., & Hoste, V. 2018) plantean además que uno de los principales obstáculos para el entrenamiento apropiado de modelos de aprendizaje automático supervisado es la obtención de los datos y que en el caso particular de suicidio es difícil de obtener. En la literatura de clasificación de texto, las máquinas de soporte vectorial y el Naive Bayes son comúnmente utilizados. Cuando se ajustan correctamente, se ha observado que logran un rendimiento similar o mejor en comparación con algoritmos más complejos, y típicamente son suficientes para resolver problemas prácticos de categorización de texto.

En otro estudio (Paz, L. et al) plantean que el aprendizaje profundo ha desempeñado un papel importante en las predicciones y proponen un enfoque novedoso de detección que utiliza aprendizaje profundo. Esencialmente, el estudio analiza datos de lenguaje natural en bruto de diferentes fuentes, como redes sociales, y detecta pensamientos suicidas a través del texto, mencionan en el estudio que el modelo de BERT puede lograr un resultado de clasificación óptimo. El conjunto de datos empleado proviene del departamento de Investigación Criminal de Kenya y otras fuentes de redes sociales. En este estudio se obtuvieron varios valores de Puntuación $F_1$ con un promedio de 0.64 para Naive Bayes y 0.90 para BERT (Paz, L. et al).

El uso de diferentes clasificadores, como la Regresión Logística, Bosque Aleatorio, Máquina de Soporte Vectorial y Redes Neuronales, se han explorado para la detección de individuos con riesgo de autolesión en redes sociales como Reddit. Estudios recientes han demostrado que la Regresión Logística y el Bosque Aleatorio muestran un mejor rendimiento en esta tarea, mientras que las redes neuronales tienden a tener más falsos positivos (Roza, T et al 2023).

En el contexto de la prevención del suicidio, la predicción sigue siendo un desafío significativo. En (Roza, T et al 2023) desarrollan clasificadores de aprendizaje automático para identificar el riesgo de suicidio en brasileños con trastornos mentales comunes. Utilizando datos clínicos y sociodemográficos de una muestra comunitaria en Brasil (n = 4,039), se implementaron varios modelos de clasificación, destacando los bosques

aleatorios con un área bajo la curva (AUC ROC) de 0.814. La presencia de síntomas depresivos fue fundamental para la identificación del riesgo suicida. Estos hallazgos subrayan la utilidad de las técnicas de aprendizaje automático en la detección temprana del riesgo de suicidio y la importancia de intervenciones preventivas (Murphy, K. P. 2006).

En (Sanderson, M., et al 2020) se comparó el rendimiento de diferentes modelos predictivos para identificar el riesgo de suicidio. Utilizando un conjunto de datos que abarcó de 2000 a 2016, con 3,548 casos de suicidio y 35,480 controles, y 4,040 predictores por persona, se evaluaron modelos de redes neuronales recurrentes, redes neuronales convolucionales unidimensionales y árboles de decisión potenciados por gradiente. Los resultados mostraron que estos modelos superaron a la regresión logística en términos de área bajo la curva, destacándose el modelo XGB *(Extreme Gradient Boosting)* por su superior discriminación y calibración.

Después de revisar trabajos significativos relacionados con el problema de la detección de indicios de suicidio en textos demuestra como los modelos de aprendizaje son capaces de llevar a cabo esta tarea de gran importancia para la sociedad, la salud y el desarrollo tecnológico en general.

En este estudio se utilizan dos clasificadores Naive Bayes y GBM para la detección de suicidio a partir de textos, se realiza una comparación entre ambos modelos con otros estudios antes realizados.

## 3 Propuesta de Solución

A continuación, se describen los pasos para llevar a cabo la propuesta de solución, la cual se basa en (Rai, P., et al 2024).

### 3.1 Colección de Datos

En este paso se utiliza una base de datos de la plataforma *Kaggle* (Komati, N. *Suicide Watch*), En la Tabla 1 se observa el formato del conjunto de datos.

**Tabla 1.** Formato del conjunto de datos.

| Unnamed | Text | Class |
|---|---|---|
| 1 | Ex wife threatening Suicide recently…. | suicide |
| 2 | Am I weird I don't get affected by compliments… | non-suicide |
| 3 | Finally, 2020 is almost over… | suicide |
| 4 | I'm so lost hello, my name is ... | suicide |

### 3.2 Preprocesado de los datos

En este paso se utiliza una base de datos de la plataforma *Kaggle* (Komati, N. Suicide Watch).

Los datos necesitan ser preprocesados para prepararlos en formatos adecuados para la subsecuente construcción del modelo. Los textos informales como los que abundan en las redes sociales tienden a requerir preprocesado y limpieza. En esta etapa del proceso se utiliza la metodología empleada en (Dus, Y., & Nefedov, G. (2023))

El procesado consiste en las siguientes etapas:

1. Eliminar palabras con acentos que tienen el mismo significado. Esto sirve además para reducir los vectores al contener menor cantidad de palabras, lo que incurriría en mayores tiempos de procesamiento y mayor complejidad de los modelos.
2. Expansión de contracciones: En las redes sociales se suele escribir de manera informal utilizando contracciones, en esta parte se expanden las palabras contraídas.
3. Conversión de caracteres a minúsculas: Las palabras no pierden su significado.
4. Eliminar símbolos, caracteres especiales, direcciones de internet, espacios en blancos innecesarios: Estos caracteres carecen de importancia para el modelo además de aumentar la complejidad de este.
5. Eliminación de palabras con letras repetidas, por ejemplo, palabras como loveee donde se repite la letra e.
6. Corrección ortográfica con Symspell *(Symmetric Spelling)* se usa para la corrección ortográfica rápida y puede manejar correcciones en cadenas de varias palabras, como separar palabras combinadas por la falta de un espacio. Es un algoritmo eficiente y rápido, su finalidad es separar palabras que se encuentren juntas.
7. Eliminar palabras vacías: Son palabras que aparecen comúnmente pero no tienen importancia o significado. Algunos ejemplos son "an", "that" y "the". Aunque "no/not" se encuentran en la lista de palabras vacías de uso común, se han mantenido porque implican una entonación negativa, lo cual puede ser útil en el contexto de la clasificación de publicaciones suicidas.

8. Lematización: Consiste en transformar las palabras derivadas de un verbo, por ejemplo, saltando, saltó en saltar. Haciendo esto se logra igual la reducción de la consistencia semántica, facilita la interpretación del modelo entre otras.

### 3.3 Limpieza de los datos

La limpieza de los datos consiste en la eliminación de palabras irrelevantes, eliminar entradas vacías al igual que las filas con un alto conteo de palabras. Aquí igual se siguió la metodología empleada en (Dus, Y., & Nefedov, G. (2023)).

1. Eliminación de palabras irrelevantes: En plantean (Dus, Y., & Nefedov, G. (2023)) que se realizó exploración de los datos y había palabras como "filler" la cual se repetía en 55442 ocasiones la cual no aportaba ningún significado a los textos y parecía más bien ruido producto de la colección de datos.
2. Eliminación de filas vacías: Después del preprocesado aparecieron algunas filas vacías las cuales fueron eliminadas.
3. Eliminar filas atípicas con alto conteo de palabras: Se obtuvo el conteo de palabras de las publicaciones preprocesadas. Para optimizar el entrenamiento del modelo en etapas posteriores, se decidió continuar con la metodología de (Dus, Y., & Nefedov, G. (2023)) donde los textos con más de 62 palabras fueron eliminados, lo que permite aumentar la velocidad y optimización del entrenamiento.

### 3.4 División de los datos

El conjunto de datos se dividió en dos partes, una para el entrenamiento del modelo y otra para las pruebas, se utilizó el 75 y 25% respectivamente para este propósito. Esta práctica es utilizada con frecuencia en tareas de aprendizaje automático y su finalidad es para llevar a cabo la validación cruzada del modelo. Otros estudios como (Roza, T. H. 2023) y (Rabani, S. T. 2021) han utilizado este conjunto de datos para entrenar clasificadores, en ambos casos se utiliza la división de los datos en entrenamiento y prueba divididos en 75 y 25% respectivamente.

### 3.5 Entrenamiento del modelo.

Una vez cumplidos los pasos anteriores se procede a entrenar los modelos. También en esta etapa se lleva a cabo la optimización de los hiperparámetros.

### 3.6 Validación cruzada.

Básicamente la idea de la validación cruzada es enfrentar al modelo a nueva información y evaluar su capacidad predictiva. Como se plantea en (Berrar, D. (2019)) una cuestión central en el aprendizaje supervisado se refiere a la capacidad de generalización del modelo resultante, lo que en otras palabras se conoce como sobreajuste.

Es muy fácil construir un modelo que se adapte perfectamente al conjunto de datos disponible, pero que luego sea incapaz de generalizar bien a nuevos datos no vistos. Idealmente, se evaluaría el modelo utilizando nuevos datos que provengan de la misma población que los datos que usamos para construir el modelo. En la práctica, sin embargo, nuevos estudios de validación independientes a menudo no son factibles. Además, antes de invertir tiempo y otros recursos en una validación externa, es aconsejable estimar primero el rendimiento predictivo. Esto generalmente se realiza mediante métodos de remuestreo de datos, como la validación cruzada.

### 3.7 Resultados y conclusiones

Como último paso se determinan las conclusiones y se analizan los resultados obtenidos del entrenamiento y la validación cruzada, para ello se analizan las métricas exactitud, precisión, puntuación $F_1$ y sensibilidad.

## 4 Resultados

A continuación, se muestran los resultados obtenidos de entrenar los modelos. Se mencionan algunos datos específicos referentes al conjunto de datos. Se muestran las métricas de cada modelo y la técnica empleada para compensar el desbalance en los datos.

### 4.1 Resultados sobre el conjunto de Datos

Inicialmente el conjunto de datos contaba 232,074 entradas, las cuales quedaron reducidas a 107,066 textos etiquetados como no suicidio y 67,902 casos etiquetados como suicidio, para un total de 174,968 entradas. Esta reducción ocurrió por el preprocesamiento de los datos, como se explica en le sección 3.2 y 3.3.

Como se observa el conjunto de datos se encuentra desbalanceado, para lo cual más adelante se explica cómo se aborda la fase de entrenamiento para el desbalance en los datos.

### 4.2 Resultados experimentales

A continuación, se muestran los resultados experimentales obtenidos al entrenar los modelos utilizando el conjunto de datos antes mencionado. Se analizan las métricas en cada caso.

#### 4.2.1 Resultados con datos no balanceados

A continuación, se muestran las métricas de los modelos entrenados Naive Bayes y GBM sobre los datos de entrenamiento para los datos no balanceados, en todos los casos se utiliza

el GBM con los hiperparámetros optimizados (max_depth = 60, n_estimators = 60, learning_rate = 0.1).

**Tabla 2.** Resultados de evaluación de los modelos.

| Métricas | Naïve Bayes | GBM |
|---|---|---|
| Exactitud | 0.91 | 0.87 |
| Precisión | 0.85 | 0.86 |
| Recall | 0.94 | 0.80 |
| Puntuación $F_1$ | 0.89 | 0.83 |

### 4.2.2 Resultados con datos balanceados (Submuestreo)

El submuestreo consiste en reducir de manera aleatoria la clase predominante del conjunto de datos para que sea similar a la clase con menor cantidad de datos que en este caso es la clase de suicidio con 67,902 entradas. Una vez hecho esto se procede a entrenar los modelos.

**Tabla 3.** Resultados de evaluación de los modelos.

| Métricas | Naïve Bayes | GBM |
|---|---|---|
| Exactitud | 0.91 | 0.87 |
| Precisión | 0.88 | 0.88 |
| Recall | 0.94 | 0.85 |
| Puntuación $F_1$ | 0.91 | 0.86 |

Como se observa en la Tabla 3, los resultados de la evaluación con Naive Bayes prácticamente se mantuvieron igual, aumentando en un 2% la precisión, algo realmente deseable cuando queremos detectar casos de suicidios. Por lo que el submuestreo tuvo un aspecto positivo en este sentido, aunque no muy significativo.

En el caso de GBM, se observa una mejora general del modelo, incluyendo una ligera mejora en la precisión, similar a la obtenida con Naive Bayes. Además, aumentaron los resultados del resto de las métricas de evaluación de las métricas, aunque no de manera significativa.

### 4.2.3 Resultados con datos balanceados (Sobremuestreo)

El sobremuestreo es el caso contrario al submuestreo, consiste en aumentar el tamaño del conjunto de datos de manera aleatoria repitiendo datos que ya existen en este. En la Tabla 4 se muestran las métricas correspondientes para los modelos Naive Bayes y GBM.

**Tabla 4.** Resultados de evaluación de los modelos

| Métricas | Naïve Bayes | GBM |
|---|---|---|
| Exactitud | 0.91 | 0.92 |
| Precisión | 0.89 | 0.92 |
| Recall | 0.95 | 0.93 |
| Puntuación $F_1$ | 0.92 | 0.92 |

De igual manera que en el caso del submuestreo, Naive Bayes no cambió de manera significativa resultados de evaluación, aunque se evidencia una pequeña mejoría.

En contraste, el GBM muestra una mejora general del modelo (ver Tabla 5), con aumentos más significativos en todas las métricas. Siendo este modelo el de mejor rendimiento obtenido.

**Tabla 5.** Métricas de los modelos.

| Métricas | GBM (Datos desbalanceados) | GBM (Submuestreo) | GBM (Sobremuestreo) |
|---|---|---|---|
| Exactitud | 0.87 | 0.87 | 0.92 |
| Precisión | 0.86 | 0.88 | 0.92 |
| Recall | 0.80 | 0.85 | 0.93 |
| Puntuación $F_1$ | 0.86 | 0.86 | 0.92 |

## 5 Discusión

En este estudio, se entrenaron dos clasificadores, Naive Bayes multinomial y GBM, para la detección de indicadores de suicidio en textos. La selección de los algoritmos estudiados estuvo basada en la amplia aplicación de Naive Bayes y el creciente de uso de algoritmos de tipo ensamble como GBM para tareas de clasificación. Durante la revisión de la bibliografía no se encontraron estudios que utilizaran este conjunto de datos con la metodología de preprocesamiento y limpieza de datos empleada en este trabajo.

Se evaluaron varias métricas para cada modelo, incluyendo la exactitud, la precisión, la sensibilidad y la puntuación $F_1$. Estas métricas se evaluaron con respecto a la porción del conjunto de datos destinado a las pruebas y proporcionan una comprensión detallada del rendimiento de los clasificadores en términos de su capacidad para predecir correctamente los casos positivos de suicidio que es objetivo del estudio.

Para el caso del entrenamiento del modelo GBM se ajustaron sus hiperparámetros debido a la importancia y sensibilidad de estos sobre este tipo de algoritmo. Luego se procedió al

entrenamiento de ambos modelos. Primero con los datos desbalanceados, luego se utilizó la técnica de submuestreo y sobremuestreo respectivamente.

Se calcularon las métricas en cada uno de los casos (ver resultados) y se observa una ligera mejora en los datos con las técnicas de submuestreo y sobremuestreo con respecto a los resultados obtenidos cuando los datos estaban desbalanceados, siendo más significativo para GBM que para Naive Bayes. En el caso particular de sobremuestreo donde GBM obtuvo los mejores resultados de las métricas con respecto a los datos de prueba, se observa durante el entrenamiento una tendencia al sobreentrenamiento, lo cual es una posibilidad en los casos donde se utiliza sobremuestreo por la repetitividad inherente de los datos. En sentido general los modelos presentan buenos resultados con 0.92 de puntaje $F_1$ en ambos casos.

Comenzando con la exactitud, ambos modelos mostraron un rendimiento 0.91 y 0.92 para Naive Bayes y GBM respectivamente. Al considerar la precisión, que es una métrica crítica en este contexto, observamos diferencias ligeramente más importantes con Naive Bayes con 0.89, mientras que GBM exhibió una precisión del 0.92. Esta pequeña disparidad indica que Naive Bayes tiende a ser más conservador en sus predicciones, lo que puede resultar en pasar por altos algunos casos de indicio de suicidio en comparación con GBM.

Al examinar la sensibilidad, que mide la capacidad de un modelo para identificar correctamente observamos que Naive Bayes alcanzó un valor del 0.95, mientras que GBM obtuvo 0.93 Indicando una gran paridad en estos casos.

Finalmente, consideramos el puntaje $F_1$, que proporciona una medida equilibrada entre precisión y recall. Naive Bayes y GBM lograron una puntuación de 0.92. Estos resultados indican una gran paridad en la capacidad predictiva de estos modelos al ser entrenados de la manera que se ha descrito en este trabajo.

En la Tabla 6 se muestran las métricas de trabajos relacionados con la clasificación de textos donde se observa que los resultados obtenidos al aplicar la propuesta de solución son satisfactorios y están acorde con los resultados obtenidos en otros estudios. En todos los casos analizados los resultados de este trabajo superan en la puntuación $F_1$ al resto, siendo esta métrica de vital importancia ya que muestra el balance de clasificación entre clases.

**Tabla 6.** Comparación con otros trabajos.

| Referencias | Algoritmos | Exactitud | Precisión | Recall | Puntuación $F_1$ |
|---|---|---|---|---|---|
| Rabani, S. (2021) | Naive Bayes | - | - | - | 0.87 |
| Roza, T. (2023) | Naive Bayes | 0.89 | 0.89 | 0.89 | 0.89 |
| Ao, Z. (2021) | LSTM | 0.92 | Desconocido | Desconocido | Desconocido |
| Este trabajo | Naive Bayes | 0.91 | 0.89 | 0.95 | 0.92 |
| Este trabajo | GBM | 0.92 | 0.92 | 0.93 | 0.92 |

En resumen, Naive Bayes y GBM presentan resultados similares y pueden ser utilizados para predecir textos que indiquen la posibilidad de que una persona pueda cometer suicidio con una alta probabilidad de acierto.

En la Tabla 7 se muestra una comparación de un trabajo donde utilizan la misma metodología de preprocesado y limpieza de datos con modelos de aprendizaje profundo. Se puede constatar que algoritmos menos complejos como Naive Bayes y GBM tienen alta capacidad predictiva en tareas de clasificación cuando son comparados con modelos más complejos como, CNN, LSTM.

**Tabla 7.** Comparación con otros trabajos.

| Referencias | Algoritmos | Exactitud | Precisión | Recall | Puntuación $F_1$ |
|---|---|---|---|---|---|
| Dus, Y., & Nefedov, G. (2023). | CNN | 0.92 | 0.91 | 0.90 | 0.91 |
| Dus, Y., & Nefedov, G. (2023). | LSTM | 0.92 | 0.86 | 0.93 | 0.90 |
| Este trabajo | Naive Bayes | 0.91 | 0.89 | 0.95 | 0.92 |
| Este trabajo | GBM | 0.92 | 0.92 | 0.93 | 0.92 |

## 6 Conclusiones

En este estudio, se evaluaron dos clasificadores, Naive Bayes y GBM, para la detección de suicidio en textos. Ambos modelos demostraron un desempeño prometedor en la tarea de identificar posibles casos de suicidio, con resultados que destacan sus respectivas fortalezas y debilidades.

Ambos modelos mostraron competitividad en sus métricas siendo ligeramente superior el modelo de GBM con respecto a Naive Bayes utilizando sobremuestreo para compensar el desbalance en los datos. Naive Bayes no mostró cambios significativos con respecto a la utilización de los datos desbalanceados y las técnicas empleadas de balanceo de datos. También se evidenció la importancia del ajuste de los hiperparámetros en los algoritmos de ensamble como GBM.

Un aspecto importante en los trabajos de clasificación de textos es que necesitan un conjunto de datos etiquetados además de que la obtención de textos apropiados tampoco es una tarea en muchos casos fácil de obtener. En muchos casos están presente los temas relacionados con la privacidad y el consentimiento de las partes o de organizaciones que manejan este tipo de información.

Como trabajo futuro se propone la evaluación de este mismo enfoque con otros modelos que han sido utilizados para el análisis de sentimientos como Bosques Aleatorios, Maquinas de Soporte Vectorial y Redes Neuronales LSTM. También la mejora en las técnicas de procesamiento de lenguaje natural supone un reto para los problemas de clasificación de texto, pues la calidad de los datos influye directamente sobre los resultados y la capacidad de predicción de los modelos. La metodología de preprocesamiento y limpieza de datos empleada eliminó una serie de entradas que fueron consideradas irrelevantes, habría que analizar en profundidad la relevancia de los datos eliminados. A modo general los

resultados obtenidos fueron satisfactorios con métricas acorde a los estudios obtenidos en problemas de clasificación de textos y detección de suicidio.

## Referencias


Aldhyani, T.H., Alsubari, S.N., Alshebami, A.S., Alkahtani, H., Ahmed, Z.A. (2022). "Detecting and analyzing suicidal ideation on social media using deep learning and machine learning models". *International journal of environmental research and public health* 19(19), 12635

Ao, Z., Lai, J., Lu, W., Mo, H. (2021). "Comparisons of LSTM and other Machine Learning Approaches on Predicting Suicidal Tendency Regarding to Social Media Posts". In: *2021 International Conference on Computers and Automation (CompAuto)*, pp. 19-23. IEEE,

Ballester, P.L., Cardoso, T.D.A., Moreira, F.P., da Silva, R.A., Mondin, T.C., Araujo, R.M., de Mattos Souza, L.D. (2021). "5-year incidence of suicide-risk in youth: A gradient tree boosting and SHAP study*". Journal of affective disorders* 295, 1049-1056

Basyouni, A., Abdelkader, H., Ali, A. (2023). "A New Approach to Suicide Ideation Detection from Text Content". IJCI. *International Journal of Computers and Information* 10(2), 1-15

Berrar, D. (2019). "Cross-validation". *Machine Learning Research Group School of Mathematics and Statistics. The Open University, Milton Keynes, United Kingdom* 542-545

Desmet, B., Hoste, V. (2018). "Online suicide prevention through optimised text classification". *Information Sciences* 439, 61-78

Dus, Y., & Nefedov, G. (2023). "An Automated Tool to Detect Suicidal Susceptibility from Social Media Posts". *arXiv preprint arXiv:2310.06056*.

Karhik, S. (2006). Gradient Boosting: Working and Applications. Medium. https://medium.com/gradient-boosting-working-limitations-time/gradient-boosting-working-and-applications-28e8d4ba866d. Página de internet.

Murphy, K.P. (2006). "Naive bayes classifiers". *University of British Columbia* 18(60), 1-8

Natekin, A., Knoll, A. (2013). "Gradient boosting machines, a tutorial". *Frontiers in neurorobotics* 7, 21

Paz, L.H.B., de León Rodrıguez, I.D., Enciso, F.A.A. (2023). "Identifying Self-Harm Risk Subjects on Social Media".

Rai, P., Prasun, K., Sharma, G., y Bhattarai, Y. (2024). "Comparison of naïve bayes, logistic regression and support vector machine for predicting suicidal tendency from social media content". International Journal of Research Publications 145(1), 18-18.

Rabani, S.T., Khan, Q.R., Khanday, A.M.U.D. (2021). "Quantifying suicidal ideation on social media using machine learning: A critical review". *Iraqi Journal of Science*, 4092-4100

Roza, T.H., de Souza Seibel, G., Recamonde-Mendoza, M., Lotufo, P.A., Benseñor, I.M., Passos, I.C., Brunoni, A.R. (2023). "Suicide risk classification with machine learning techniques in a large Brazilian community sample". *Psychiatry research* 325, 115258

K., Sharma, G., y Bhattarai, Y. (2024). "Comparison of naïve bayes, logistic regression and support vector machine for predicting suicidal tendency from social media content". *International Journal of Research Publications* 145(1), 18-18.

Sanderson, M., Bulloch, A.G., Wang, J., Williamson, T., Patten, S.B. (2020). "Predicting death by suicide using administrative health care system data: can recurrent neural network, one-dimensional convolutional neural network, and gradient boosted trees models improve prediction performance?". *Journal of affective disorders* 264, 107-114

Sariyildiz, Y. (2022). “Gradient Boosting Machines Tutorial”. Kaggle. https://www.kaggle.com/code/yasinnsariyildiz/gradient-boosting-machines-tutorial. Página de internet

Sharma, A. (2020). Naive Bayes with Hyperparameter Tuning. Kaggle. https://www.kaggle.com/code/akshaysharma001/naive-bayes-with-hyperpameter-tuning. Página de internet

Spasić, I., Burnap, P., Greenwood, M., y Arribas-Ayllon, M. (2012). “A naïve bayes approach to classifying topics in suicide notes”. *Biomedical informatics insights*, 5, BII-S8945.

# Capítulo 2

## Calibración de hiper-parámetros en algoritmos metaheurísticos para la detección de fraude digital

Gabriel Hurtado Avilés[1], Román Anselmo Mora Gutiérrez[1], José Alejandro Reyes Ortiz[1]

[1] Universidad Autónoma Metropolitana
Unidad Azcapotzalco
Maestría en Ciencias de la Computación

al2232800343@azc.uam.mx, mgra@azc.uam.mx, jaro@azc.uam.mx

**Resumen.** Este estudio se centra en la calibración de hiperparámetros en *algoritmos metaheurísticos* para la detección de fraude digital. Se comparan distintos algoritmos incluyendo *Recocido Multiarranque (MSA), Búsqueda Dispersa (SS), Búsqueda en Vecindades Variables (VNS), Algoritmo Genético (GA)* y *Optimización por Enjambre de Partículas (PSO),* con el objetivo de detectar fraude digital. Se utilizan tres corpus en español para evaluar su efectividad en la identificación de noticias falsas. Además, se propone construir un nuevo corpus específico para experimentar herramientas más efectivas para proteger a las personas más vulnerables a los efectos del fraude digital como los adultos mayores y aquellos con menor acceso a la educación y la información.

**Palabras Clave:** *noticias falsas, fraude digital, algoritmos metaheurísticos.*

## 1 Introducción

A pesar de los avances en la detección de noticias falsas en inglés, la detección de noticias falsas en español es todavía un desafío debido a la falta de datos etiquetados en español. Sin embargo, se han hecho esfuerzos para superar este desafío. Uno de ellos es el desarrollo de recursos para analizar y detectar información engañosa en sitios web de noticias en español *(Posadas-Durán et al., 2019).* Estos recursos son corpus de noticias en español anotadas con etiquetas (real y falso) para la detección automática de noticias falsas.

La detección de noticias falsas, también conocidas como *fake news* ha sido abordada desde diversos ángulos incluyendo la detección basada en estilometría, donde se propuso un enfoque basado en el *Procesamiento de Lenguaje Natural (NLP)* y el *Reconocimiento de Entidades Nombradas (NER) (Tsai, 2023).* Otro enfoque se centra en la modelización y resolución de problemas, planteando la detección de fake *news* como un problema complejo que requiere una planificación meticulosa y programación eficaz *(Aqil & Lahby, 2021).*

Las redes sociales también han sido objeto de estudio en el contexto de las noticias falsas, por ejemplo, el análisis del impacto de las noticias políticas falsas en Facebook durante las elecciones presidenciales de 2016 en los Estados Unidos *(Ali y Zain-Ul-Abdin, 2020).* Este estudio evidenció su rápida propagación y cómo estas pueden tener graves consecuencias en la sociedad, ya que pueden distorsionar la realidad, manipular la opinión pública, incitar a la violencia, difundir propaganda política, fomentar el odio y provocar pánico.

Más allá de las noticias falsas, la detección de fraudes laborales también ha cobrado relevancia. Se han propuesto soluciones basadas en aprendizaje automático, como el uso de *Redes Neuronales Artificiales (ANN) (Nasser et al., 2021).* Este enfoque se basa en la idea de que ciertos patrones en los datos de reclutamiento pueden indicar la presencia de fraude.

El fraude digital no solo causa pérdidas económicas a las víctimas, sino que también puede tener graves consecuencias emocionales y psicológicas. en casos extremos, el fraude digital puede tener consecuencias fatales, como en el caso de personas que han sido estafadas con ofertas de trabajo falsas y han terminado siendo víctimas de violencia o desaparición. Por lo tanto, la motivación para llevar a cabo esta investigación se fundamenta en proteger a las personas más vulnerables, aquellas que carecen de las herramientas y el conocimiento necesarios para discernir entre la información legítima de la fraudulenta, lo que las convierte en blancos fáciles de estafadores.

Para elaborar el estudio, los autores trabajaron en conjunto en todo momento, sobre todo, en la revisión del estado del arte. Ahora bien, cabe resaltar que el tercer autor fue quien identificó la problemática relacionada con la propagación e identificación de noticias falsas en español. El segundo autor se encargó de proponer y supervisar la implementación de algoritmos metaheurísticos para abordar la problemática, mientras que el primer autor identificó que la problemática podría ser aprovechada para incluir diferentes tipos de fraude digital y llevó a cabo la implementación de la metodología final propuesta.

El estudio se organiza en varias secciones. Comienza con una introducción que presenta el problema de la detección de noticias falsas y la importancia de calibrar hiper-parámetros en algoritmos metaheurísticos. Luego, la revisión de literatura analiza estudios previos y técnicas utilizadas en este campo. La sección de metodología describe los algoritmos empleados y el proceso de selección de los corpus de datos. A continuación, se presentan los resultados, donde se comparan la efectividad de los diferentes algoritmos implementados. Se interpretan los hallazgos y sus implicaciones. Para finalizar, el estudio concluye con una sección de conclusiones que resume los hallazgos y ofrece recomendaciones, seguidas de recomendaciones para un trabajo futuro que propone mejoras y necesidades para investigaciones futuras. Finalmente, se listan las referencias citadas.

## 2 Revisión del Estado del Arte

La revisión de la literatura muestra que la detección precisa de noticias falsas sigue siendo un desafío complejo en el que se deben considerar las limitaciones actuales al interpretar los resultados de las predicciones. Diversos investigadores han dedicado un gran

esfuerzo al desarrollo de métodos automáticos para la detección de noticias falsas, como el análisis de redes sociales *(Pulido et al., 2020)*, el aprendizaje profundo *(Hu et al., 2022)* o el Procesamiento del Lenguaje Natural. Estas propuestas se complementan con revisiones del estado del arte que resumen los avances en la detección de noticias falsas, analizando diferentes técnicas y enfoques *(Bondielli & Marcelloni, 2019 y Mridha et al., 2021)*

Se han creado iniciativas para poner en práctica estos avances, como la creación de *corpus* de noticias en español para el análisis y detección de información engañosa *(Posadas-Durán et al., 2019)* y la exploración del uso de técnicas de aprendizaje profundo para la detección de noticias falsas en este idioma *(Martínez-Gallego et al., 2021)*. Entre las técnicas más comunes se tienen las que utilizan el *PLN* (ver Tabla 1).

Tabla. 1. Investigaciones enfocadas en el *Procesamiento del Lenguaje Natural*.

| **Contribuciones Principales.** ***Referencia*** **(Idioma) [Tema]** | **Dataset** |
|---|---|
| Propone un método basado en el estilo de escritura para detectar noticias falsas en español. *(Posadas-Durán et al., 2019)* (ESP) [Fake News] | SpanishFakeNewsCorpus, proveniente de https://github.com/jpposadas/FakeNewsCorpusSpanish |
| Evalúa diferentes arquitecturas de *Deep Learning* para la detección de noticias falsas en español. *(Martínez-Gallego et al., 2021)* (ESP) [Fake News] | Se utiliza información de https://www.kaggle.com/clmentbisaillon/fake-and-real-news-dataset |
| Combina *NLP* y *NER* para detectar noticias falsas en inglés, utilizando datasets como *LIAR* y *FakeNews AMT*. *(Tsai, 2023)* (INGL) [Fake News] | https://www.kaggle.com/mrisdal/fake-news y el corpus LIAR (https://www.cs.ucsb.edu/~william/data/liar_dataset.zip |

Por otro lado, *Aqil & Lahby, 2021* proponen un enfoque para el proceso de detección de noticias falsas, modelándolo como un problema de programación de flujo de taller, también conocido como *Problema de Secuenciación de Tareas en Taller (Job Shop Scheduling Problem)*. Los autores presentan tres *metaheurísticas*: el *algoritmo codicioso iterado IG (Iterated Greedy)*, el *algoritmo genético GA (Genetic Algorithm)* y el *algoritmo de colonia de abejas artificiales ABC (Artificial Bee Colony)*, todos diseñados para minimizar el tiempo de finalización máximo. Según los resultados de la simulación, el *algoritmo IG* supera en rendimiento a los otros algoritmos.

Tabla. 2. Análisis de las principales técnicas para la detección de noticias falsas y fraude laboral

| **Contribuciones Principales** ***Referencia*** **(Idioma) [Tema]** | **Dataset** |
|---|---|
| **TF-IDF** | |
| Utiliza *TF-IDF* para representación textual en un modelo de *Deep Learning* para detectar fake news. *(Thota et al., 2018)* (INGL) [Fake News] | Recopila datos de http://www.fakenewschallenge.org/ |
| **Modelos de Lenguaje** | |
| Evalúa la detección de noticias falsas con *modelos de lenguaje* mejorados con conocimiento. (Whitehouse et al., 2022) (INGL) [Fake News] | Utiliza el corpus LIAR, y datos provenientes de https://newschecker.in |

| | | |
|---|---|---|
| | *Método heurístico, Máquina de Vectores de Soporte* optimizada con *Algoritmo Genético*. *(Hidayattullah et al., 2020)* (INGL) | Menciona la colaboración con firmas de contabilidad en Indonesia para etiquetar los datos en diferentes categorías. |

Además de las *metaheurísticas*, existen diferentes técnicas (ver Tabla 2) que han demostrado ser prometedoras para la detección de noticias falsas y rumores en el contexto de la comunicación mediada por computadora, abarcando desde el aprendizaje supervisado hasta el no supervisado y semi-supervisado *(Bondielli & Marcelloni, 2019).* Estas técnicas a menudo se combinan con el *Procesamiento de Lenguaje Natural*, el *preprocesamiento de datos*, la *vectorización de datos* y la *extracción de características (Thota et al., 2018, Martínez-Gallego et al., 2021; Mridha et al., 2021)* para mejorar la detección de noticias falsas. Un ejemplo de la aplicación de estas técnicas es la *metodología SISM (Social Impact in Social Media)* propuesta por *Pulido et al. (2020),* que es una de las técnicas propuestas para abordar la proliferación de noticias falsas en salud en las redes sociales.

El impacto de las noticias falsas se extiende a diversos ámbitos, desde los medios de comunicación *(Cárcamo-Ulloa et al., 2020) [9] y Dasilva et al., 2020)* hasta el fraude laboral *(Nasser et al., 2021 y Álvarez, 2021),* las redes sociales *(Singh et al., 2023 [5])* y el fraude financiero. El fraude financiero es una problemática que también se ve afectada por la desinformación, particularmente, *Hidayattullah et al., 2020* se centran en la detección de fraudes en los estados financieros de las empresas cotizadas en Indonesia, utilizando técnicas de aprendizaje automático y optimización *metaheurística* para desarrollar modelos de predicción de fraudes El fraude laboral es otro ámbito donde las noticias falsas pueden tener un impacto significativo. *Nasser et al. (2021)* presentan un modelo basado en Redes Neuronales Artificiales para detectar publicaciones de trabajo fraudulentas, mientras que *Álvarez, 2021* aborda el fraude laboral en la era digital, destacando su evasión de costos y las nuevas modalidades en plataformas digitales.

Los *métodos heurísticos* ofrecen una alternativa para la detección de noticias falsas, particularmente en escenarios donde la complejidad del problema o la falta de datos dificulta la tarea. Por ejemplo, se propuso un marco de ensamblaje basado en incertidumbre impulsado por *heurísticas* para la detección de noticias falsas en tweets y artículos de noticias *(Das et al., 2022)*, mientras que *Ali & Zain-Ul-Abdin (2021),* analizaron el impacto de las noticias políticas falsas en Facebook durante las elecciones presidenciales de 2016 en los Estados Unidos. Este estudio describe el procesamiento heurístico de las noticias falsas políticas en Facebook durante las elecciones presidenciales de EE. UU. de 2016 y demuestra se pueden utilizar diversas técnicas para abordar la problemática de la detección, particularmente, utilizando *métodos heurísticos* como se muestra en la Tabla 3.

Tabla. 3. Investigaciones que hacen uso de *métodos heurísticos.*

| **Contribuciones Principales** ***Referencia*** **(Idioma) [Tema]** | **Dataset** |
|---|---|
| Utiliza *modelos de lenguaje* preentrenados, *BERT* seguidos de una red de fusión de características estadísticas, utiliza *algoritmos heurísticos* e incorpora atributos presentes en las noticias como fuente, manejadores de usuario, dominios *URL* y autores como característica estadística. | 1. https://www.eonline.com/ap y 2. https://competitions.codalab.org/competitions/26655 |

| | |
|---|---|
| *(Das et al., 2022)* (INGL) [Fake News] | |
| Hace un Análisis Textual Cualitativo que se centra en descubrir significados latentes y patrones implícitos en el texto, prestando atención a las suposiciones culturales e ideológicas subyacentes.<br>*(Ali & Zain-Ul-Abdin, 2021)* (INGL) [Fake News] | Indica que se recopilaron datos de Facebook y Twitter durante las elecciones presidenciales de Estados Unidos de 2016. |
| Utiliza *metaheurísticas*, específicamente tres algoritmos: *Iterated Greedy (IG), Genetic Algorithm (GA)* y *Artificial Bee Colony (ABC)*, para minimizar el tiempo de finalización máximo de un conjunto de documentos, conocido como *makespan*.<br>Referencia 20 (INGL) [Fake News] | No menciona un dataset específico, solo indica que se utilizaron datos simulados y datos de Facebbok y Twitter. |

Los *algoritmos metaheurísticos* se han utilizado para calibrar los *hiperparámetros* de los modelos de aprendizaje automático, demostrando ser efectivos en la mejora de la detección *(Aqil & Lahby, 2021; Yazdi et al., 2020).*

## 3 Enfoque Metaheurístico (metodología)

La metodología propuesta se estructura en cinco etapas. La primera etapa consistió en la detección de la problemática que se describe a detalle en la revisión del estado del arte y se puede concluir que si bien, la *Recuperación de Información (RI),* los *modelos de lenguaje* y los *algoritmos metaheurísticos* ofrecen un potencial significativo para la detección de fraude digital, también es importante reconocer sus limitaciones, en especial, en el idioma español debido a que es la segunda lengua más hablada del mundo, con más de 500 millones de hablantes nativos, lo que lo convierte en un blanco atractivo para los perpetradores de fraude digital.

La segunda etapa consistió en la selección y limpieza de los datos, en esta etapa, la *RI* es fundamental porque se centra en el desarrollo de métodos y herramientas para buscar y recuperar información relevante de grandes volúmenes de datos no estructurados o semiestructurados. En este sentido, la "*Conference and Labs of the Evaluation Forum" (CLEF Initiative, 2024)* juega un papel crucial al ofrecer un espacio para la evaluación y comparación de sistemas de *PLN* y *RI* enfocados en la detección de noticias falsas. Su objetivo es promover la investigación y el desarrollo en estas áreas mediante la creación de conjuntos de datos, tareas y evaluaciones. Desde su creación en el año 2000, *CLEF* ha organizado talleres y conferencias que han atraído a investigadores de todo el mundo, impulsando avances significativos en el campo del *PLN* y la *RI*

Si bien *CLEF* inicialmente se centró en el inglés, la iniciativa ha evolucionado hacia un enfoque multilingüe, reconociendo la importancia de evaluar el rendimiento de los sistemas en diferentes idiomas. Dentro de las diversas áreas de enfoque de *CLEF*, destaca la iniciativa *CLEF CheckTha*t que proporciona un conjunto de datos y tareas que permiten a los investigadores desarrollar y probar sus métodos. Además, a partir del año 2020, *CLEF* comenzó a incluir tareas y conjuntos de datos en español, permitiendo a los investigadores evaluar el rendimiento de sus sistemas en un contexto lingüístico más amplio.

La incorporación del español a las evaluaciones de *CLEF* abre nuevas oportunidades para la investigación y el desarrollo de sistemas de *PLN y RI* en este idioma. Ahora bien, aunque la iniciativa *CLEF* ofrece un valioso recurso para la investigación en detección de

noticias falsas y fraude digital en general, también presenta limitaciones debido a la falta de etiquetado en verdadero o falso y en cuanto a la determinación de la veracidad absoluta de las noticias.

Para superar estas limitaciones, se optó por utilizar el corpus presentado por *Posadas-Durán et al. (2019)*. Este corpus se utilizó para la tarea de detección de *fake news* en español en el congreso *IberLEF (Iberian Languages Evaluation Forum)* 2021, donde, de acuerdo a los resultados de la competición*(MexLef, 2021)*, un equipo del CIC (Centro de Investigación en Computación) del IPN (Instituto Politécnico Nacional) obtuvo un Valor-F ($F_1$ *score*) de 0.6542, mientras que el equipo con el mayor puntaje obtuvo un Valor-F de 0.7666 Los detalles del concurso se pueden consultar en la página principal del concurso y el corpus se puede descargar en el repositorio de *Github (Ramírez Cruz, J. M. et al., 2021).*

Además del corpus que presenta *Posadas-Durán et al., 2019*, la metodología se enriquece con otros conjuntos de datos disponibles en *Kaggle (Manzano-Velasco, C. E. et al., 2024 & Zules Acosta, F. A., 2019)*. Particularmente, el segundo conjunto llamado "*Spanish Fake and Real News*" *(Manzano-Velasco, C. E. et al., 2024)* fue creado durante la elaboración de un trabajo de fin de máster en ciberseguridad de la Universidad Politécnica de Madrid y se utiliza en diferentes estudios. Su autor es Fabricio A. Zules y la creación del corpus fue posible gracias al apoyo de la Secretaría de Educación Superior, Ciencia, Tecnología e Innovación del Ecuador.

El tercer y último corpus *(Zules Acosta, F. A., 2019)* fue extraído de la página web colombiacheck.com para fines de investigación y contiene noticias verificadas por periodistas colombianos, su autor es Enrique Manzano, un desarrollador Full Stack que reside en Popayán, Cauca, Colombia. A pesar de que Colombiacheck es un medio de comunicación dedicado a verificar y validar numerosas noticias, no todas son verificadas estrictamente como verdaderas o falsas similarmente al caso de CLEF. Algunas pueden contener elementos de ambas categorías o estar sujetas a interpretación. En resumen, para simplificar la tarea de clasificación de noticias en verdaderas o falsas en base al corpus de Colombiacheck, se re etiquetaron las categorías originales (Verdadero, Falso, Cuestionable, etc.) para abordar la detección de veracidad como un problema binario, facilitando la tarea de clasificación.

La segunda etapa concluye al construir una *bolsa de palabras (BoW)*, producto del preprocesamiento de los datos de texto con técnicas de *tokenización, etiquetado de la parte del discurso, eliminación de palabras de parada, lematización, eliminación de puntuación y filtrado de palabras por longitud* para el análisis posterior. Las técnicas empleadas en esta etapa se basan en métodos estándar de preprocesamiento de texto que se encuentran ampliamente documentadas en el campo del *PLN*. La creación de una *BoW* es una técnica clásica para representar documentos como vectores numéricos. Si bien la *BoW* es simple y efectiva, existen técnicas más avanzadas como *TF-IDF (Term Frequency-Inverse Document Frequency)* que pueden ponderar la importancia de las palabras en función de su frecuencia en el documento y su rareza en el *conjunto de datos (Thota et al., 2018).* Implementar *TF-IDF* en lugar de *BoW* es una buena alternativa para realizar la representación vectorial del texto.

La tercera etapa se realizó implementando cinco *algoritmos metaheurísticos: VNS, PSO, MSA, SS y GA*. La descripción y el seudocódigo para implementar los métodos se describe en la Tabla 4:

Tabla. 4. Algoritmos propuestos en el enfoque *metaheurístico*

| Metaheurística | Pseudocodigo |
|---|---|
| **Búsqueda en Vecindades Variables (VNS)**<br>*La Búsqueda en Vecindarios Variables (VNS)* es un método *metaheurístico* iterativo que explora el espacio de búsqueda mediante la exploración de diferentes vecindarios de soluciones. En cada iteración, se selecciona una solución actual y se exploran sus vecindarios para encontrar una mejor solución. La estrategia de selección de vecindarios se modifica iterativamente para evitar caer en ciclos locales y explorar nuevas regiones del espacio de búsqueda. | Inicio<br>Leer parámetros (TI, TF, alpha, pasos, puntos, vecindades, %_seleccion, lineas)<br>Cargar datos (Problema, y) y Seleccionar características (Problema_reducido)<br>Inicializar soluciones (sol_inial, sol_inial_p, peso, claseFinal)<br>Función evaluar<br>Establecer parámetros (TA, c32, total_pasos, progreso)<br>Bucle principal (TA > TF)<br>Para cada punto (k)<br>Evaluar solución actual (Eval_sol_punto)<br>Para cada vecindario (v)<br>Para cada paso (p)<br>c32 += 1<br>progreso += 1<br>Si progreso % 1000 == 0<br>Imprimir progreso<br>Seleccionar índice aleatorio (idx)<br>Crear solución vecina (sol_b1, sol_b2, sol_b3)<br>Evaluar solución vecina (Eval_sol_b)<br>Calcular delta_E<br>Calcular probabilidad de aceptación (metropoli)<br>Si metropoli > random.random()<br>Actualizar solución actual<br>Actualizar Eval_sol_punto<br>Reducir temperatura (TA *= alpha)<br>Fin<br>Seleccionar mejor solución (mejor_sol)<br>Evaluar mejor solución (claseFinal)<br>Guardar resultado |
| **Optimización por Enjambre de Partículas (PSO)**<br>*La Optimización por Enjambre de Partículas (PSO)* es un método *metaheurístico* inspirado en el comportamiento de los enjambres, por ejemplo, enjambres de abejas, aves o peces. En este algoritmo, cada partícula representa una posible solución al problema y se mueve por el espacio de búsqueda en función de su posición actual y la mejor posición encontrada por sí misma o por el enjambre. | Inicio<br>Leer parámetros (b, N, porcentaje_seleccion, num_lineas)<br>Cargar datos (Problema, y) & Seleccionar características (Problema_reducido)<br>Función evaluar<br>Inicializar enjambre (swarm)<br>Crear partícula con posición aleatoria (particle)<br>Crear velocidad aleatoria (velocity)<br>Evaluar partícula (fitness)<br>Agregar partícula al enjambre<br>Inicializar mejor global (global_best)<br>Bucle principal (iteraciones N)<br>Actualizar velocidades<br>Para cada partícula (particle, velocity, fitness)<br>Calcular nuevos coeficientes (r1, r2)<br>Actualizar velocidad (considerando mejor global y partícula)<br>Actualizar partícula en el enjambre<br>Actualizar posiciones<br>Para cada partícula (particle, velocity, fitness)<br>Actualizar posición (sumando velocidad)<br>Evaluar nueva posición (fitness)<br>Actualizar partícula en el enjambre<br>Actualizar mejor global (si hay mejor partícula)<br>Fin<br>Seleccionar mejor solución (best_solution)<br>Guardar resultado (claseFinal)<br>Fin |
| **Recocido Multiarranque (MSA)**<br>*El recocido simulado multiarranque (MSA)* es una *metaheurística* que, en lugar de realizar una sola búsqueda con *SA*, ejecuta múltiples instancias de *SA* desde diferentes puntos iniciales (arranques) y se combinan los resultados, lo que puede mejorar significativamente la calidad de las soluciones encontradas. Esta técnica se basa en la analogía del proceso de recocido de metales, donde un material se calienta a una alta temperatura y luego se enfría | Inicio<br>Cargar datos (Problema, y)<br>Reducción de dimensionalidad (Problema_reducido)<br>Inicializar soluciones (sol_inial, sol_inial_p, peso)<br>Función de evaluación (evaluar)<br>Establecer hiperparámetros (TI, TF, alpha, pasos, puntos, vecinos)<br>Bucle principal (temperatura TA > TF)<br>Para cada punto (k)<br>Establecer Eval_sol_punto a None<br>Bucle sobre pasos en cada temperatura<br>Generar solución vecina (sol_b1, sol_b2, sol_b3)<br>Evaluar solución vecina (clase_b, Eval_sol_b)<br>Calcular diferencia de energía (delta_E)<br>Aplicar criterio de Metropolis<br>Actualizar solución si se cumple la condición<br>Actualizar Eval_sol_punto<br>Fin del bucle sobre pasos<br>Seleccionar mejor solución en el punto actual |

| | |
|---|---|
| lentamente para obtener una estructura cristalina más estable. | Fin del bucle sobre puntos<br>Enfriar la temperatura (TA *= alpha)<br>Fin<br>Seleccionar mejor solución global<br>Guardar resultados (claseFinal)<br>Fin |
| **Búsqueda Dispersa (SS)**<br>*Scatter Search (SS)* es un método *metaheurístico* basado en la población, diseñado para resolver problemas de optimización complejos. Fue introducido por Fred Glover y Manuel Laguna en la década de 1970 y se ha convertido en una herramienta popular para abordar una amplia gama de problemas en diversas áreas.<br>A diferencia de los algoritmos evolutivos tradicionales como los algoritmos genéticos, SS se basa en la combinación de soluciones existentes en lugar de la mutación y recombinación aleatorias. | Inicio<br>Leer parámetros (b, N, porcentaje_seleccion, num_lineas)<br>Cargar datos (Problema, y)<br>Seleccionar características (Problema_reducido)<br>Inicializar P (conjunto vacío)<br>Función evaluar<br>Generar soluciones (aleatorias)<br>Evaluar y agregar a P<br>Construir RefSet (b mejores soluciones de P)<br>Evaluar RefSet<br>Scatter Search (N iteraciones)<br>NuevaSolucion = False<br>Generar subconjuntos aleatorios de RefSet<br>Mientras subconjuntos no esté vacío<br>Seleccionar un subconjunto aleatorio<br>Descomponer en (sol_inial1, sol_inial2)<br>Combinar soluciones (sol_inial_comb)<br>Mejorar solución combinada<br>Si Eval_sol_comb > mejor solución en RefSet<br>Actualizar mejor solución en RefSet<br>Ordenar RefSet por Eval_sol_punto<br>NuevaSolucion = True<br>N -= 1<br>Seleccionar mejor solución (mejor_sol)<br>Evaluar mejor solución (claseFinal)<br>Guardar resultado |
| **Algoritmo Genético (GA)**<br>*El Algoritmo Genético (GA)* es un método *metaheurístico* inspirado en la evolución biológica. En este algoritmo, se mantiene una población de soluciones, denominadas cromosomas, que representan posibles soluciones al problema. Las soluciones se combinan (cruce) y se modifican (mutación) para generar nuevas soluciones. Las soluciones con mayor "aptitud" (evaluadas por la función objetivo) tienen más probabilidades de sobrevivir a la siguiente generación. | Inicio<br>Leer parámetros (b, N, porcentaje_seleccion, num_lineas)<br>Cargar datos (Problema, y)<br>Seleccionar características (Problema_reducido)<br>Definir función de evaluación (evaluar)<br>Definir límites de parámetros (bounds)<br>Algoritmo Genético (differential_evolution)<br>Establecer función de evaluación (evaluar)<br>Establecer límites de parámetros (bounds)<br>Establecer tamaño de población (popsize)<br>Establecer número máximo de iteraciones (maxiter)<br>Establecer tolerancia (tol)<br>Ejecutar algoritmo<br>Obtener solución (result.x)<br>Seleccionar mejor solución (mejor_sol)<br>Evaluar mejor solución (claseFinal)<br>Guardar resultado (claseFinal)<br>Fin |

Finalmente, la cuarta y quinta etapa consiste en la selección y el ajuste de *parámetros* e *hiperparámetros* en las metaheurísticas implementadas, la elección de estos valores debe considerarse cuidadosamente en función de las características del problema específico y los recursos computacionales disponibles. En conclusión, la metodología propuesta, se puede resumir en la Figura 1.

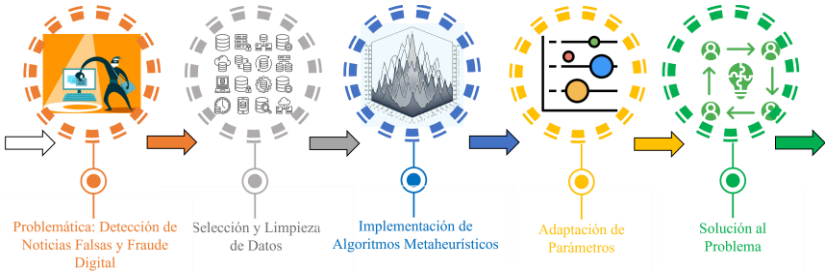


Figura 1. Metodología propuesta para abordar la problemática.

# 4 Evaluación y Análisis de los Resultados

Los resultados de un problema de clasificación binaria, se suelen representar comúnmente en una *matriz de confusión*. La matriz resultante del enfoque metaheurístico refleja cada tipo de predicción que el modelo hizo correctamente o incorrectamente.

En la clasificación binaria, si la clase positiva es 1 y la negativa 0, la matriz debería interpretarse formalmente como:

$$\begin{bmatrix} TP & FN \\ FP & TN \end{bmatrix}, \text{ donde:}$$

*TP* es el número de casos positivos que el modelo ha predicho correctamente como positivos (Verdaderos Positivos), *FN* es el número de casos positivos que el modelo ha predicho incorrectamente como negativos (Falsos Negativos), *FP* es el número de casos negativos que el modelo ha predicho incorrectamente como positivos (Falsos Positivos) y *TN* es el número de casos negativos que el modelo ha predicho correctamente como negativos (Verdaderos Negativos). Es importante minimizar los falsos positivos porque pueden llevar a tomar decisiones erróneas basadas en información incorrecta.

Es por eso que, para evaluar el rendimiento del enfoque metaheurístico propuesto, se emplean diferentes métricas que permiten cuantificar la efectividad de los algoritmos en los corpus descritos en la primera etapa del enfoque (ver Tabla 5).

Tabla 5. Lista de los corpus utilizados para abordar la problemática

| *Número de corpus* | *#1:* | *#2:* | *#3:* |
|---|---|---|---|
| *Nombre del corpus* | *"Spanish Fake News Corpus V. 2.0"* | *"Spanish Fake and Real News"* | *"ColombiaCheck*[3] |

En resumidas cuentas, la *exactitud (accuracy)* es una medida general del rendimiento que considera todos los verdaderos positivos y verdaderos negativos, pero no distingue entre los tipos de errores (falsos positivos y falsos negativos).

Por otro lado, la *precisión (precision)* mide cuántas de las instancias clasificadas como positivas son realmente positivas, mientras que el *recall (*también conocido como el *'ratio de verdaderos positivos')* mide cuántas de las instancias realmente positivas fueron identificadas correctamente como positivas por el modelo. Un *recall* alto significa que el modelo es bueno para detectar los verdaderos positivos, pero no necesariamente implica que haya muchos falsos positivos.

Si un modelo tiene un *recall* alto pero una precisión baja, significa que está identificando correctamente muchas instancias positivas, pero también está cometiendo muchos errores al clasificar las negativas como positivas. Los falsos positivos están más relacionados con la métrica de precisión que con el *recall,* y por ello, la clave está en encontrar un buen equilibrio entre *precisión* y *recall,* lo cual a menudo se hace mirando el puntaje $F_1$. El puntaje $F_1$ es útil porque combina la *precisión* y el *recall* en una sola métrica que puede proporcionar una visión más equilibrada (ver Tabla 6):

Tabla 6. Resultados obtenidos al implementar y evaluar los *algoritmos metaheurísticos.*

| **Metaheurística** ↓ | **Exactitud** ↓ $\frac{TP+TN}{TP+TN+FP+FN}$ | | | **Precision** ↓ $\frac{TP}{TP+FP}$ | | | **Recall** ↓ $\frac{TP}{TP+FN}$ | | | **$F_1$** ↓ $\frac{2*precision*recall}{precision+recall}$ | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| corpus | #1 | #2 | #3 | #1 | #2 | #3 | #1 | #2 | | #1 | #2 | #3 |
| Recocido Multiarranque (MSA) | .52 | .66 | .13 | .51 | .64 | .08 | .88 | .89 | .86 | .65 | .74 | .15 |
| Búsqueda Dispersa (SS) | .67 | .77 | .90 | .72 | .78 | .55 | .56 | .82 | .14 | .63 | **.80** | .23 |
| Algoritmo Genético (GA) | .63 | .73 | .89 | .65 | .71 | .37 | .58 | .88 | .16 | .61 | **.79** | .22 |
| Búsqueda en Vecindades Variables (VNS) | .63 | .55 | .90 | .72 | .56 | .44 | .42 | .97 | .01 | .53 | .71 | .02 |
| Enjambre de Partículas (PSO). | .54 | .56 | .90 | .80 | .56 | .47 | .12 | 1.0 | .05 | .22 | 0.72 | .09 |

En base a los resultados obtenidos con el conjunto de datos de la competición *IberLEF 2021*, se lograron resultados comparables a los obtenidos por el equipo del CIC del IPN en dicha competición *(MexLef, 2021)*. Por otro lado, al experimentar con el corpus "*Spanish Fake and Real News",* se obtuvieron resultados aún más favorables. Específicamente, utilizando el método de *Búsqueda Dispersa (SS)*, se alcanzó una **puntuación *F1* de 0.80,** mientras que con el *Algoritmo Genético* (*GA)* se llegó a 0.79. Por el contrario, con el dataset de Colombiacheck, los *F1 scores* fueron significativamente más bajos (0.02 o 0.09), lo significa que, a pesar de utilizar tres corpus altamente confiables, no hay garantía absoluta de que la información sea auténtica.

Por último, es relevante mencionar que cada algoritmo se ejecutó 5 veces con diferentes *hiperparámetro*s para comparar sus resultados, aunque en el caso del algoritmo *MSA*, fue necesario ejecutarlo alrededor de 25 veces para obtener mejores resultados, demostrando así que es útil para la fase de exploración, pero no para la etapa final de resolución, lo que lleva a concluir que los métodos de búsqueda poblacional, como *Búsqueda Dispersa (SS) y Algoritmo Genético (GA)*, son más adecuados para la detección de noticias falsas en comparación con los métodos de búsqueda guiado como el *Recocido Multiarranque (MSA)*.

## 5 Conclusiones

En este trabajo, se ha diseñado e implementado una metodología para la detección de noticias falsas y fraude digital en español con el potencial de integrar mejoras futuras, como los *modelos de lenguaje*. El algoritmo de *Búsqueda Dispersa (SS)* se destacó al alcanzar

un puntaje *F1* de **0.80** en el segundo corpus experimentado *("Spanish Fake and Real News").* Además, el *Algoritmo Genético (GA)* también mostró buenos resultados, con un F1 de **0.79** en este mismo corpus.

Adicionalmente, la creación del nuevo *corpus* de fraudes en español representa un aporte significativo al área de investigación, ya que proporciona un recurso valioso para el desarrollo y la evaluación de nuevas técnicas de detección de fraudes.

Si bien, aunque los resultados fueron prometedores, se requieren grandes cantidades de datos etiquetados para entrenar y probar los algoritmos, similarmente a los *modelos de lenguaje*, como *BERT, BETO* y *RoBERTa*. Estos modelos se benefician de un corpus diverso y representativo que abarque diferentes temas y fuentes. Por ello, se propone un nuevo corpus etiquetado que contiene más de 41,000 casos de noticias y ofertas de trabajo en español proveniente de diversas fuentes y temáticas para investigar el fraude digital (ver Tabla 7).

Tabla 7. Estructura del nuevo corpus propuesto.

| ***Fuente*** | **Descripción** |
|---|---|
| *Spanish Fake News Corpus Version 2.0* | Contiene noticias falsas y reales en español extraídas de Twitter y Facebook. Cubre 9 temas diferentes y se recopiló entre noviembre de 2020 y marzo de 2021. Principalmente se obtuvieron de sitios web de verificación de hechos y redes sociales. |
| *Fake Job Posting Prediction* | Basado en el conjunto *EMSCAD (European Multilingual Job Offer Corpus).* Contiene ofertas de trabajo falsas y reales en inglés, algunas traducidas al español. Dado que el conjunto de datos original *EMSCAD* está en inglés, se debe llevar a cabo un proceso de traducción y adaptación para hacerlo accesible en español. |
| *ColombiaCheck* | Contiene noticias verificadas por periodistas colombianos. Proveniente de la página web colombiacheck.com para fines de investigación. Autor: Enrique Manzano. |
| *Spanish Fake and Real News* | Recopilado durante el año 2019 de diversos sitios web públicos. Creado para identificar noticias falsas y reales en español. Autor: Fabricio A. Zules. |
| *Noticias Falsas en español* | Contiene noticias falsas en español recopiladas de diversas fuentes. |

La propuesta del corpus se presenta como una iniciativa que complementa la metodología propuesta. Al estar disponible en español, el *conjunto de datos* puede ser utilizado por hispanohablantes para desarrollar métodos para detectar noticias falsas, fraude digital y ofertas de trabajo fraudulentas. Finalmente, se contempla la integración de *modelos de lenguaje* en el *corpus* para mejorar aún más la precisión y la eficiencia de la metodología.

## Referencias


Ali, K., Liu, C., Zain-Ul-Abdin, K., & Zaffar, M. A. (2021). "*Fake news on Facebook: examining the impact of heuristic cues on perceived credibility and sharing intention."* Internet Research, 32(1), 379–397. https://doi.org/10.1108/intr-10-2019-0442

Ali, K., & Zain-Ul-Abdin, K. (2020). "*Post-truth propaganda: heuristic processing of political fake news on Facebook during the 2016 U.S. presidential election".* Journal of Applied

Communication Research/Journal of Applied Communications Research, 49(1), 109–128. https://doi.org/10.1080/00909882.2020.1847311

Alvarez, O. H. (2021). *"Fraude laboral en la era digital".* Revista General de Derecho del Trabajo y de la Seguridad Social, ISSN-e 1696-9626, Nº. 59, 2021. https://dialnet.unirioja.es/servlet/articulo?codigo=8029600

Aqil, S., & Lahby, M. (2021). *"Modeling and solving the fake news detection scheduling problem"*. Studies in computational intelligence (pp. 231–242). https://doi.org/10.1007/978-3-030-90087-8_11

Aragón, M. E., Jarquín, H., Gómez, M. M. Y., Escalante, H. J., Villaseñor-Pineda, L., Gómez-Adorno, H., ... & Posadas-Durán, J. P. (Septiembre de 2020). *"Overview of mex-a3t at iberlef 2020: Fake news and aggressiveness analysis in mexican Spanish"*. In Notebook Papers of 2nd SEPLN Workshop on Iberian Languages Evaluation Forum (IberLEF), Malaga, Spain

Bondielli, A., & Marcelloni, F. (2019). *"A survey on fake news and rumour detection techniques. Information Sciences",* 497, 38–55. https://doi.org/10.1016/j.ins.2019.05.035

Cao, J., Luo, X., & Zhang, W. (2020). *"Corporate employment, red flags, and audit effort".* Journal of Accounting and Public Policy, 39(1), 106710. https://doi.org/10.1016/j.jaccpubpol.2019.106710

CLEF Initiative. (2024). The CLEF Initiative. https://clef-initiative.eu

Das, S. D., Basak, A., & Dutta, S. (2022). "*A heuristic-driven uncertainty-based ensemble framework for fake news detection in tweets and news articles".* Neurocomputing, 491, 607–620. https://doi.org/10.1016/j.neucom.2021.12.037

Dasilva, J. P., Ayerdi, K. M., & Galdospín, T. M. (2020). *"Fake news y coronavirus: detección de los principales actores y tendencias a través del análisis de las conversaciones en Twitter".* Profesional De La Informacion, 29(3). https://doi.org/10.3145/epi.2020.may.08

Gómez-Adorno, H., Posadas-Durán, J. P., Enguix, G. B., & Capetillo, C. P. (2021). "*Overview of FakeDeS at IberLEF 2021: Fake News Detection in Spanish Shared Task"*. Procesamiento del Lenguaje Natural, 67, 223-231

Hu, L., Wei, S., Zhao, Z., & Wu, B. (2022). *"Deep learning for fake news detection: A comprehensive survey"*. AI Open, 3, 133–155. https://doi.org/10.1016/j.aiopen.2022.09.001

Nasser, I. M., Alzaanin, A. H., & Maghari, A. Y. (2021). *"Online Recruitment Fraud Detection using ANN"*. 2021 Palestinian International Conference on Information and Communication Technology (PICICT) (pp. 13-17). https://doi.org/10.1109/PICICT53635.2021.00015

M. F. Mridha, A. J. Keya, M. A. Hamid, M. M. Monowar and M. S. Rahman. (2021). *"A Comprehensive Review on Fake News Detection With Deep Learning,"* in *IEEE Access*, vol. 9, pp. 156151-156170, 2021, https://doi.org/10.1109/ACCESS.2021.3129329

Manzano-Velasco, C. E. & Daza-Rosero L. S. (2024). "Sistema de alerta temprana para monitorear la propagación de noticias falsas: caso de estudio contexto político colombiano" [Trabajo de Grado, Universidad del Cauca]. Recuperado el 15 de agosto de 2024, de http://repositorio.unicauca.edu.co:8080/xmlui/handle/123456789/9485

Manzano-Velasco, C. E. (2024). DataSet Web scraping noticias verificadas. Kaggle. Recuperado el 15 de agosto de 2024, de https://www.kaggle.com/datasets/enriquemanzano/dataset-web-scraping-noticias-verificadas/data

Martínez-Gallego, K., Álvarez-Ortiz, A. M., & Arias-Londoño, J. D. (13 de octubre, 2021). *"Fake News Detection in Spanish Using Deep Learning Techniques".* https://doi.org/10.48550/arXiv.2110.06461

MexLef (2021). Fake News Detection in Spanish Shared Task. IberLef 2021 shared tasks. Recuperado el 15 de agosto de 2024, de https://sites.google.com/view/fakedes/home

MexLef. (2021). Results from The Fake News Detection task. IberLef 2021 shared tasks. Recuperado el 15 de agosto de 2024, de https://sites.google.com/view/fakedes/results

Posadas-Durán, J. P., Gómez-Adorno, H., Sidorov, G., & Escobar, J. J. M. (2019). *"Detection of fake news in a new corpus for the Spanish language"*. Journal of Intelligent and Fuzzy Systems, 36(5), 4869–4876. https://doi.org/10.3233/jifs-179034

Pulido, C., Ruiz-Eugenio, L., Redondo-Sama, G., & Villarejo-Carballido, B. (2020). *"A New Application of Social Impact in Social Media for Overcoming Fake News in Health*." International Journal of Environmental Research and Public Health, 17(7), 2430. https://doi.org/10.3390/ijerph17072430

Ramírez Cruz, J. M., Palacios Alvarado, S. Ú., Franca Tapia, K. E., Posadas Durán, J. P. F., Gómez Adorno, H. M., & Sidorov, G. (2021). The Spanish Fake News Corpus Version 2.0 en GitHub. Recuperado de https://github.com/jpposadas/FakeNewsCorpusSpanish

S. Hidayattullah, I. Surjandari and E. Laoh. (Octubre, 2020) *"Financial Statement Fraud Detection in Indonesia Listed Companies using Machine Learning based on Meta-Heuristic Optimization," 2020 International Workshop on Big Data and Information Security (IWBIS)*, Depok, Indonesia, 2020, pp. 79-84, https://doi.org/10.1109/IWBIS50925.2020.9255563

Singh, M. K., Ahmed, J., Alam, M. A., Raghuvanshi, K. K., & Kumar, S. (2023). *"A comprehensive review on automatic detection of fake news on social media"*. Multimedia Tools and Applications. https://doi.org/10.1007/s11042-023-17377-4

Thota, A., Tilak, P., Ahluwalia, S., & Lohia, N. (2018). *"Fake News Detection: A Deep Learning Approach"*. SMU Scholar. https://scholar.smu.edu/datasciencereview/vol1/iss3/10/

Tsai, C. M. (2023). *"Stylometric fake news detection based on natural language processing using named entity recognition: In-Domain and Cross-Domain analysis."* Electronics, 12(17), 3676. https://doi.org/10.3390/electronics12173676

Ulloa, L. C., Neira, C. C., Sáez-Trumper, D., & Bran, C. T. (2021). *"Fake news en Chile y España: ¿Cómo los medios nos hablan de noticias falsas?"* Journal of Iberian and Latin American Research, 1–18. https://doi.org/10.1080/13260219.2020.1909849

Whitehouse, C., Weyde, T., Madhyastha, P., & Komninos, N. (Mayo, 2022). "Evaluation of fake news detection with knowledge-enhanced language models". *Proceedings of the international AAAI conference on web and social media* (Vol. 16, pp. 1425-1429). https://doi.org/10.48550/arXiv.2204.00458

Yazdi, K. M., Yazdi, A. M., Khodayi, S., Hou, J., Zhou, W., & Saedy, S. (2 de enero, 2020). "*Improving fake news detection using K-Means and support vector machine approaches"*. https://publications.waset.org/10011058/improving-fake-news-detection-using-k-means-and-support-vector-machine-approaches

Zules Acosta, F. A. (2019). "Construcción de un dataset de noticias para el entrenamiento y evaluación de clasificadores automatizados". [Trabajo Fin de Máster Universitario en Ciberseguridad, Universidad Politécnica de Madrid]. ResearchGate. https://doi.org/10.13140/RG.2.2.31181.49126

Zules Acosta, F. A. (2019). "DataSet Web scraping noticias verificadas". Kaggle. Recuperado el 15 de agosto de 2024, de https://www.kaggle.com/datasets/zulanac/fake-and-real-news

# Capítulo 3

## Exploración de modelos de lenguaje de gran tamaño utilizando Procesamiento de Lenguaje Natural para el análisis de sentimientos en español

Alejandro Flores Ramírez[1] José A. Reyes-Ortiz[1] Josué Padilla Cuevas[1]
Maricela Bravo[1]

[1]Departamento de Sistemas, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Unidad Azcapotzalco

{flra,jaro,jpc,mcbc}@azc.uam.mx

**Resumen.** En el presente trabajo se realiza una revisión sobre la utilización de técnicas de Inteligencia Artificial en el ámbito del análisis de sentimientos, especialmente en el contexto de la pandemia de COVID-19. Específicamente, se describe el diseño y la implementación de un enfoque computacional para el análisis de sentimientos basado en grandes volúmenes de datos extraídos de la red social X (antes Twitter). Este enfoque emplea técnicas avanzadas de Procesamiento del Lenguaje Natural (PLN) para clasificar los sentimientos expresados en los tweets relacionados con el COVID-19. Este análisis se realiza con la exploración de diversos modelos de lenguaje de gran tamaño, comparándolos y almacenando los mejores resultados. Un proceso de evaluación fue llevado a cabo en términos de la métrica, bien conocida, exactitud, donde los resultados alentadores obtenidos proporcionan información valiosa sobre el estado emocional de la sociedad durante este periodo, permitiendo a los investigadores y profesionales de la salud pública interpretar fácilmente los datos y extraer conclusiones significativas.

**Palabras Clave:** COVID-19, análisis de sentimientos, procesamiento de lenguaje natural, modelos del lenguaje de gran tamaño.

## 1 Introducción

La pandemia global del COVID-19 ha tenido un impacto significativo y duradero en la salud mental de muchas personas, desencadenando una amplia gama de emociones y sentimientos, predominantemente negativos, durante los primeros años de la crisis sanitaria. Estas emociones se han manifestado principalmente en plataformas como las redes sociales, donde los usuarios han podido expresar sus opiniones, temores y sentimientos en respuesta a la emergencia global. En particular, Twitter ha destacado como un espacio crucial para que las personas compartieran sus experiencias y emociones.

El análisis de los sentimientos expresados en tweets se vuelve esencial, ya que brinda una perspectiva única sobre el estado emocional de la sociedad durante este periodo. Cada

tweet captura el estado emocional de los individuos durante esta crisis sanitaria. Examinar y comprender estas emociones en el contexto de la pandemia permite identificar patrones, tendencias y áreas de preocupación significativas, proporcionando información valiosa sobre cómo las personas enfrentan y procesan los desafíos que implica la situación sanitaria global.

Para abordar esta necesidad, se utilizarán tecnologías avanzadas como el análisis de sentimientos, también conocido como minería de opiniones o inteligencia artificial de sentimientos. Esta técnica de procesamiento del lenguaje natural (PLN) es el estudio computacional de las opiniones, sentimientos, emociones, estados de ánimo y actitudes de las personas (Liu, 2015).

La utilización de estas tecnologías permite analizar de manera automatizada grandes volúmenes de texto, identificando las emociones subyacentes expresadas por los usuarios. El uso de esta técnica de procesamiento requiere manejar grandes volúmenes de información, comúnmente referidos como Big Data. Este término se describe en relación con los desafíos de administración de datos que, debido al incremento en el volumen, la velocidad y la variedad de los datos, no pueden ser resueltos con bases de datos tradicionales (Marz y Warren, 2015). En el contexto de este proyecto, se recolectarán y analizarán grandes cantidades de datos provenientes de tweets relacionados con el COVID-19 para obtener una visión global y detallada del estado emocional de la población.

Por lo tanto, este artículo presenta un enfoque para el de análisis de sentimientos basado en grandes volúmenes de datos extraídos de mensajes de redes sociales. Este enfoque emplea técnicas avanzadas de procesamiento del lenguaje natural (PLN) y modelos de lenguaje de gran tamaño para extraer y clasificar los sentimientos expresados en los textos de los tweets relacionados con el COVID-19. Este análisis es de utilidad para interpretar fácilmente los datos y extraer conclusiones significativas. Este trabajo también explora una experimentación con modelos de lenguaje de gran tamaño (LLM) para la tarea de análisis de sentimientos, para ello se utilizan mensajes de redes sociales relacionados con el período de la pandemia del COVID-19. Por otro lado, se utilizan técnicas de Procesamiento de Lenguaje Natural para estudiar el comportamiento de los LLM.

El resto del artículo se organiza como sigue. La Sección 2 presenta una revisión del estado del arte. La Sección 3 expone el proceso completo de limpieza, procesamiento y clasificación de mensajes de redes sociales para el análisis de sentimientos utilizando modelos de lenguaje de gran tamaño. La Sección 4 muestra la experimentación y los resultados comparativos de los modelos de lenguaje de gran tamaño. Finalmente, la Sección 5 presenta las conclusiones y el trabajo a futuro.

## 2 Estado del arte

En esta sección, se explorarán diversos estudios que han evaluado el rendimiento de diferentes modelos y enfoques para el análisis de sentimientos. Se analizarán investigaciones recientes que comparan el desempeño de grandes modelos de lenguaje

(LLMs), como GPT-3.5, GPT-4, y Llama 2, con métodos tradicionales de aprendizaje profundo y aprendizaje por transferencia. Asimismo, se revisarán estudios que investigan la eficacia de técnicas como incrustación de palabras y TF-IDF en modelos populares de aprendizaje profundo, así como la aplicación de modelos específicos para el análisis de sentimientos en contextos idiomáticos y temáticos diversos, incluyendo el ruso y el español.

El estudio llevado a cabo por Krugmann et al (2024) se centra en evaluar el rendimiento de tres grandes modelos de lenguaje de última generación (LLMs) - GPT-3.5, GPT-4 y Llama 2 como analizadores de sentimiento de cero disparos para tareas de clasificación de sentimiento binaria y de tres clases. Este enfoque se compara con métodos tradicionales de aprendizaje por transferencia y otros modelos de alto rendimiento, como SiEBERT y RoBERTa. Al igual que en el estudio de Jnoub et al (2020), que se centró en proporcionar un modelo generalizado para el análisis de sentimientos combinando CNN con su propio algoritmo para transformar reseñas en vectores, este artículo busca abordar limitaciones específicas en la investigación actual del análisis de sentimientos.

Por otro lado, la investigación de Dang et al (2020) se basa en realizar un estudio comparativo del rendimiento de tres modelos populares de aprendizaje profundo (DNN, CNN y RNN) para el análisis de sentimientos. El estudio evalúa estos modelos en ocho conjuntos de datos utilizando dos técnicas de procesamiento de texto: incrustación de palabras (word embedding) y TF-IDF. El objetivo es comparar el rendimiento de estas técnicas y contribuir a la literatura de vanguardia sobre tareas de análisis de sentimientos. Los modelos se aplican para predecir la polaridad del sentimiento en el texto y clasificarlo en consecuencia, con evaluaciones basadas en métricas como la precisión general, el recall, la F-score y el Área Bajo la Curva (AUC).

En la investigación descrita por Ligthart et al (2021) se centra en realizar un estudio terciario sobre el análisis de sentimientos. El estudio tiene como objetivo proporcionar una visión consolidada del análisis de sentimientos mediante la síntesis de resultados de estudios de revisión sistemática. Se utilizan varios modelos y enfoques para el análisis de sentimientos, incluyendo algoritmos de aprendizaje automático como el aprendizaje profundo (por ejemplo, CNN, LSTM), aprendizaje no supervisado, aprendizaje en conjunto, métodos basados en léxico y modelos híbridos.

Rusnachenko et al (2024) investigan la aplicación de modelos de lenguaje de gran tamaño (LLMs) en el análisis de sentimiento dirigido de textos en ruso, centrándose específicamente en el sentimiento hacia entidades nombradas en artículos de noticias. El estudio utiliza el conjunto de datos RuSentNE-2023, que está anotado con el sentimiento hacia entidades nombradas en textos de noticias rusas. Los modelos utilizados en el experimento incluyen tanto "modelos cerrados" como la serie ChatGPT (GPT-3.5 y GPT-4) y "modelos abiertos". La investigación tiene como objetivo mejorar el rendimiento de las tareas de análisis de sentimiento en ruso aprovechando los modelos de lenguaje de gran tamaño y explorando diferentes estrategias para mejorar la precisión de la clasificación de sentimientos.

El artículo de Pan et al (2023) se centra en la evaluación de modelos de Transformers para el análisis de sentimiento dirigido en el ámbito financiero en español. Los investigadores tenían como objetivo extraer automáticamente el principal objetivo

económico de los textos financieros y determinar la polaridad del texto hacia el principal objetivo económico, la sociedad y otras empresas. Para lograr esto, compilaron un corpus de tweets financieros y titulares de noticias en español. El estudio compara el rendimiento de diferentes modelos de lenguaje grandes específicos para español, siendo MarIA y BETO los que lograron los mejores resultados. Los investigadores evaluaron la clasificación de sentimientos hacia el principal objetivo económico, la sociedad y otras empresas, logrando puntuaciones F1 macro de 76.04%, 74.16% y 68.07%, respectivamente. Además, la precisión para la detección de objetivos se reportó en 69.74%.

Con la revisión del estado del arte, se hace evidente que el uso de grandes modelos de lenguaje y algoritmos de aprendizaje automático son relevantes para la tarea de análisis de sentimientos en español, por ello este trabajo se centra en utilizar técnicas de pre-procesamiento de los datos basadas en PLN en esta tarea.

## 3 Enfoque propuesto

En esta sección se presenta el desarrollo del enfoque propuesto a partir del uso de grandes modelos del lenguaje en el análisis de sentimiento de un conjunto de datos que almacenan tweets del periodo del COVID-19. El proceso completo de este enfoque incluye la recolección de los mensajes de la red social, el procesamiento de los textos en español, el análisis de sentimientos mediante modelos de lenguaje de gran tamaño y el almacenamiento de los resultados.

### 3.1 Recolección de datos

Para construir el conjunto de datos, se recolectaron tweets en tiempo real durante el periodo de la pandemia, los cuales constan de más de 7 millones de tweets en español, extraídos durante el período de la pandemia de COVID-19. Cada uno de estos tweets refleja un sentimiento, lo cual subraya la importancia de conocer y analizar distintos modelos de análisis de sentimientos. El objetivo es determinar cuál de estos modelos es el más útil para esta tarea específica, basándonos en los resultados que cada uno ofrece. Para ello, se tomaron dos muestras significativas y distintas de tweets contenidos en el conjunto, con el fin de evaluar el desempeño de los grandes modelos de lenguaje en el análisis de sentimientos. Esto origina dos conjuntos de textos.

Para asegurar la precisión del análisis, cada tweet de estos dos conjuntos fue clasificado manualmente por humanos según la polaridad o el sentimiento que transmite (positivo, negativo, neutro). Estas clasificaciones manuales se realizaron para proporcionar un conjunto de datos de referencia confiable contra el cual evaluar el rendimiento de los grandes modelos de lenguaje en tareas de análisis de sentimientos.

Los dos conjuntos obtenidos se almacenaron en archivos de texto separados, con cada línea del archivo conteniendo un tweet en formato de texto plano seguido de su clasificación

manual correspondiente. Esto permite una fácil manipulación y acceso a los datos para posteriores análisis comparativos y entrenamiento de modelos.

Este enfoque metodológico permite observar y comparar de manera efectiva los resultados obtenidos por distintos modelos de lenguaje, facilitando la identificación del modelo más adecuado para tareas de análisis de sentimientos en el contexto específico de tweets en español durante la pandemia de COVID-19.

A estos dos conjuntos de textos se añade un tercer conjunto obtenido de *Kaggle*, el cual se describe abajo. Por lo tanto, se obtiene tres conjuntos de textos de la siguiente manera:

- Conjunto de textos 1: A partir del conjunto descrito anteriormente, se obtuvieron 1000 tweets relacionados con COVID-19 durante el año del 2020 clasificado manualmente.
- Conjunto de textos 2: A partir del conjunto descrito anteriormente, se obtuvieron 1000 tweets relacionados con COVID-19 durante el año del 2021 clasificado manualmente.
- Conjuntos de textos 3: A partir de *Kaggle*, se obtuvieron 1000 tweets en español de México obtenidos de la redsocial X (anteriormente Twitter), etiquetados y validados por expertos. Estos datos están clasificados en positivo, negativo y neutro.

### 3.2 Procesamiento de textos

Para evaluar el rendimiento de cada uno de los modelos, es esencial realizar un procesamiento de textos enfocado en la limpieza y extracción de información significativa de los tweets. Esto se debe a que los resultados del análisis de polaridad pueden variar significativamente según la técnica de preprocesamiento utilizada. Se pueden emplear una variedad considerable de métodos para ajustar nuestro conjunto de datos, y cada uno se realiza con tareas específicas para la limpieza de los tweets. A continuación, se explican cada una de estas técnicas:

- **Limpieza ligera**: Esta función se encarga de eliminar caracteres inusuales y componentes no contextuales. Dado que el conjunto de textos contiene muchos caracteres y cadenas de caracteres inusuales, como '¶', 'ý', 'à', '=51', '=61', estos se eliminan junto con enlaces y menciones de usuarios. Esta tarea se realiza utilizando expresiones regulares para encontrar y eliminar estos patrones de caracteres inusuales.
- **Limpieza profunda**: Esta función realiza una limpieza más exhaustiva basada en el resultado obtenido de la limpieza ligera. Aquí, mediante expresiones regulares, se eliminan caracteres no alfanuméricos, caracteres numéricos, *stopwords*, espacios extras, palabras de un solo carácter y hashtags.
- **Lematización**: La tarea principal de esta función es encontrar la forma base de cada palabra, con el objetivo de facilitar la interpretación del texto y reducir la ambigüedad lingüística. Partiendo del resultado de la limpieza profunda, la

lematización se realiza utilizando la biblioteca *es_core_news_sm* de spaCy, la cual produce el tweet lematizado.

- ***Stemming***: Esta función se enfoca en reducir cada palabra a su raíz, eliminando prefijos y sufijos. Utilizando el resultado de la limpieza profunda, la biblioteca que implementa el algoritmo denominado *SnowBallStemmer* de Porter (2001) en NLTK se emplea para obtener la lista completa de raíces en español, comparando cada palabra del tweet y sustituyéndola por su correspondiente raíz.

Este enfoque metodológico garantiza que los datos estén en su forma más limpia y estructurada posible antes de ser procesados por los modelos de análisis de sentimientos, lo que permite obtener distintos resultados al realizar el análisis del sentimiento, dependiendo de qué tipo de pe-procesado se utilice.

### 3.3 Análisis de sentimiento con modelos de lenguaje de gran tamaño

El análisis de sentimientos a partir de los mensajes de la red social se realiza mediante explorar distintos modelos de lenguaje de gran tamaño, los cuales son comparados para obtener el que mejores resultados ofrece.

El proceso de análisis de sentimientos extrae información sobre sentimientos y opiniones presentes en textos. Estos análisis son de gran valor para estudiar la tendencia de polaridades en un tema en específico. En este caso, es de suma importancia el observar la polaridad que existió en el periodo de la pandemia del COVID-19, pero además de ello el utilizar un gran modelo del lenguaje con un buen porcentaje de fiabilidad conlleva a obtener resultados más acertados a la hora de obtener la polaridad del tweet. Por ello, se utilizaron distintos modelos de lenguaje de gran tamaño para observar su fiabilidad a partir de la clasificación real de cada tweet, de los cuales se utilizaron los siguientes:

- *Sentiment-analysis-spanish* (CNN): Este modelo de lenguaje utiliza redes neuronales convolucionales (CNN) para predecir el sentimiento de oraciones en español. El modelo fue entrenado con más de 800,000 reseñas de usuarios provenientes de diversas plataformas como ElTenedor, Decathlon, TripAdvisor, FilmAffinity y eBay (Bello, 2021). Para el experimento, se empleó esta biblioteca utilizando su función “sentiment”, la cual retorna el sentimiento que el modelo predice a partir del tweet proporcionado. Esta herramienta permite analizar eficientemente la polaridad de los tweets, aprovechando la robustez de las CNN para el procesamiento del lenguaje natural.
- *Pysentimiento*: Utiliza modelos pre-entrenados de Transformers basados en *BETO* y *RoBERTuito* que han sido entrenados y ajustados específicamente para comprender y analizar textos en español. Estos modelos son capaces de captar matices y contextos complejos en los tweets, proporcionando análisis de sentimientos precisos. Para este experimento, se utilizó la función *analyzer.predict*, que devuelve la predicción de la polaridad de cada tweet

analizado. Esta función es fundamental, ya que permite aprovechar la capacidad de BERT para entender el contenido de los tweets, proporcionando una evaluación detallada y precisa del sentimiento expresado en cada uno.

- *Twitter-xlm-roberta-base-sentiment*: Este modelo de análisis de sentimientos utiliza la arquitectura de *XLM-RoBERTa*, que es una versión multilingüe de *RoBERTa* (Robustly Optimized BERT Approach). Para llevar a cabo tareas de análisis de sentimientos, se emplea específicamente el modelo Twitter-xlm-roberta-base-sentiment creado por Cardiff University (2023). Utilizando su función "*sentiment-analysis*", este modelo puede predecir la polaridad de cada tweet proporcionado, ofreciendo una evaluación precisa del sentimiento expresado en cada uno.
- *GPT (Generative Pre-trained Transformer):* Familia de modelos de lenguaje basados en la arquitectura Transformer, con la característica principal generar texto de manera coherente y relevante basándose en la entrada que reciben. Este modelo no está específicamente diseñado para el análisis de sentimientos como su tarea principal, pero puede usarse para esa tarea de manera indirecta. Para la investigación, se utilizó el API de GPT distribuido por OpenAI (2023), utilizando características de base del modelo en su versión "GPT-3.5-turbo-1106". Comenzando con contextualizar al modelo con la tarea a realizar para después darle indicaciones de las salidas y por último proporcionarle el tweet a analizar, para así obtener la predicción de polaridad.

## 4 Experimentación y resultados

Un proceso de experimentación con los diversos modelos de lenguaje de gran tamaño es desempeñado con la finalidad de comprarlos. De esta manera, cada modelo se utiliza para clasificar los sentimientos a partir del procesado de texto, y se obtiene el porcentaje de aciertos (exactitud) que los modelos logran. Esta exactitud se obtiene a partir de los 3 conjuntos de mensajes de la red social previamente clasificados de manera manual (por humanos).

Estos modelos son combinados con diversas configuraciones experimentales que incluyen a) procesamiento utilizando limpieza ligera; b) procesamiento de limpieza profunda; c) aplicación de la técnica de lematización sobre los textos; d) aplicación de la técnica de *stemmings* sobre los textos.

Los resultados obtenidos de los diferentes modelos y sus configuraciones experimentales se presentan a continuación, mostrando los porcentajes de exactitud alcanzados. Además, se incluye una breve descripción de estos resultados.

La Tabla 1 muestra los resultados de exactitud utilizando el modelo de lenguaje de gran tamaño basado en redes neuronales convolucionales.

Tabla 1. Resultados de Exactitud con Redes Neuronales Convolucionales (CNN)

| Conjunto de textos 1 | | | | |
|---|---|---|---|---|
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **sentiment-analysis-spanish** | 45.4% | 41.8% | 41.2% | 46.6% |
| Conjunto de textos 2 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **sentiment-analysis-spanish** | 48.7% | 47.1% | 45.9% | 50.0% |
| Conjunto de textos 3 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **sentiment-analysis-spanish** | 48.8% | 47.3% | 47.9% | 36.3% |

El análisis de rendimiento del modelo de análisis de sentimientos "sentiment-analysis-spanish" en diferentes conjuntos de datos y con diversos preprocesamientos revela patrones importantes. En los Datasets 1 y 2, el modelo muestra un rendimiento superior al trabajar con datos crudos y con stemming, sugiriendo que estas formas de preprocesamiento preservan o incluso resaltan características esenciales para la tarea de análisis de sentimientos. En contraste, los datos limpios y lematizados tienden a reducir el rendimiento del modelo, posiblemente debido a la eliminación de información contextual relevante. Por otro lado, en el Dataset 3, el modelo tiene un rendimiento general más bajo, con los datos crudos y lematizados ligeramente por delante, lo que indica que la clasificación previa de los datos puede impactar negativamente en la efectividad del preprocesamiento. Estos resultados destacan la importancia de ajustar las técnicas de preprocesamiento según las características específicas de cada conjunto de datos para optimizar el rendimiento del modelo de análisis de sentimientos.

La Tabla 2 expone los resultados del análisis de sentimientos utilizando el modelo basado en RoBERTuito que a su vez está sustentado en el modelo base BETO.

Tabla 2. Resultados de Exactitud con RoBERTuito/BETO

| Conjunto de textos 1 | | | | |
|---|---|---|---|---|
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **beto-sentiment-analysis** | 61.7% | 48.5% | 54.4% | 47.1% |
| Conjunto de textos 2 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **beto-sentiment-analysis** | 69.4% | 54.7% | 60.3% | 45.0% |
| Conjunto de textos 3 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **beto-sentiment-analysis** | 64.4% | 49.3% | 41.8% | 39.0% |

Para el modelo de "beto-sentiment-analysis" se observa que arroja resultados significativos para cada uno de los datasets utilizados. En el Dataset 1, el modelo muestra un rendimiento superior al trabajar con datos crudos (61.7%), mientras que el preprocesamiento con lematización también proporciona un rendimiento aceptable (54.4%). Sin embargo, la limpieza y el stemming reducen considerablemente la eficacia del modelo (48.5% y 47.1%, respectivamente). En el Dataset 2, los datos crudos nuevamente resultan en el mejor desempeño (69.4%), seguidos por la lematización (60.3%). Aquí también se observa que la limpieza y el stemming afectan negativamente al rendimiento (54.7% y 45%, respectivamente). En el Dataset 3, el modelo alcanza un buen rendimiento con los datos crudos (64.4%), aunque se observa una disminución considerable con la limpieza (49.3%), la lematización (41.8%) y especialmente con el stemming (39%).

Estos resultados indican que, para el modelo "beto-sentiment-analysis", los datos crudos son consistentemente los más eficaces, probablemente porque retienen toda la información contextual necesaria para el análisis de sentimientos. La lematización también muestra ser útil en algunos casos, aunque no alcanza el mismo nivel de rendimiento que los datos crudos. En contraste, la limpieza y el *stemming* tienden a reducir significativamente la eficacia del modelo, lo que sugiere que estos preprocesamientos pueden eliminar detalles cruciales.

La Tabla 3 presenta los resultados del análisis de sentimientos utilizando el modelo basado en RoBERTa.

Tabla 3. Resultados de Exactitud con RoBERTa

| Conjunto de textos 1 | | | | |
|---|---|---|---|---|
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **twitter-xlm-roberta-base-sentiment** | 62% | 61.9% | 62% | 59.4% |
| Conjunto de textos 2 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **twitter-xlm-roberta-base-sentiment** | 66% | 65% | 65.4% | 59.2% |
| Conjunto de textos 3 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **twitter-xlm-roberta-base-sentiment** | 59.8% | 54.5% | 51.3% | 49.6% |

Para este modelo se observa que el rendimiento es notablemente consistente en los textos crudos, limpios y lematizados, con puntuaciones muy similares en el Dataset 1 (62%) y Dataset 2 (66% y 65%, respectivamente). Sin embargo, el rendimiento disminuye ligeramente con el stemming (59% en ambos datasets). En el Dataset 3, los textos crudos alcanzan un rendimiento cosiderable (59.8%), seguidos por los textos lematizados (51.3%) y limpios (54.5%), mientras que el stemming muestra nuevamente el menor rendimiento (49.6%). Estos resultados indican que este modelo maneja bien los textos crudos, limpios y

lematizados, manteniendo un rendimiento robusto, pero el stemming tiende a reducir su eficacia, probablemente debido a la pérdida de información contextual crítica.

La Tabla 4 presenta los resultados del análisis de sentimientos utilizando el modelo de lenguaje de gran tamaño GPT-3.

Tabla 4. Resultados de Exactitud con GPT-3.

| Conjunto de textos 1 | | | | |
|---|---|---|---|---|
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **GPT** | 66.4% | 54.3% | 51.0% | 39.2% |
| Conjunto de textos 2 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **GPT** | 66.4% | 62.2% | 52.4% | 43.0% |
| Conjunto de textos 3 | | | | |
| Modelo | Crudo | Limpio | Lematizado | Stemming |
| **GPT** | 64.2% | 50.5% | 44.0% | 40.6% |

Los resultados presentados reflejan el rendimiento del modelo GPT-3 en diferentes versiones de datos textuales: crudos, limpios, lematizados y con stemming, evaluados en tres conjuntos de datos distintos. En todos los casos, el rendimiento del modelo es más alto con los datos sin procesar (crudos), obteniendo un 66.4%, 66.4% y 64.2% de precisión para los conjuntos 1, 2 y 3 respectivamente. Sin embargo, el rendimiento disminuye considerablemente cuando los datos son procesados. La limpieza de datos, aunque a veces mejora la precisión, generalmente muestra una reducción en comparación con los datos crudos. Los datos lematizados y con *stemming* presentan las mayores caídas en el rendimiento, con precisiones que descienden hasta 51% y 39.2% en el conjunto de textos 1, 52.4% y 43% en el conjunto de textos 2, y 44% y 40.6% en el conjunto de textos 3.

Los resultados obtenidos sugieren que para los modelos utilizados de aprendizaje profundo maneja mejor la complejidad del lenguaje natural cuando trabaja con datos en su forma original, ya que el preprocesamiento puede eliminar información contextual crucial. Aunque la limpieza de datos puede ser beneficiosa en ciertos contextos, en este caso, parece perjudicar la capacidad del modelo para hacer predicciones precisas. Los procesos de lematización y stemming, diseñados para simplificar el texto, resultan en una pérdida significativa de precisión, probablemente debido a la eliminación de matices importantes del lenguaje.

Por otro lado, para el modelo de Red Neuronal Convolucional muestra una dependencia diferente del preprocesamiento en comparación con los modelos basados en Transformers. las CNN pueden beneficiarse del *stemming* y lematización al simplificar el texto y reducir la variabilidad léxica, ayudando a identificar patrones consistentes. Sin embargo, la limpieza de datos no siempre es beneficiosa y puede eliminar información valiosa, esto se puede observar en los resultados obtenidos, debido a que el stemming que parece

beneficioso en algunos casos al simplificar el léxico y ayudar a *identificar* patrones, en otros casos, como el conjunto de textos clasificado (dataset 3), esta simplificación puede eliminar demasiada información crucial.

En general, al comparar los diferentes modelos utilizados para el análisis de sentimientos, se observa una clara tendencia: los modelos RoBERTuito/BETO y GPT muestran un rendimiento superior con datos crudos, alcanzando sus mejores porcentajes de precisión (66.5% y 69.4%, respectivamente) debido a que estos modelos aprovechan al máximo la riqueza contextual completa de los textos originales. Esto se debe a que el preprocesamiento, como la lematización y el *stemming*, puede eliminar matices y detalles cruciales del lenguaje natural que estos modelos de Transformers utilizan para hacer predicciones precisas. Por otro lado, el modelo RoBERTa mantiene un rendimiento consistente con datos crudos, limpios y lematizados, ya que su arquitectura es robusta frente a distintas formas de preprocesamiento, aunque aún sufre una ligera caída con el *stemming*. Las Redes Neuronales Convolucionales (CNN), aunque a veces benefician del *stemming* al simplificar el texto y reducir la variabilidad léxica, generalmente presentan un rendimiento inferior debido a su menor capacidad para capturar la complejidad contextual en comparación con los modelos basados en Transformers.

Como resumen podemos observar que el modelo BETO se destaca como el más efectivo con datos crudos debido a su capacidad de manejar información contextual rica, seguido de cerca por GPT, mientras que los modelos CNN y RoBERTa muestran más variabilidad dependiendo del preprocesamiento aplicado.

## 5 Conclusiones

Este artículo ha presentado un enfoque para el análisis de sentimientos a partir de mensaje de la red social en español utilizando modelos de lenguaje de gran tamaño. Los modelos de lenguaje de gran tamaño fueron experimentados de manera detallada para presentar una comparativa entre ellos y obtener el que mejore resultados ha mostrado para los conjuntos de textos estudiados.

Las principales aportaciones de este trabajo son a) la recolección de conjuntos de textos en español y el etiquetado manual de un conjunto de textos basados en su sentimiento; b) la implementación de técnicas de procesamiento de lenguaje natural para la limpieza y preparación de los textos; c) el uso de grandes modelos de lenguaje para la tarea de análisis de sentimientos en español; d) el enfoque comparativo de diversos modelos de lenguaje de gran tamaño y su adaptación a la tarea; e) la obtención de los mejores resultados para el conjunto de textos estudiados.

El proceso de comparación de los diversos modelos ha mostrado que, para el análisis de sentimientos, se observa que los modelos RoBERTuito/BETO y GPT muestran un rendimiento superior con datos crudos, alcanzando sus mejores porcentajes de exactitud 69.4% y 66.4%, respectivamente.

Este análisis de sentimientos detallado puede servir como un indicador temprano de problemas emergentes, facilitando una respuesta más informada y eficaz a las necesidades de salud mental durante crisis sanitarias globales.

Como trabajo a futuro se propone el uso y experimentación de otros modelos de lenguaje de gran tamaño como los basados en GPT-4, LlaMa, Gemini y PaLM. Así como experimentar con características sintácticas y semánticas de los textos para alimentar estos modelos.

# 6 Referencias


Bello, H. (2021). *Sentiment-analysis-spanish 0.0.25*. Recuperado de https://pypi.org/project/sentiment-analysis-spanish/

Cardiff University (2023). *Twitter-XLM-RoBERTa-base-sentiment.* Recuperado de https://huggingface.co/cardiffnlp/twitter-xlm-roberta-base-sentiment

Dang, N. C., Moreno-García, M. N., De la Prieta, F. (2020). "Sentiment analysis based on deep learning: A comparative study ", *Electronics*, vol. 9, pp. 483.

Jnoub, N., Al Machot, F., Klas, W. (2020). "A domain-independent classification model for sentiment analysis using neural models ", *Applied Sciences*, vol. 10, pp. 6221.

Krugmann, J. O., Hartmann, J. (2024). "Sentiment Analysis in the Age of Generative AI", *Customer Needs and Solutions*, vol. 11, pp. 3.

Ligthart, A., Catal, C., Tekinerdogan, B. (2021). "Systematic reviews in sentiment analysis: a tertiary study", *Artificial Intelligence Review*, vol. 54, pp. 4997-5053.

Marz, N., & Warren, J. (2015). "A new paradigm for Big Data" en Simon and Schuster. (eds), *Big Data: Principles and Best Practices of Scalable Realtime Data Systems* (pp. 23), Manning Publications.

OpenAI (2023). *New models and developer products announced at DevDay*. Recuperado de https://platform.openai.com/docs/models/gpt-3-5-turbo

Pan, R., García-Díaz, J. A., Garcia-Sanchez, F., Valencia-García, R. (2023). "Evaluation of transformer models for financial targeted sentiment analysis in Spanish". *PeerJ Computer Science*, vol. 9, pp. 1-33

Porter, M. F. (2001). *Snowball: A language for stemming algorithms.* Recuperado de http://snowball.tartarus.org/texts/introduction.html

Rusnachenko, N., Golubev, A., Loukachevitch, N. (2024). *Large Language Models in Targeted Sentiment Analysis*. Recuperado de https://arxiv.org/pdf/2404.12342.

Zhao, J., Liu, K., Xu, L. (2016). *Sentiment analysis: mining opinions, sentiments, and emotions*, Cambridge University Press.

# Capítulo 4

## Clasificación automática de noticias criminales en línea utilizando aprendizaje automático

Aarón E. Alvarez-Gómez[1], José A. Reyes-Ortiz[1], Josué Padilla-Cuevas[1], Leonardo D. Sánchez-Martinez[1]

[1] Universidad Autónoma Metropolitana, Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería

{aeag, jaro, jpc, ldsm}@azc.uam.mx

**Resumen.** En la actualidad, la información disponible en línea ha incrementado notablemente, esta gran cantidad de información ha llevado a que el acceso y la interpretación de las noticias alcancen niveles de complejidad que dificultan el seguimiento de las noticias en línea sobre eventos delictivos. Este artículo aborda el desafío mediante el desarrollo de un sistema de recuperación y análisis de noticias criminales utilizando aprendizaje automático. Inicialmente, se extraen datos del sitio de un periódico en línea mediante *web scraping*, y posteriormente se clasifican empleando cuatro algoritmos: *Random Forest* (RF), *Support Vector Machine* (SVM), *Naive Bayes* (NB) y *Gradient Boosting* (GB). Estos algoritmos se aplican a datos preprocesados mediante las técnicas de *stemming* y *lematización*. Los resultados indican que RF alcanzó la mayor precisión (97.14%) utilizando datos lematizados, mientras que NB mostró el peor desempeño debido a su suposición de independencia de características. La técnica de lematización superó al *stemming* en términos de precisión promedio (0.9007 frente a 0.8804). Los datos se organizaron en tres conjuntos distintos: un conjunto completo, un conjunto equilibrado y un conjunto que solo incluía las categorías predominantes. El análisis revela que el tercer grupo, que excluía la categoría con menos ocurrencias, tuvo el mejor desempeño en todas las métricas. De esta manera, el trabajo realizado demostró con éxito la viabilidad y eficacia de un sistema automatizado para recuperar y clasificar noticias, facilitando la identificación eficiente de eventos criminales.

**Palabras Clave:** Recuperación de noticias, Aprendizaje supervisado, Clasificación de eventos criminales.

# 1 Introducción

La creciente digitalización de la información ha transformado la manera en que se accede y procesa las noticias. En este contexto, el presente trabajo aborda un desafío específico en la gestión de información: la identificación y clasificación de noticias criminales. El problema central se manifiesta en la gran cantidad de noticias generadas constantemente, lo que dificulta la tarea de identificar y clasificar eventos criminales relevantes. El presente artículo tiene como objetivo general recuperar y clasificar noticias a partir de un medio digital para identificar eventos criminales. Para lograrlo, se propone recolectar automáticamente noticias desde el sitio web de El Universal (n.d.), procesarlas utilizando técnicas de procesamiento de lenguaje natural, y clasificarlas en las categorías de robo, secuestro y homicidio mediante aprendizaje supervisado. Finalmente, se evalúan los resultados mediante métricas estándar para la medición del rendimiento de los algoritmos.

El sistema de recuperación y clasificación de noticias criminales responde a la necesidad imperante de gestionar la sobreabundancia de información desorganizada en tiempo real. En la era digital actual, la rapidez con la que se generan y diseminan noticias ha llevado a un entorno informativo saturado, dificultando la tarea de identificar y entender de manera eficiente los eventos criminales que acontecen a nivel local e internacional.

La obtención de esta clasificación y organización eficiente de la información beneficiará a una amplia gama de usuarios, desde el público en general hasta profesionales en seguridad y periodistas. La solución propuesta simplifica la navegación en un entorno informativo complejo y proporciona una herramienta valiosa para tomar decisiones informadas y reconocer rápidamente patrones críticos.

El resto del artículo está distribuido en las siguientes secciones. En la sección 2, se revisan estudios previos y se comparan enfoques y algoritmos utilizados en la clasificación de noticias. La sección 3 detalla el proceso de recolección de datos mediante *web scraping*, el procesamiento y etiquetado de noticias. En la sección 4 se presentan los resultados de la clasificación junto con las métricas de evaluación. Finalmente, en la sección 5 se discuten los hallazgos y la efectividad del sistema propuesto para identificar eventos criminales en medios digitales.

# 2 Estado del Arte

Numerosos estudios han abordado la clasificación de noticias utilizando diversas técnicas de procesamiento de lenguaje natural y aprendizaje automático.

Rahman et al. (2021) evaluaron el rendimiento de varias técnicas de aprendizaje profundo y automático en la clasificación de noticias en idioma Bengalí. Utilizaron algoritmos como BiLSTM, GRU-LSTM, LSTM, Char-CNN, *BERT* y Text-GCN, encontrando que Text-GCN superó a los otros modelos con una precisión, exhaustividad *y* $F_1$ del 96.25%, y un AUC de 0.98 en la curva *ROC*, demostrando un rendimiento superior en la clasificación de noticias.

Leonard et al. (2022) desarrollaron un clasificador de noticias basado en titulares utilizando el algoritmo *Support Vector Classifier* (SVC) y comparándolo con otros modelos de aprendizaje automático como *Logistic regression*, *Naive Bayes Multinomial*, *Decision tree y Random Forest*. Utilizando un conjunto de datos de Kaggle con noticias clasificadas en siete categorías, el SVC logró una precisión del 96.34%, una exhaustividad del 95.50% y un $F_1$ del 95.92%, superando a *Logistic Regression*, que obtuvo una precisión del 92.45%, una exhaustividad del 91.20% y un $F_1$ del 91.82%, a *Naive Bayes Multinomial* con una precisión del 89.75%, una exhaustividad del 88.60% y un $F_1$ del 89.17%, a *Decision Tree* con una precisión del 90.25%, una exhaustividad del 89.00% y un $F_1$ del 89.62%, y a *Random Forest* con una precisión del 93.80%, una exhaustividad del 92.70% y un $F_1$ del 93.24%.

A Transformer Network-Based Deep Learning Approach: Hossain et al. (2023) emplearon redes neuronales basadas en transformadores para clasificar noticias sobre crímenes y modelar actividades relacionadas con drogas en artículos bengalíes. Utilizando técnicas de clasificación *Zero-Shot* y *Named-Entity Recognition* (NER), el modelo alcanzó una precisión de 0.96, una exhaustividad de 0.92 y un $F_1$ de 0.94 en la clasificación binaria de crímenes, y una precisión de 0.95, una exhaustividad de 0.98 y un $F_1$ de 0.97 en la clasificación multiclase de tipos de crímenes.

Tabassoum y Akber (2024) investigaron la clasificación de categorías de noticias utilizando algoritmos de aprendizaje automático y técnicas de inteligencia artificial explicable (XAI). Implementaron *Random Forest, Logistic Regression, Decision Tree y K-Nearest Neighbors* (KNN) utilizando *embeddings* contextuales generados por *Sentence-BERT* (*SBERT*). *Random Forest* obtuvo la mayor precisión con un 91.48%, seguido de *Logistic Regression* con 86.86%, *K-Nearest Neighbors* con 86.35% y *Decision Tree* con 74.70%. Para mejorar la interpreta bilidad, se empleó *LIME (Local Interpretable Model-agnostic Explanations)*, una técnica *XAI*, que proporcionó explicaciones claras de las predicciones del modelo.

N. N y C. J (2024) desarrollaron un enfoque híbrido para la clasificación de noticias que combina técnicas de aprendizaje profundo y aprendizaje automático. Utilizaron un conjunto de datos de más de 450,000 registros provenientes de fuentes como la BBC y Microsoft. El modelo *BERT* alcanzó una precisión del 98%, seguido de L*inear Support Vector Machine* (LinearSVC) con 95%, seguido de *Logistic Regression multinomial* con un 79%, *Naive Bayes multinomial* con un 68%, y el modelo de *Random Forest* con un 45%, demostrando la efectividad de BERT en la clasificación de noticias.

Sreekala et al. (2024) propusieron una integración novedosa del clustering jerárquico y los algoritmos de clasificación en conjunto. Utilizando un conjunto de datos de noticias de la BBC, *Bagging Classifier* alcanzó una precisión del 88%, una *exactitud* del 89.5% y una exhaustividad del 88%. *Random Forest* obtuvo una precisión del 90.2%, una exactitud del 92.9% y una exhaustividad del 93.1%. *Gradient Boosting* logró una precisión del 91%, una exactitud del 90% y una exhaustividad del 93%. Al integrar las etiquetas de *clustering* jerárquico, *Bagging Classifier* mejoró a una precisión del 90%, una exactitud del 90.5% y una exhaustividad del 89%. Random Forest alcanzó una precisión del 94%, una *exactitud*

del 93.6% y una exhaustividad del 94%. El Gradient Boosting aumentó a una precisión del 94%, una exactitud del 92% y una exhaustividad del 93%.

P. S et al. (2023) implementaron un modelo basado en transformadores, específicamente *BERT*, para la clasificación de noticias, logrando una precisión del 90%, comparativamente, modelos como LSTM y GRU mostraron precisión del 82.74% y 87.48%, respectivamente. Utilizaron un conjunto de datos de Kaggle con 20,957 registros distribuidos en 42 categorías de noticias, destacando la efectividad de *BERT* en esta tarea.

Agarwal et al. (2023) evaluaron la efectividad de cuatro algoritmos de *machine learning*: Ran*dom Forest, Decision Tree, K-Neighbor Classifier y Gaussian NB*, en la clasificación de noticias, utilizando un conjunto de datos de Kaggle con aproximadamente 210,000 titulares de noticias. *Random Forest* alcanzó la mayor precisión con un 95%, seguido por el *K-Neighbor Classifier* con un 93%, Gaussian *NB* con un 85% y por último, *Decision Tree* obtuvo una precisión del 80%.

S. B y Santhanalakshmi (2023) categorizaron automáticamente artículos de noticias utilizando técnicas de *embeddings* como *Glove, Word2Vec, FastText y ELMo*, junto con algoritmos de *Machine Learning* y *Deep Learning*. Utilizaron el *Antonio Gulli's (AG) News Topic Classification Dataset, que contiene* 12,000 artículos para entrenamiento y 7,600 para prueba, distribuidos en cuatro categorías: Mundo, Deportes, Negocios y Ciencia/Tecnología. Los resultados mostraron que los modelos de *Deep Learning*, específicamente LSTM con *embeddings Glove*, lograron una precisión del 92%, mientras que SVM con FastText obtuvo un 87%. *Modelos de Machine Learning* como *Naive Bayes y Decision Tree* lograron precisiones de hasta el 80% y 71%, respectivamente, demostrando la superioridad de los modelos de *Deep Learning* en la clasificación de noticias.

Veziroğlu et al. (2024) evaluaron la efectividad de los algoritmos de Naive Bayes en la clasificación de titulares de noticias utilizando el BBC News Corpus, que incluye 2225 titulares distribuidos en cinco categorías. Implementaron cuatro variantes de *Naive Bayes* junto con otros algoritmos populares. *Complement Naive Bayes* obtuvo la mayor precisión (98.31%), seguido de *Multinomial Naive Bayes* (98.20%), *Bernoulli Naive Bayes* (96.67%) y *Gaussian Naive Bayes* (92.92%). Entre los algoritmos de aprendizaje automático, *MLP Classifier* mostró el mejor rendimiento con una precisión del 98.31%, igualando al *Complement Naive Bayes*. *Linear SVC y Random Forest* también mostraron un rendimiento alto con precisiones de 97.97% y 97.86%, respectivamente. En contraste, *Decision Tree y KNN* tuvieron un rendimiento significativamente inferior, con precisiones de 82.69% y 55.84%, respectivamente.

Estos estudios demuestran la diversidad de enfoques y algoritmos que se pueden utilizar para la clasificación de noticias, destacando la importancia de seleccionar el modelo adecuado y las técnicas de procesamiento de datos para mejorar la precisión y el rendimiento en tareas de clasificación automática de noticias.

El presente artículo se diferencia de los estudios previos al centrarse en la recuperación y clasificación de noticias en español desde un sitio web. Mientras que otros trabajos han abordado la clasificación de noticias en diversos idiomas y contextos más generales, este trabajo emplea técnicas de procesamiento de lenguaje natural adaptadas a noticias en español y utiliza cuatro algoritmos de aprendizaje automático: *Random Forest (RF)*,

*Support Vector Machine (SVM), Naive Bayes (NB) y Gradient Boosting (GB).* Además, se destaca por el uso de dos versiones preprocesadas de datos textuales: *stemming* y *lematizado*, y por la implementación de un sistema de recolección automática de noticias. La evaluación mediante métricas estándar y la organización de los datos en tres grupos distintos aseguran la comparabilidad y relevancia de los resultados.

# 3 Metodología

El presente artículo se enfoca en la recuperación y clasificación de noticias de un medio digital con el objetivo de identificar eventos criminales específicos. Para lograrlo, se ha diseñado una metodología estructurada que, como se puede apreciar en la Figura 1, se divide en cuatro fases.

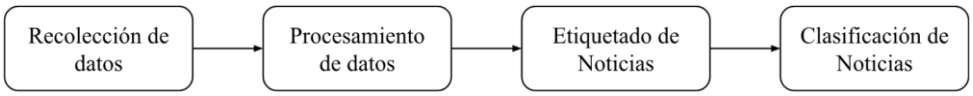


Figura 1. Diagrama de bloques que muestra las fases del proceso de clasificación de noticias

A continuación, se describen de manera general cada una de estas fases, para luego entrar en detalle en los procedimientos específicos y las herramientas utilizadas en cada etapa del proceso.

El trabajo comenzó con la recolección de datos, extrayendo noticias de la sección "minuto por minuto" del sitio web de El Universal mediante un script que navegó por la página, identificando y recolectando la información relevante. Posteriormente, se transformaron y organizaron los datos recolectados en un formato estructurado y limpio, generando un archivo CSV actualizado con los datos preprocesados, adecuado para la siguiente fase. Luego, se realizó un etiquetado inicial de las noticias limpias utilizando patrones específicos definidos mediante expresiones regulares (regex), lo cual sirvió como base para el entrenamiento de los modelos de clasificación. Finalmente, se le asignó una categoría específica a cada noticia empleando modelos de aprendizaje automático entrenados con cuatro algoritmos diferentes, clasificando las noticias en robo, secuestro u homicidio.

## 3.1 Recolección de datos

En esta sección se describe el proceso de recolección de noticias desde el sitio web El Universal mediante web *scraping*, utilizando *Selenium* debido a la falta de una API pública. Se examina la estructura del sitio web y se ofrece una visión general del funcionamiento del script de extracción, presentando finalmente los datos obtenidos.

Para la recolección de datos se empleó *Selenium*, una herramienta de automatización de navegadores web que permite simular la interacción humana. Esta herramienta facilitó la automatización de la navegación, búsqueda y extracción de noticias de manera eficiente.

El Universal.com, un periódico destacado en México organiza sus noticias cronológicamente en la sección 'Minuto por Minuto'. Cada entrada incluye un título, una marca de tiempo y, en algunos casos, un breve resumen.

El script de extracción automatiza la recolección de datos desde la sección 'Minuto por Minuto' usando *Firefox* y *GeckoDriver*. Finalmente, el DataFrame completo se guarda en un archivo CSV.

Cada registro en el archivo CSV representa un artículo de noticias y contiene información detallada sobre el evento reportado, incluyendo la fecha y hora de publicación, la categoría del artículo (como EDOMEX), el título, la URL y el texto completo del artículo. La Figura 2 muestra un archivo CSV con los datos en bruto antes de cualquier procesamiento o limpieza. Para este trabajo, se descargaron noticias filtradas por las categorías de Estados, Edomex, Metrópoli y Nación.

| MARCA TEMPORAL | CATEGORIA | TITULO | URL | CONTENIDO |
|---|---|---|---|---|
| 24/04/2027 14:07 | NACIÓN | López Obrador supervisa avances del Tren Maya en Campeche | https://www.eluniversal.com.mx/nacion/lopez-obrador-supervisa-avances-del-tren-maya-en-campeche/ | El presidente Andrés Manuel López Obrador realiza una gira de trabajo por Calkiní, Campeche, donde supervisa los avances de la operación del Tren Maya.A través de sus redes sociales, la Gobernadora de Campeche, Layda Sansores (Morena), informó que acompañó al titular de Ejecutivo federal en la referida gira.“Acompañé a nuestro querido Presidente Andrés Manuel @lopezobrador_ en el #TrenMaya de Campeche a Calkini. Les comparto algunas fotos con el...”En las imágenes se observa al Presidente de la República y a la gobernadora Sansores con usuarios del llamado “Jaguar Rodante”.El presidente López Obrador estará todo el fin de semana en el sureste mexicano donde realiza una gira de trabajo.nro/mcc |

Figura 2. CSV con los datos en bruto

En la Tabla1 se muestra la distribución de las 33,000 noticias recolectadas proporcionando una visión clara de la composición del conjunto de datos.

Tabla 1. Cantidad de noticias descargas por sección del periódico.

| **Sección** | **EDOMEX** | **METRÓPOLI** | **NACIÓN** | **ESTADOS** |
|---|---|---|---|---|
| **Cantidad de noticias** | 4,247 noticias | 9,542 noticias | 9,649 noticias | 9,562 noticias |

### 3.2 Procesamiento de datos

Se llevaron a cabo diversas acciones para mejorar la calidad de los datos y facilitar su análisis. Inicialmente, se identificó y eliminó el ruido, incluyendo caracteres especiales, signos de puntuación y palabras sin relevancia para el análisis. También se eliminaron las palabras vacías (*stopwords*) utilizando una lista predefinida proporcionada por la librería *nltk*. Posteriormente, se aplicó *lematización* para reducir las palabras a su forma base y

*stemming* para simplificarlas a su raíz. Estas técnicas ayudan a simplificar el vocabulario y reducir la variabilidad en los datos, lo que facilita su análisis. Finalmente, los datos procesados se guardaron en un nuevo archivo CSV con la fecha y hora actuales para su identificación y uso futuro.

El archivo CSV resultante contiene nuevos campos derivados del procesamiento realizado. El campo "SIN RUIDO" incluye el texto del contenido después de haber sido limpiado de caracteres especiales y palabras irrelevantes. El campo "LEMATIZADO" presenta el texto sin ruido tras ser *lematizado*, reduciendo las palabras a su forma base. El campo "DERIVADO" muestra el texto sin ruido después de aplicar *stemming*, reduciendo las palabras a su raíz más simple.

La Figura 3 proporciona una muestra de los datos tras finalizar esta fase, donde se pueden observar los datos procesados y refinados, libres de ruido y preparados para el siguiente paso.

| MARCA TEMPORAL | CATEGORIA | TITULO | URL | CONTENIDO | SIN_RUIDO | LEMATIZADO | DERIVADO |
|---|---|---|---|---|---|---|---|
| 24/04/2027 14:07 | NACIÓN | López Obrador supervisa avances del Tren Maya en Campeche | https://www.eluniversal.com.mx/nacion/lopez-obrador-supervisa-avances-del-tren-maya-en-campeche/ | El presidente Andrés Manuel López Obrador realiza una gira de trabajo por Calkiní, Campeche, donde supervisa los avances de la operación del Tren Maya.A través de sus redes sociales, la Gobernadora de Campeche, Layda Sansores (Morena), informó que acompañó al titular de Ejecutivo federal en la referida gira."Acompañé a nuestro querido Presidente Andrés Manuel @lopezobrador_ en el #TrenMaya de Campeche a Calkini. Les comparto algunas fotos con el..."En las imágenes se observa al Presidente de la República y a la gobernadora Sansores con usuarios del llamado "Jaguar Rodante".El presidente López Obrador estará todo el fin de semana en el sureste mexicano donde realiza una gira de trabajo.nro/mcc | el presidente andrés manuel lópez obrador realiza gira trabajo calkiní campeche supervisa avances operación tren maya a través redes sociales gobernadora campeche layda sansores morena informó acompañó titular ejecutivo federal referida gira "acompañé querido presidente andrés manuel @lopezobrador_ #trenmaya campeche calkini les comparto fotos el..."en imágenes observa presidente república gobernadora sansores usuarios llamado "jaguar rodante" el presidente lópez obrador fin semana sureste mexicano realiza gira trabajo nro/mcc | el presidente andrés manuel lópez obrador realizar gira trabajo calkini campeche supervisar avanz operación tren maya a través red social gobernadoro campeche layda sansor morena informar acompañar titular ejecutivo federal referido gira " acompañé querido presidente andrés manuel @lopezobrador _ # trenmaya campeche calkini él compartir foto el ... "en imagen observar presidente república gobernadora sansor usuario llamado " jaguar rodante " el presidente lópez obrador fin semana sureste mexicano realizar gira trabajo nro / mcc | el president andres manuel lopez obrador realiz gir trabaj calkin campech supervis avanc oper tren may a traves red social gobern campech layd sansor moren inform acompañ titul ejecut federal refer gir " acompañ quer president andres manuel @ lopezobrador_ # trenmay campech calkini les compart fot el... " en imagen observ president republ gobern sansor usuari llam " jagu rodant " el president lopez obrador fin seman surest mexican realiz gir trabaj nro/mcc |

Figura 3. Muestra de los datos luego de pasar por la fase de Procesamiento de Datos

### 3.3 Etiquetado de noticias

Para el etiquetado inicial de los textos procesados, se utilizaron patrones y expresiones regulares adaptados del trabajo **cita oculta para la revisión a ciegas** el cual se centró en la extracción de eventos criminales mediante patrones lingüísticamente enriquecidos.

#### 3.3.1 Categorías de noticias

Se definieron tres tipos de crímenes: asalto, secuestro y homicidio. Un asalto se describe como un evento en el que el atacante utiliza violencia para aprovecharse de la víctima en condiciones de inseguridad. El secuestro se define como la privación de libertad de la víctima con el fin de obtener un beneficio monetario. El homicidio se refiere a la privación de la vida de una persona por cualquier medio.

#### 3.3.2 Descripción de los patrones

Originalmente, los patrones incluían etiquetas gramaticales como “DT” (determinante) y “PREP” (preposición). Sin embargo, estos patrones se simplificaron para enfocarse directamente en palabras clave relevantes como "robo", "asalto", "asaltar", "robar" y "despojar". Se eliminaron las etiquetas gramaticales y se añadieron delimitadores de palabra

(\b) para asegurar la identificación precisa de las palabras exactas, reduciendo la ambigüedad y mejorando la precisión.

### 3.3.3 Proceso de etiquetado

El objetivo del script fue etiquetar cada noticia con una de las tres categorías específicas a partir de listas de los patrones ya descritos. El script implementó una función para buscar coincidencias en los textos procesados utilizando la biblioteca *re* de *Python*.

Se buscaron coincidencias para cada tipo de crimen y cada versión del texto (contenido original, sin ruido, lematizado y derivado). Los resultados se almacenaron en nuevas columnas del DataFrame, indicando el tipo de crimen y la versión del texto.

Posteriormente, se revisaron los resultados para descartar tanto noticias sin coincidencias como noticias con coincidencias en más de una categoría, asegurando que cada noticia pertenezca únicamente a una categoría. Esto ayuda al modelo a aprender patrones claros y específicos asociados con cada categoría, mejorando la precisión de la clasificación. Se asignó una etiqueta numérica a cada noticia: 1 para asalto, 2 para homicidio y 3 para secuestro.

Finalmente, el DataFrame modificado se guardó en un nuevo archivo CSV con las noticias clasificadas listas para el siguiente paso.

En esta fase, se trabajó exclusivamente con las noticias que coincidieron con ciertos patrones en la columna "LEMATIZADO". Esta columna se seleccionó porque contenía la versión lematizada de las palabras, lo que aumentó significativamente la probabilidad de encontrar coincidencias. Los patrones aplicados fueron específicamente diseñados para identificar estas formas básicas de las palabras. Se registraron 24,329 noticias que presentaron al menos una correspondencia con las categorías establecidas. Después de excluir los artículos que coincidieron con más de una categoría quedaron únicamente 7,824 noticias distribuidas conforme las categorías mostradas en la Tabla 2.

Tabla 2. Distribución del conjunto definitivo de noticias seleccionado para el entrenamiento.

| ASALTO | HOMICIDIO | SECUESTRO |
|---|---|---|
| 4,097 noticias | 3,422 noticias | 305 noticias |

## 3.4 Clasificación de noticias mediante aprendizaje automático

La tarea de clasificación de noticias se implementó utilizando cuatro algoritmos de la librería *scikit-learn* de *Python*: *Random Forest, Support Vector Machine, Naive Bayes y Gradient Boosting.*

Los datos se cargaron desde un archivo CSV y se dividieron en conjuntos de entrenamiento y prueba en una proporción del 80% y 20%, respectivamente. Las características (texto derivado) y las etiquetas (categorías) se separaron para su posterior procesamiento. Para la vectorización del texto, se utilizó TF-IDF (*Term Frequency-Inverse*

*Document Frequency*), configurando el vectorizador para utilizar un máximo de 1000 características.

Cada algoritmo tuvo una configuración específica de hiperparámetros. El algoritmo *Random Forest* se configuró con 100 estimadores y un estado aleatorio de 42 para asegurar la reproducibilidad de los resultados. El algoritmo *Support Vector Machine* utilizó un kernel lineal y un parámetro de regularización C=1.0, además de un estado aleatorio de 42. Para *Naive Bayes*, se empleó el clasificador *MultinomialNB*, adecuado para datos discretos como las frecuencias de palabras en documentos de texto. El algoritmo *Gradient Boosting* se configuró con 100 estimadores, una tasa de aprendizaje de 1.0, una profundidad máxima de 1 y un estado aleatorio de 42.

Estos algoritmos fueron evaluados en términos de exactitud para determinar su efectividad en la clasificación de noticias.

## 4 Resultados

En la *Tabla* 4 de esta sección se presentan los resultados obtenidos a partir de la implementación de los cuatro algoritmos de clasificación de noticias: *Random Forest (RF), Support Vector Machine (SVM), Naive Bayes (NB) y Gradient Boosting (GB)*. Se muestra el desempeño de cada modelo en términos de exactitud permitiendo evaluar la efectividad de cada algoritmo en la clasificación de noticias y compararlos para determinar cuál ofrece el mejor rendimiento en este contexto.

Para este paso, los datos se organizaron en los tres distintos grupos que se muestran en la Tabla 3, asegurando una variedad en la distribución que permitiera una validación rigurosa.

Tabla 3. Distribución de las noticias en los tres grupos de datos utilizados para el entrenamiento de los algoritmos de clasificación.

| **Grupo** | **Asalto** | **Homicidio** | **Secuestro** |
|---|---|---|---|
| 1. Conjunto Completo de Datos | 4,097 noticias | 3,422 noticias | 305 noticias |
| 2. Conjunto de Datos Equilibrados | 400 noticias | 400 noticias | 305 noticias |
| 3. Solo las Categorías Predominantes | 4,097 noticias | 3,422 noticias | No aplicable |

Tabla 4: Resultados de la Exactitud de los Algoritmos de Clasificación

| Datos de entrada | Algoritmo | Exactitud | | |
|---|---|---|---|---|
| | | **Grupo 1** | **Grupo 2** | **Grupo 3** |
| *Stemming* | RF | 0.939258 | 0.93665 | 0.93879 |
| | SVM | 0.900895 | 0.80995 | 0.90619 |
| | NB | 0.785166 | 0.77828 | 0.79441 |
| | GB | 0.912404 | 0.93213 | 0.95542 |
| *Lematizado* | RF | 0.974425 | 0.95475 | 0.97139 |
| | SVM | 0.934783 | 0.81448 | 0.94478 |
| | NB | 0.808824 | 0.75113 | 0.82435 |
| | GB | 0.957801 | 0.93213 | 0.97804 |

# 5 Discusión

Los resultados de la clasificación, detallados en las tablas correspondientes, muestran la exactitud de los algoritmos *Random Forest*, SVM, *Naive Bayes y Gradient Boosting*, utilizando técnicas de *stemming* y *lematización*. Los datos se organizaron en tres grupos: conjunto completo de datos, conjunto de datos equilibrados y solo en las categorías predominantes.

En términos de exactitud, *Random Forest* (RF) demostró un rendimiento superior en todos los grupos de datos, alcanzando una precisión del 97.14% en el grupo 3 con datos *lematizados*. Por otro lado, *Naive Bayes* (NB) tuvo el rendimiento más bajo, posiblemente debido a su suposición de independencia entre características, que puede no ser válida en el contexto de las noticias.

Comparando los algoritmos, *Random Forest* obtuvo el mejor promedio de exactitud (0.952), seguido de *Gradient Boosting* (0.942), *Support Vector Machine* (0.880) y *Naive Bayes* (0.788). El alto rendimiento de *Random Forest* se atribuye a su capacidad para combinar múltiples árboles de decisión y manejar datos ruidosos y valores atípicos, mejorando la precisión general. En contraste, *Naive Bayes*, aunque eficiente, puede verse afectado negativamente por su simplicidad y suposiciones de independencia.

El impacto del preprocesamiento de texto también se analizó, mostrando que la *lematización* obtuvo un mejor desempeño en comparación con el *stemming*. La *lematización*, al reducir las palabras a su forma base considerando el contexto y la gramática, mejora la precisión de los modelos de clasificación. En cambio, el *stemming*, aunque más rápido, puede distorsionar el significado de las palabras al cortar sus raíces sin considerar el contexto.

Los promedios de exactitud para los diferentes grupos de datos fueron: Grupo 1 (conjunto completo) con 0.902, Grupo 2 (datos equilibrados) con 0.864 y Grupo 3 (categorías predominantes) con 0.914. Estos resultados indican que el Grupo 3 tuvo el mejor

desempeño, posiblemente debido a la exclusión de la categoría de secuestro, simplificando así la clasificación. El Grupo 1 también mostró un buen rendimiento, mientras que el Grupo 2 tuvo el menor promedio de exactitud debido a la menor cantidad de datos.

En resumen, la *lematización* y el uso del algoritmo *Random Forest* proporcionaron los mejores resultados en la clasificación de noticias para identificar eventos criminales, con un promedio general de exactitud de 0.891 considerando todos los grupos de datos.

En cuanto a la comparación con trabajos existentes en la literatura en el presente trabajo, *Random Forest* (RF) mostró un rendimiento alto y constante en todos los grupos, alcanzando su mayor exactitud con *lematización* en el Grupo 1 (97.44%). Comparativamente, el estudio de Tabassoum y Akber (2024) encontró que RF fue el mejor modelo con una precisión del 91.48%.

*Support Vector Machine* (SVM) varió significativamente en rendimiento. En nuestro estudio, tuvo una precisión más baja con datos equilibrados (Grupo 2), pero alcanzó su mejor rendimiento con lematización en el Grupo 3 (94.48%). Esto es similar al estudio de Leonard et al. (2022), donde SVC alcanzó una precisión del 96.34%.

*Naive Bayes* (NB) tuvo un rendimiento moderado, con su mejor exactitud en *lematización* para el Grupo 3 (82.44%). En comparación, Veziroğlu et al. (2024) mostraron precisiones de hasta 98.31% en algunos casos, destacando su variabilidad según el tipo de datos y preprocesamiento.

*Gradient Boosting* (GB) mostró un rendimiento notable, especialmente con *lematización* en el Grupo 3 (97.80%). Esto es comparable al estudio de N. N y C. J (2024), donde *BERT* combinado con técnicas de *boosting* alcanzó una precisión del 98%.

En general, nuestros resultados son consistentes con estudios anteriores. La precisión varía según el preprocesamiento y la distribución de datos. La *lematización* mejoró significativamente la exactitud de todos los algoritmos, alineándose con estudios como los de P. S et al. (2023) y S. B y Santhanalakshmi (2023).

Una limitación principal del estudio fue el desbalance en la cantidad de noticias por categoría, especialmente en secuestro, lo que afectó la precisión de los modelos de clasificación. Además, al usar datos exclusivamente de El Universal, se restringió la diversidad de fuentes, lo que puede limitar la generalización de los hallazgos y no capturar completamente la variedad de eventos criminales en otros contextos o regiones.

## 6 Conclusiones y trabajo a futuro

El objetivo de este artículo consistió en implementar un sistema para la recuperación y clasificación de noticias provenientes de un medio digital específico, orientado a la identificación precisa de eventos criminales. Este proceso se estructuró en cuatro fases: primero, la recolección automática de noticias desde el medio digital objetivo; segundo, el procesamiento del texto mediante técnicas avanzadas de procesamiento de lenguaje natural (PLN) para transformar y preparar los datos textuales, tercero el etiquetado inicial de las noticias limpias utilizando patrones específicos definidos mediante expresiones regulares

(regex), y cuarto, la aplicación de algoritmos de aprendizaje supervisado, seleccionados y configurados para la clasificación efectiva de las noticias en las categorías de robo, homicidio, y secuestro. Los algoritmos implementados incluyeron *Random Forest, Support Vector Machine (SVM), Naive Bayes y Gradient Boosting*. Este enfoque integrador no solo buscó mejorar la eficiencia en la identificación y clasificación de eventos delictivos, sino también sentar las bases para investigaciones futuras en la optimización de métodos de PLN y la exploración de nuevos algoritmos de aprendizaje automático para tareas similares.

La contribución de este artículo radica en la implementación exitosa de un sistema automatizado para la recuperación y clasificación de noticias criminales a partir de un medio digital específico. Se destacan tres aspectos principales de esta investigación: primero, la aplicación efectiva de técnicas de procesamiento de lenguaje natural, como la *lematización* y la *vectorización* TF-IDF, para preparar los datos textuales; segundo, la evaluación de varios algoritmos de aprendizaje supervisado, entre ellos *Random Forest*, SVM, *Naive Bayes y Gradient Boosting*, demostrando que *Random Forest* obtuvo el mejor rendimiento; y tercero, el análisis del impacto del balance de datos en la precisión de los modelos, subrayando la importancia de un conjunto de datos equilibrado para mejorar la capacidad predictiva de los sistemas de clasificación automatizados en el ámbito del análisis de noticias criminales.

Para futuras investigaciones, se sugiere ampliar la recolección de noticias a otros portales y explorar diferentes idiomas para enriquecer la diversidad del conjunto de datos. Además, sería beneficioso incorporar técnicas avanzadas de caracterización de datos, como *Word2vec*, para mejorar la representación semántica de las noticias. También se recomienda explorar el uso de redes neuronales y modelos de aprendizaje profundo, como *BERT*, para la clasificación, aprovechando su capacidad para capturar relaciones complejas y contextuales en el texto, lo cual podría potenciar la precisión y robustez del sistema de clasificación de eventos criminales.

## Referencias


Agarwal, J., Christa, S., Pai H, A., Kumar, M. A., y P. M. S. G. (2023). "Machine Learning Application for News Text Classification," *2023 13th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Noida, India, 2023, pp. 463-466. doi: 10.1109/Confluence56041.2023.10048856.

B, S., y Santhanalakshmi, S. (2023). "News Article Topic Classification Using Embeddings," *2023 14th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Delhi, India, 2023, pp. 1-7. doi: 10.1109/ICCCNT56998.2023.10306603.

El Universal. (n.d.). Recuperado el 2 de julio de 2024, de https://www.eluniversal.com.mx/

Hossain, M. M., Chowdhury, Z. R., Rezwanul Haque Akib, S. M., Ahmed, M. S., Hossain, M. M., y Miah, A. S. M. (2023). "Crime Text Classification and Drug Modeling from Bengali News Articles: A Transformer Network-Based Deep Learning Approach," *2023 26th International Conference on Computer and Information Technology (ICCIT)*, Cox's Bazar, Bangladesh, 2023, pp. 1-6. doi: 10.1109/ICCIT60459.2023.10441195.

Leonard, G., Sisnadi, F., Wardhana, N. V., Aziz Al-Ghofari, M. A., y Girsang, A. S. (2022). "News Classification Based On News Headline Using SVC Classifier," *2022 16th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, Lombok, Indonesia, 2022, pp. 1-4. doi: 10.1109/TSSA56819.2022.10063879.

N. N., y C. J. (2024). "Hybrid Approach for Multi-Classification of News Documents Using Artificial Intelligence," *2024 5th International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, Tirunelveli, India, 2024, pp. 466-473. doi: 10.1109/ICICV62344.2024.00079.

P. S., S. M., V. K. S., y S. N. S. (2023). "News Category Classification using Natural Language Processing Transformer," *2023 Second International Conference on Augmented Intelligence and Sustainable Systems (ICAISS)*, Trichy, India, 2023, pp. 1185-1189. doi: 10.1109/ICAISS58487.2023.10250566.

Rahman, M. M., Khan, M. A. Z., y Biswas, A. A. (2021). "Bangla News Classification using Graph Convolutional Networks," *2021 International Conference on Computer Communication and Informatics (ICCCI)*, Coimbatore, India, 2021, pp. 1-5. doi: 10.1109/ICCCI50826.2021.9402567.

Sreekala, K., Sivajyothi, M., Chiranjevi, D., Sunitha, G., y Swetha, M. (2024). "A Novel Integration of Hierarchical Clustering and Ensemble Classification Algorithms for News Classification," 2024 International Conference on Cognitive Robotics and Intelligent Systems (ICC - ROBINS), Coimbatore, India, 2024, pp. 524-528. doi: 10.1109/ICC-ROBINS60238.2024.10533963.

Tabassoum, N., y Akber, M. A. (2024). "Interpretability of Machine Learning Algorithms for News Category Classification Using XAI," *2024 6th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT)*, Dhaka, Bangladesh, 2024, pp. 770-775. doi: 10.1109/ICEEICT62016.2024.10534385.

Veziroğlu, M., Veziroğlu, E., y Bucak, İ. Ö. (2024). "Performance Comparison between Naive Bayes and Machine Learning Algorithms for News Classification," *Bayesian Inference - Recent Trends*, IntechOpen, Jan. 17, 2024. doi: 10.5772/intechopen.1002778.

# Capítulo 5

## Detección de acoso en redes sociales a través de redes convolucionales para grafos

Ana Laura Lezama Sánchez[1], Mireya Tovar Vidal[2]

Benemérita Universidad Autónoma de Puebla
[1] Escuela de Artes Plásticas y Audiovisuales
[2] Facultad de Ciencias de la Computación

yumita1102@gmail.com, mireya.tovar@correo.buap.mx

**Resumen.** El acoso cibernético ha proliferado con el crecimiento exponencial de Internet, convirtiéndose en una preocupación omnipresente en la era digital. El impacto psicológico puede ser profundo, afectando la autoestima, la salud mental y el bienestar emocional de quienes lo sufren. El acoso cibernético también puede tener consecuencias legales y laborales, especialmente cuando implica difamación, discriminación o violación de la privacidad. Las víctimas pueden enfrentarse a problemas legales, perder oportunidades laborales o sufrir daños en su reputación debido a la difusión de información falsa o comprometedora. Por tanto, es esencial brindar atención y tomar medidas para prevenir, detectar y abordar el acoso cibernético. En este trabajo, se propone la detección de mensajes con potencial de acoso en redes sociales en español. Los elementos por emplear son una combinación de técnicas avanzadas como la clasificación de grafos a través de Graph Convolutional Network (GCN), con el objetivo de identificar patrones de comportamiento nocivos. La clasificación de los mensajes se realizará en tres categorías: positiva (presencia de acoso), negativa (ausencia de acoso) y neutra (mensajes ambiguos), lo que permitirá una mejor comprensión y estudio del problema del acoso cibernético en línea.

**Palabras Clave:** acoso cibernético, procesamiento del lenguaje natural, grafos.

## 1 Introducción

El auge de las redes sociales ha transformado la forma en que nos comunicamos, pero también ha dado lugar a nuevos desafíos, entre ellos, el ciberacoso. Si bien las plataformas de redes sociales ofrecen una ventana única a la diversidad y la conectividad humana, son el escenario de comportamientos abusivos y dañinos. La omnipresencia de Internet en la era digital ha amplificado este fenómeno, convirtiéndolo en una preocupación significativa en la sociedad contemporánea.

El acoso, en sus diversas formas, ha sido un problema social persistente a lo largo de la historia. Desde el acoso escolar hasta el acoso laboral, ha afectado a personas en distintos ámbitos de la vida. Sin embargo, con la llegada de la era digital, el ciberacoso ha emergido como una forma única y especialmente preocupante de hostigamiento, que se manifiesta a través de múltiples plataformas en línea, como redes sociales, foros y mensajería instantánea, y que genera graves consecuencias para sus víctimas.

El ciberacoso presenta una serie de riesgos, ya que es capaz de alcanzar a las personas en cualquier momento y lugar, sin límites geográficos. Además, la facilidad con la que se difunde información en línea aumenta la velocidad y el alcance del acoso, lo que lo hace aún más pernicioso. Las personas más susceptibles de sufrir ciberacoso son, en muchos casos, los adolescentes y jóvenes adultos, que son usuarios ávidos de las redes sociales y a menudo son víctimas de la intimidación y el hostigamiento en línea.

Los problemas causados por el ciberacoso son diversos y graves. A nivel psicológico, provoca ansiedad, depresión, baja autoestima e incluso pensamientos suicidas en sus víctimas. Socialmente, lleva al aislamiento y dificultades en las relaciones interpersonales. Además, el ciberacoso puede tener consecuencias legales y laborales, especialmente cuando implica difamación, discriminación o violación de la privacidad.

En respuesta a este desafío, la inteligencia artificial ha comenzado a desempeñar un papel crucial en la detección y prevención del ciberacoso. Algoritmos avanzados de aprendizaje automático y análisis de datos se utilizan para identificar patrones de comportamiento abusivo en línea y tomar medidas preventivas para proteger a los usuarios vulnerables.

En este artículo, proponemos la clasificación de mensajes potencialmente abusivos en redes sociales en español mediante la representación de cada clase como grafos individuales. Estos grafos se construyen a partir de un conjunto de palabras específicas del documento, interconectadas según su coocurrencia en el texto. Utilizamos *word2vec*, un modelo de aprendizaje automático que convierte palabras en vectores numéricos, preservando sus relaciones semánticas y contextuales. Para capturar el significado profundo de las palabras, cada nodo del grafo se asocia con una representación vectorial obtenida mediante *word2vec*. Esta representación permite que cada nodo tenga atributos que reflejan el significado intrínseco de las palabras, facilitando un análisis más preciso y significativo de las relaciones entre términos en el texto.

El resto del artículo está organizado de la siguiente manera: en el capítulo 2 se exponen los conceptos relevantes que dan soporte a esta investigación. En la sección 3 se muestran algunos de los trabajos existentes en la detección del ciberacoso. La sección 4 presenta la metodología empleada en la detección de acoso cibernético mientras que la sección 5 presenta los resultados obtenidos por la metodología propuesta. Las conclusiones, así como los trabajos a futuro se exponen en la sección 6. Finalmente, se presentan las referencias consultadas durante el desarrollo de este trabajo.

## 2 Marco teórico

En esta sección se introducen algunos conceptos relevantes que se aplicaron en el desarrollo de esta investigación.

El psicólogo noruego Olweus (Olweus et al., 2010), pionero en la investigación sobre el acoso escolar, define el acoso como un comportamiento negativo y agresivo que es intencional, repetido y que implica un desequilibrio de poder entre el acosador y la víctima. Este tipo de comportamiento no se limita al entorno escolar, ya que se manifiesta de diversas formas en otros contextos como el acoso laboral, el acoso sexual y el ciberacoso. El acoso laboral, también conocido como *mobbing*, implica comportamientos abusivos que tienen como objetivo intimidar o aislar a un individuo en el entorno laboral (Leymann, 1990). Por otro lado, el acoso sexual se refiere a comportamientos no deseados de naturaleza sexual que crean un ambiente hostil para la víctima (Fitzgerald et. al., 1995). Finalmente, el ciberacoso, o acoso en línea, utiliza tecnologías de comunicación digital para hostigar, intimidar o difamar a otros a través de medios electrónicos como redes sociales o mensajes de texto (Hinduja et. al., 2017).

El análisis del acoso cibernético o ciberbullying ha sido abordado utilizando varias técnicas de Procesamiento del Lenguaje Natural (PLN), Aprendizaje Automático (ML) y Aprendizaje Profundo (DL). Estas técnicas se han utilizado para identificar patrones de comportamiento abusivo en el lenguaje y para desarrollar sistemas de detección automática de mensajes de acoso en línea (Raj et. al., 2021).

En el estado del arte podemos encontrar trabajos donde se emplean técnicas como el análisis de sentimientos para identificar el tono emocional de los mensajes, la extracción de características lingüísticas para detectar el uso de lenguaje ofensivo o intimidatorio, y la clasificación de texto para etiquetar mensajes como positivos, neutrales o negativos en términos de su contenido (Mercado et. al., 2018).

Por otro lado, el aprendizaje automático ha sido utilizado para entrenar modelos predictivos capaces de identificar automáticamente el acoso cibernético en función de características propias del texto. Algunos de los algoritmos de clasificación que se han aplicado son *Support Vector Machines* (SVM), *Naive Bayes*, y *Random Forests*, entre otros, para clasificar los mensajes como acoso o no acoso (Lemus et. al., 2022).

Además, el aprendizaje profundo ha emergido como una herramienta poderosa para el análisis del acoso cibernético, especialmente en tareas de procesamiento de texto. Las redes neuronales convolucionales (CNN) y las redes neuronales recurrentes (RNN) como las redes LSTM (*Long Short-Term Memory*) y GRU (*Gated Recurrent Unit*), se han utilizado para capturar relaciones complejas en el texto y aprender representaciones de alta calidad para distinguir entre mensajes de acoso y mensajes no abusivos con mayor precisión.

La aplicación de técnicas de Procesamiento del Lenguaje Natural (PLN), aprendizaje automático y aprendizaje profundo ha representado un avance significativo en la detección y análisis del acoso cibernético. Estos enfoques han permitido el desarrollo de modelos computacionales capaces de detectar este tipo de problemas en cualquier conjunto de texto, principalmente en español e inglés (Afrifa et. al., 2022).

Los modelos de clasificación utilizados para abordar el problema del acoso cibernético son evaluados mediante métricas como precisión, exactitud, recall y $F_1$. Estas métricas proporcionan una medida objetiva de la efectividad de los modelos en la identificación y clasificación precisa de mensajes de acoso en línea. La precisión representa la proporción de mensajes correctamente identificados como acoso entre todos los mensajes clasificados como tal. La exactitud, por su parte, mide la proporción de mensajes correctamente clasificados, ya sean de acoso o no. El *recall*, también conocido como exhaustividad, indica la proporción de mensajes de acoso que son correctamente identificados por el modelo entre todos los mensajes de acoso presentes en el conjunto de datos. Por último, el $F_1$ combina la precisión y el *recall* en una sola métrica, proporcionando una medida equilibrada del rendimiento del modelo en términos de cuántos falsos positivos como falsos negativos existen en el modelo obtenido. Estas métricas son fundamentales para evaluar la efectividad de los modelos de clasificación y guiar el desarrollo de sistemas más precisos y confiables para abordar el problema del acoso cibernético.

## 3 Trabajos relacionados

En esta sección se exponen los trabajos relacionados en el mismo campo. Algunos autores incorporaron en sus metodologías la detección del ciberacoso, abordando aspectos específicos como el elemento del sarcasmo, técnicas estadísticas para analizar datos provenientes de redes sociales, regresión lineal, análisis de sentimientos, entre otros.

En el estudio de Recalde Monar (Recalde Monar et. al., 2021), examinan casos de ciberacoso en Ecuador de los últimos cinco años, utilizando técnicas estadísticas para analizar datos de ciberacoso provenientes de redes sociales. El objetivo era identificar deficiencias en la detección por parte de las autoridades reguladoras. Los autores encontraron un aumento significativo en las denuncias de violación a la intimidad y pornografía adolescente. Además, proyectaron un crecimiento del 70% en los casos de ciberacoso para el año 2025. Por lo que los autores sugirieron la necesidad de abordar estos riesgos en las redes sociales mediante algoritmos, arquitecturas, software o procesos. La información analizada la obtuvieron de la fiscalía general del Estado basada en leyes de acceso a la información pública a la que le aplicaron regresión lineal para proyectar las denuncias futuras. La tendencia mostró un aumento anual promedio del 13.45%.

Por otro lado, el estudio realizado por (Llampa et al., 2020) presenta la brecha en el procesamiento del lenguaje natural entre el inglés y el español. El objetivo de los autores fue desarrollar un modelo automático efectivo en español para el análisis de sentimientos en comentarios de videos de *YouTube*. Para ello, proponen un análisis para detectar eficazmente casos de violencia verbal, identificando tanto a las víctimas como a sus agresores, lo que permite tomar decisiones informadas. Dicho análisis combinó el análisis de sentimientos con un algoritmo de aprendizaje supervisado, específicamente *Support Vector Machine* (SVM). El proceso implica etapas de tratamiento de texto adaptadas al español, selección y entrenamiento del modelo SVM. Como lenguaje de programación

utilizaron Python junto con bibliotecas como *spacy, scikit-learn* y *matplotlib* para el desarrollo, análisis y visualización de datos. Los comentarios de *YouTube* los obtuvieron a través de la *YouTube Data API* de *Google*. Los autores destacaron la importancia de la transformación de datos textuales a representaciones numéricas, conocida como *feature extraction*, donde la vectorización de palabras (*Word Embeddings*) se utiliza para convertir cada comentario en un vector compatible con los modelos de aprendizaje automático. Además, sugieren considerar opciones más avanzadas, como las redes neuronales, para alcanzar niveles de complejidad más altos.

El trabajo presentado por Mamani Apaza (Mamani Apaza, 2020) tiene como objetivo utilizar la Inteligencia Artificial para identificar conversaciones conflictivas que afectan a niños y jóvenes en distintas partes del mundo. Sin embargo, un reto que los autores encontraron fue la evolución de las complejidades lingüísticas como las jergas y el argot, utilizadas para perpetuar la violencia virtual. Además, señalan que el estado de ánimo del emisor de los mensajes influye en el significado semántico de los textos. Por lo tanto, los autores consideraron el análisis del contexto de los textos, incluso cuando están escritos de manera confusa. Por lo que emplearon procedimientos para el procesamiento de lenguaje natural y se procesó la información adecuadamente, para su análisis entrenando una Red Neuronal Recurrente que permita comprender mejor las formas de expresión escrita utilizadas por las personas en su comunicación.

El estudio de Al-Garadi et al. (2019) ofrece una revisión exhaustiva del proceso de detección del ciberacoso en redes sociales. Los autores llevaron a cabo una revisión de la literatura existente para identificar comportamientos agresivos en sitios web de redes sociales mediante enfoques de aprendizaje automático. En su análisis, abordaron cuatro aspectos fundamentales: la recolección de datos, la ingeniería de características, la construcción del modelo de detección de ciberacoso y la evaluación de los modelos desarrollados. Además, resumieron diversos tipos de características discriminativas empleadas para detectar ciberacoso en línea y evaluaron los clasificadores de aprendizaje automático supervisado más efectivos para esta tarea. Una contribución clave del estudio fue la definición de métricas de evaluación específicas para medir el rendimiento de los modelos. Estas métricas permitieron a los autores identificar parámetros significativos y comparar diversos algoritmos de aprendizaje automático.

El artículo propuesto por Hani (Hani et. al., 2019) presenta un enfoque de aprendizaje automático supervisado para detectar y prevenir el ciberacoso. Este enfoque constó de tres pasos principales: preprocesamiento, extracción de características y clasificación. En el paso de preprocesamiento, se limpiaron los datos eliminando el ruido y el texto innecesario. Luego, en la extracción de características, los datos textuales los transformaron en un formato adecuado para los algoritmos de aprendizaje automático, utilizando *TFIDF* para extraer características y análisis de sentimientos para agregar la polaridad de las oraciones. Además, emplearon *NGram* para considerar diferentes combinaciones de palabras durante la evaluación del modelo. Los autores utilizaron los clasificadores de máquinas de soporte vectorial y una red neuronal para entrenar y reconocer acciones de acoso. La evaluación del enfoque propuesto mostró un rendimiento superior, logrando una precisión del 92.8%.

En Ali (Ali et. al., 2020) se expone un enfoque para detectar el ciberacoso, incluyendo el sarcasmo en él. Por lo tanto, propusieron la recopilación de datos utilizando conjuntos de datos etiquetados disponibles públicamente en Internet. Antes de aplicar el preprocesamiento, extrajeron algunas características que requieren que los datos estén en su forma original. Estas características incluyeron puntuación de sentimiento y profanidad para la detección de ciberacoso, y varios conteos de características para la detección de sarcasmo. Luego, aplicaron el preprocesamiento, que incluyó la eliminación de caracteres especiales, caracteres únicos, espacios múltiples y palabras vacías, y la conversión del texto a minúsculas. Posteriormente, llevaron a cabo la clasificación de los conjuntos de datos aplicando los algoritmos de aprendizaje automático como *SVM* (máquinas de soporte vectorial), Bayes ingenuo y Bosque Aleatorio. Los resultados demostraron que el clasificador *SVM* tuvo un mejor rendimiento que el resto, con una precisión promedio del 79%, siendo este el mayor valor obtenido.

## 4 Aproximación propuesta

En esta sección se presentará la aproximación propuesta para detectar si un mensaje tiene la intención de acosar al receptor (positivo), si es un mensaje con algún propósito distinto al acoso (negativo), o si simplemente es un mensaje sin ningún objetivo en particular (neutro).

La arquitectura propuesta se presenta en la Figura 1 e incluye las siguientes etapas: la recopilación de conjunto de datos, detallada en la Tabla 1 y el preprocesamiento de estos con el fin de homogenizar el texto. La creación de grafos no dirigidos para capturar el contexto de las palabras y frases dentro de la estructura de nodos y relaciones, lo que facilita una comprensión más completa del significado y la intención del mensaje. Por otro lado, para la clasificación de estos grafos se empleó una red convolucional para grafos. Finalmente, los resultados obtenidos son evaluados utilizando métricas estándar como precisión, *recall*, exactitud y medida $F_1$.

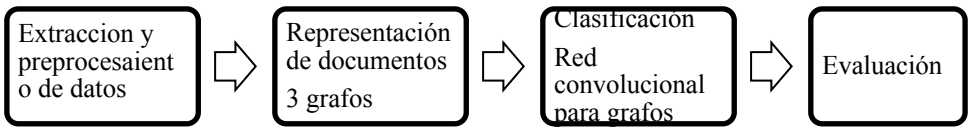


*Figura 1: Arquitectura propuesta*

La aproximación propuesta está compuesta de 4 fases, las cuales están formadas de las siguientes tareas:

- Recopilación de conjunto de datos y preprocesamiento:

- El conjunto de datos se descargó de la base de datos (Kaggle, 2022). Este conjunto está formado por tres clases:
  - Positivo: 432 documentos
  - Negativo: 211 documentos
  - Neutro: 359 documentos

  Con un total de 1,002 documentos.
  Ejemplo:
  - Positivo: “Me pondré en contacto con algún servicio de emergencias para poder ayudarte. NO ESTÁS SOLA”
  - Negativo: “no la conozco ni en pelea de gatos”
  - Neutro: “Veis como contestáis vosotros. Por eso lo digo, igualdad, ni feminismo ni machismo, ya esta bien”
- Preprocesamiento de datos: El conjunto de datos fue sometido a:
  - Sustitución de emoticonos por palabras: en los documentos fueron reemplazados por palabras descriptivas que reflejan su significado. Por ejemplo, 😄 se sustituyó por la palabra sonrisa.
  - Eliminación de Signos de Puntuación: Se eliminaron todos los signos de puntuación (como puntos, comas, signos de interrogación, etc.) para evitar que estos signos influyan en el análisis del texto, ya que no aportan información relevante.
  - Eliminación de Menciones: Se eliminaron las menciones, que son etiquetas que identifican a usuarios específicos (por ejemplo, "@usuario"). Las menciones se eliminaron para centrar el análisis en el contenido del texto y no en las personas mencionadas.
  - Eliminación de Retweets (RT): Se eliminaron los indicadores de retweets (RT) y cualquier otro contenido que indique que el texto es una repetición de un mensaje. Esto se hace para evitar duplicaciones y concentrarse en el contenido original del mensaje.
  - Eliminación de Números: Los números fueron eliminados del texto, ya que en muchos casos no aportan información relevante para el análisis del contenido textual.
  - Conversión de Mayúsculas a Minúsculas: Todos los caracteres en mayúsculas se convirtieron a minúsculas. Esto se hace para asegurar que el modelo trate las palabras de manera uniforme, sin considerar las diferencias de mayúsculas y minúsculas.

  Ejemplo de documentos preprocesados:
  - Positivo: “pondré contacto servicio emergencias poder ayudarte estás sola”
  - Negativo: “conozco pelea gatos”

        - Neutro: “veis contestáis vosotros digo igualdad feminismo ni machismo esta bien”

- Grafos
    - Se generó un grafo para cada clase en el corpus, es decir, se construyeron un total de tres grafos. Los grafos fueron creados siguiendo los pasos detallados a continuación:
        - Generación de bigramas: Tras el preprocesamiento del texto, se procede a la generación de bigramas. Los bigramas son pares de palabras consecutivas extraídas del texto, generados mediante la función *n*-grams de NLTK. Esta etapa es crucial para capturar las relaciones contextuales inmediatas entre palabras. La creación de bigramas permite construir las conexiones entre nodos en el grafo, facilitando el análisis de la estructura semántica del texto.
        - Creación de grafo a partir de bigramas: Con los bigramas generados, se crea un *dataFrame*, que se almacena en un archivo CSV. Este *dataFrame* organiza las conexiones entre nodos en el grafo, donde cada fila representa un borde (*edge*). El *dataFrame* está compuesto por las siguientes columnas:
            - *source*: Indica el nodo inicial del borde.
            - *target*: Indica el nodo final del borde.

      Esta estructura tabular facilita la conversión de los datos de texto en una representación gráfica, permitiendo una visualización y análisis más claros de las relaciones entre palabras.
- Conversión a la Representación StellarGraph

  Una vez construidos los grafos, se convierten en un formato compatible con la biblioteca StellarGraph. La cual permite trabajar con grafos enriquecidos con características de nodos y bordes. Cada grafo se representa como un objeto StellarGraph, en el cual, los nodos tienen características asignadas, y las aristas reflejan las conexiones basadas en los datos de bigramas previamente generados.
- Clasificación
    - Se construye un modelo de clasificación de grafos utilizando redes neuronales convolucionales para grafos (GCN). Este modelo se entrena para clasificar los grafos en diferentes categorías basadas en las características de los nodos y las relaciones entre ellos. La red neuronal se compone de varias capas convolucionales, seguidas de capas densas que permiten la clasificación final. Se emplea la técnica de EarlyStopping para detener el entrenamiento cuando el rendimiento en el conjunto de validación comienza a degradarse, asegurando que el modelo generalice bien.
- Evaluación del Modelo y Predicción

- Los resultados obtenidos después del entrenamiento realizado, el modelo se evalúa utilizando el conjunto de validación. Los resultados se evalúan empleando las métricas de evaluación precisión, exactitud, exhaustividad y $F_1$.

# 5 Resultados y discusión

En esta sección se presentan los resultados obtenidos en la clasificación de mensajes con acoso, sin acoso o es tema neutro.

## 5.1. Conjunto de datos

El conjunto de datos fue obtenido de la base de datos k*aggle* (ver la Tabla 1) compuesta de 1,002 mensajes 432 positivos, 211 negativos y 359 neutros.

*Tabla 1 Conjunto de datos*

| Clase | Número de mensajes |
|---|---|
| Positivo | 432 |
| Negativo | 211 |
| Neutro | 359 |

## 5.2. Resultados y discusión

En la Tabla 2 se presentan tres ejemplos de mensajes originales que forman parte del corpus antes del preprocesamiento, correspondientes a las categorías positiva, negativa y neutra. Además, se muestra el texto de cada mensaje después de aplicar el preprocesamiento descrito en la sección de Aproximación Propuesta.

*Tabla 2 Ejemplo de textos etiquetados según la clase a la que pertenecen*

| Clase | Sin preprocesamiento | Con preprocesamiento |
|---|---|---|
| Positivo | @jmartineze_ Haber como apoyaron a mayra goñi... También que saquen este tipo de acciones que dejan mucho que desear en que cerebro podrido cabe escribir de esa manera🤦🤦 | haber como apoyaron a mayra goñi también que saquen este tipo de acciones que dejan mucho que desear en que cerebro podrido cabe escribir de esa manera confuso confuso |
| Negativo | @Motta01Alonso @munchii_tx Alonso mejor hablemos de la sacada de pate a mayra goñi | alonso mejor hablemos de la sacada de pate a mayra goñi |
| Neutro | @diarioojo Lamentable de Mayra Goñi,,, si haces directo hazlo en privado | lamentable de mayra goñi si haces directo hazlo en privado con la |

| | | |
|---|---|---|
| | con la gente que te conoce... El hecho que sea público te expones a esos eventos. | gente que te conoce el hecho que sea público te expones a esos eventos |

La Tabla 3 expone los resultados algunas de las clases recuperadas después de llevar a cabo la clasificación. En cada clase se observa que la etiqueta asignada por el modelo de clasificación empleado acierta con la predefinida en los datos originales.

*Tabla 3 Ejemplo de las etiquetas predichas con los modelos propuestos*

| **Clase Original** | **Clase predicha** | **Corpus con preprocesamiento** |
|---|---|---|
| Positivo | 1 (positivo con acoso) | haber como apoyaron a mayra goñi también que saquen este tipo de acciones que dejan mucho que desear en que cerebro podrido cabe escribir de esa manera confuso confuso |
| Negativo | 0 (negativo sin acoso) | alonso mejor hablemos de la sacada de pate a mayra Goñi |
| Neutro | 2 (neutro otros temas) | lamentable de mayra goñi si haces directo hazlo en privado con la gente que te conoce el hecho que sea público te expones a esos eventos |

La Tabla 4 expone los resultados obtenidos de la clasificación del corpus. Los resultados obtenidos fueron evaluados con las métricas precisión, *exhaustividad, exactitud* y $F_1$. Estas medidas se calcularon utilizando un promedio ponderado que considera la distribución de las clases en el conjunto de datos. Por lo que hemos llegado a la conclusión que los resultados obtenidos revelan una coherencia en las métricas de evaluación, indicando que el rendimiento del modelo es consistente en la clasificación de comportamiento acosador en las redes sociales.

*Tabla 4 Resultados obtenidos al clasificar los grafos*

| **Métrica** | **Resultado** |
|---|---|
| Precisión | 0.61 |
| *Exhaustividad* | 1.0 |
| *Exactitud* | 0.62 |
| $F_1$ | 0.76 |

Los resultados obtenidos nos han permitido llegar a la conclusión que clasificar aplicando un modelo de redes de grafos convolucionales logra resultados significativos. Ya

que alcanzamos una precisión del 61%, junto con una exhaustividad del 100%, exactitud del 62% y un $F_1$ de 0.76. Estos resultados son significativamente superiores a la precisión del 56.2% reportada por Lemus et. al., 2022 en un contexto similar.

## 6 Conclusiones

En este trabajo se propone la detección del acoso o ciberbullying en textos provenientes de redes sociales. El corpus se divide en tres clases: positivo (acoso presente), negativo (sin acoso) y neutro (temas generales). Tras el preprocesamiento del texto, que incluye la conversión a minúsculas, la eliminación de signos de puntuación, elementos HTML y menciones @usuario, se generaron grafos no dirigidos para cada clase, resultando en un total de tres grafos en formato CSV. Estos grafos fueron utilizados como entrada para una red neuronal convolucional diseñada para grafos.

Los resultados de la clasificación fueron evaluados mediante métricas de precisión, recall, exactitud (accuracy) y $F_1$, calculadas con un promedio ponderado para reflejar la distribución de las clases en el conjunto de datos. Las métricas obtenidas muestran una evaluación consistente, con resultados que se mantienen dentro de un rango relativamente estrecho en múltiples ejecuciones, lo que sugiere que el modelo mantiene un rendimiento uniforme y confiable en la clasificación de textos como acoso o no acoso. Sin embargo, aunque los resultados son consistentes, se identifican áreas para mejorar el modelo. Se propone explorar técnicas adicionales como análisis de sentimientos o corrección ortográfica para potencialmente aumentar la precisión de la detección. Además, se considera usar otras alternativas en las estructuras de grafos, técnicas de preprocesamiento y métodos de aprendizaje automático o profundo para enriquecer la capacidad del modelo y adaptarse a diferentes contextos y tipos de acoso en redes sociales.

## Referencias


Olweus, D., & Limber, S. P. (2010). Bullying in school: Evaluation and dissemination of the Olweus Bullying Prevention Program. *American Journal of Orthopsychiatry*, 80(1), 124-137. https://doi.org/10.1111/j.1939-0025.2010.01015.x

Leymann, H. (1990). Mobbing and psychological terror at workplaces. *Violence and victims*, 5(2), 119-126.

Fitzgerald, L. F., Gelfand, M. J., & Drasgow, F. (1995). Measuring sexual harassment: Theoretical and psychometric advances. *Basic and applied social psychology*, 17(4), 425-445.

Hinduja, S., & Patchin, J. W. (2017). Cultivating youth resilience to prevent bullying and cyberbullying victimization. *Child abuse & neglect*, 73, 51-62.

Recalde Monar, J. A. (2021). El ciberacoso por redes sociales en el Ecuador (Bachelor's thesis). Universidad Politécnica Salesiana sede Guayaquil.

Llampa, Á. F., Farfán, J., & Rodríguez, M. E. (2020). Análisis de sentimiento aplicado a la detección de ciberacoso en YouTube como red social. *In VI Simposio Argentino de Ciencia de Datos y Grandes* datos (AGRANDA 2020) - *JAIIO 49* (Modalidad virtual, pp. 77-88), Argentina.

Mamani Apaza, M. 2020, Modelo basado en inteligencia artificial para la detección de ciberacoso (Doctoral dissertation). Universidad Mayor de San Andrés Ciudad de La Paz Estado Plurinacional de Bolivia

Al-Garadi, M. A., Hussain, M. R., Khan, N., Murtaza, G., Nweke, H. F., Ali, I., ... & Gani, A. (2019). Predicting cyberbullying on social media in the big data era using machine learning algorithms: review of literature and open challenges. *IEEE Access*, 7, 70701-70718.

Hani, J., Nashaat, M., Ahmed, M., Emad, Z., Amer, E., & Mohammed, A. (2019). Social media cyberbullying detection using machine learning. *International Journal of Advanced Computer Science and Applications*, 10(5), 703-707.

Ali, A., & Syed, A. M. (2020). Cyberbullying detection using machine learning. *Pakistan Journal of Engineering and Technology*, 3(2), 45-50.

Lemus, J. A. L., & Morales, S. C. (2022). Clasificación de polaridad: Ciberacoso en Twitter. *Universidad de Guanajuato*, 1-8.

Kaggle. (2020). Acoso cibernético. Recuperado en diciembre de 2023 de https://www.kaggle.com/datasets/lexandermorales/acosocibernético?select=segura_dataset.csv

Raj, C., Agarwal, A., Bharathy, G., Narayan, B., & Prasad, M. (2021). Cyberbullying detection: Hybrid models based on machine learning and natural language processing techniques. *Electronics, 10*(22), 2810.

Mercado, R. N. M., Chuctaya, H. F. C., & Gutierrez, E. G. C. (2018). Automatic cyberbullying detection in spanish-language social networks using sentiment analysis techniques. *International Journal of Advanced Computer Science and Applications*, 9(7). 228-235

Afrifa, S., & Varadarajan, V. (2022). Cyberbullying detection on twitter using natural language processing and machine learning techniques. *International Journal of Innovative Technology and Interdisciplinary Sciences*, 5(4), 1069-1080.

# Capítulo 6

## Identificación automática de depresión y conductas suicidas en textos de redes sociales en español usando aprendizaje automático

Gabriela Martínez-Cuazitl[1], José A. Reyes-Ortiz[1], Josué Padilla-Cuevas[1], Gabriela A. García-Robledo[1]

[1]Universidad Autónoma Metropolitana, Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería

{al2232800334,jaro,jpc,gagr}@azc.uam.mx

**Resumen.** La depresión y las conductas suicidas se han convertido, en los últimos años, en los trastornos mentales que afectan a la sociedad. La detección automática de estos trastornos a partir de textos de redes sociales se ha vuelto un gran reto para el área de investigación en procesamiento de lenguaje natural, debido a la complejidad de los textos en español. El uso de algoritmos de aprendizaje automático en identificación de trastornos mentales ha sido de gran importancia en los últimos años, gracias a la tarea de clasificación que se puede llevar a cabo con estos algoritmos. En este trabajo se utilizan 4 algoritmos de clasificación de aprendizaje automático: Naive Bayes, Árboles de decisión, Máquinas de Soporte Vectorial y Perceptrón para la clasificación de textos con conducta suicida y no suicida, así como textos depresivos y no depresivos. Se utiliza la métrica de exactitud para medir el rendimiento de la clasificación obtenida del entrenamiento de los algoritmos.

**Palabras Clave:** Salud mental, textos de redes sociales, Procesamiento de Lenguaje Natural, extracción de características.

## 1 Introducción

Uno de los elementos importantes en la vida de los seres humanos es la salud mental, muchos factores pueden llegar a afectarla o alterarla. Los problemas de salud mental pueden llegar a ser desde leves a muy graves; uno de los trastornos más comunes; y que puede afectar en gran medida la vida de una persona es la depresión. La depresión es tipo de trastorno que no representa debilidad personal ni defecto de carácter, los dos síntomas más comunes de padecer depresión son sentirse triste o desesperanzado casi todos los días por lo menos 2 semanas, perder el interés por la mayoría de las actividades cotidianas que solía hacer la persona o no disfrutarlas y sentirse así casi todos los días durante por lo menos 2 semanas (Cigna Healthcare, 2024). Además, es muy importante señalar que un síntoma

grave de la depresión es pensar en la muerte o en el suicidio, por ello resulta imparte su identificación.

Si bien existen tratamientos, no siempre las personas logran salir a delante de la depresión y no siempre las personas que la padecen buscan ayuda o tiene acceso a ella.

De acuerdo con la Organización Mundial de la Salud, el suicidio es la cuarta causa de muerte entre el grupo de 15-29 años en todo el mundo (Organización Mundial de la Salud, 2023). Además, en el año 2022 se estimó que 2 de cada 10 mexicanos padecen depresión o alguno de sus síntomas (Salud mental, 2023). La depresión puede presentarse en cualquier momento a cualquier edad, sin razón aparente o ser causa de problemas graves como lo fue la pandemia COVID-19, por ejemplo, se sabe que, entre una de las tantas consecuencias de esta, se incrementaron los casos de depresión en México, afectado así la vida de las personas.

Es importante indicar que uno de los grupos con mayor riesgo son personas jóvenes, que muchas veces, si bien no buscan ayuda, es muy frecuente que utilizan redes sociales en su vida cotidiana y como un desahogo de sus problemas.

Por todo lo anterior, es necesario que existan herramientas computacionales de apoyo a expertos en salud mental para detectar rasgos de depresión y conductas de suicidio, por lo tanto, en este artículo se propone hacer una comparación de algoritmos de aprendizaje automático para la identificación automática de la depresión y conductas de suicidio a partir de textos provenientes de las redes sociales, utilizando características a nivel de pesado de palabras.

El resto del artículo está organizado como sigue. En la Sección 2 se presenta la revisión de los trabajos relacionados. La Sección 3 expone la metodología utilizada. En la sección 4 el procesamiento dado a los textos de los conjuntos utilizados. La sección 5 se aborda la lematización aplicada a los textos. En la sección 6 se explica la extracción de características con TF. En la sección 7 se mencionan los algoritmos de aprendizaje automáticos utilizados. En la sección 8 se encuentran los conjuntos y la segmentación realizada. La sección 9 contiene las configuraciones experimentales realizadas. En la sección 10 se presentan los resultados obtenidos. En la sección 11 se exponen las conclusiones y los trabajos a futuro.

## 2 Trabajos relacionados

Detectar la depresión en textos es de suma importancia y apoyo para este tipo de trastorno mental. Muchas investigaciones se han enfocado en clasificar textos en depresivos y no depresivos (Lan et al., 2024), (Zogan et al., 2022), (Nuñez-Prado et al., 2023), (Poświata et al., 2022), (Pool-Cen et al., 2023) en los textos depresivos también han identificado clasificaciones más detalladas, como un etiquetado de textos depresivos en Trastorno Depresivo Mayor y Trastorno Depresivo Persistente (Noguez et al., 2023), asimismo han encontrado otras clasificaciones, identificando el texto en un nivel de depresión en leve, moderada y grave (Rizwan et al., 2022). Se sabe que la depresión es factor clave para llegar

al suicidio por lo tanto ha sido importante identificar textos en depresivos y suicidio (Ghosal et al., 2023), entre otros trastornos de salud mental (Mármol-Romero et al., 2023).

Para llevar a cabo las distintas clasificaciones han utilizado métodos de aprendizaje automático supervisado como lo es XGBoost (Lan et al., 2024), (Ghosal et al., 2023), uno de los algoritmos más elementales de *machine learning* es k-vecinos más cercanos, que es un clasificador de aprendizaje supervisado, pero no paramétrico, mismo que han implementado (Nuñez-Prado et al., 2023). En algunos otros trabajos han optado por utilizar redes neuronales, como perceptrón multicapa, la cual es una red neuronal artificial que se utiliza para aprendizaje supervisado (Zogan et al., 2022), (Nuñez-Prado et al., 2023).

Igualmente se han basado en modelos de lenguaje como BERT, RoBERTa, XLNet (Poświata et al., 2022), RoBERTa Base, RoBERTa Large, DeBERTA (Mármol-Romero et al., 2023), modelo BETO (Pool-Cen et al., 2023), DistilBert, entre otros modelos de lenguaje (Rizwan et al., 2022). El ajuste fino en los modelos es muy importante para acoplar mucho mejor a un tema específico (Poświata et al., 2022).

Para la evaluación de resultados las métricas más usadas han sido *Accuracy* , *F1 score Precision* y *Recall* ya que son métricas para poder evaluar la calidad de los clasificadores, en múltiples investigaciones se utilizan las matrices de confusión para visualizar de mejor manera los resultados. La métrica que se destaca con mayor frecuencia en los resultados de los trabajos analizados es *F1-score* (Ghosal et al., 2023), (Zogan et al., 2022), (Rizwan et al., 2022), (Poświata et al., 2022), (Tlatelpa, 2020), (Pool-Cen et al., 2023).

Con la revisión del estado del arte, se hace evidente la necesidad de contar con herramientas computacionales para la identificación de depresión y conductas suicidas a partir de textos en español, ya que la mayoría de los enfoques existentes se centran en inglés y otros idiomas.

## 3 Metodología.

En esta sección se explica la metodología que se lleva a cabo para la identificación automática de depresión y conducta suicida en textos de redes sociales en español usando aprendizaje automático.

### 3.1 Recolección de conjuntos de datos

Para la búsqueda de conjuntos de datos se consultaron distintas plataformas, entre ellas Huggin Face (https://huggingface.com/) y Dataset Search de Google (https://datasetsearch.research.google.com/), en este último, algunos de los registros encontrados dirigían a la plataforma web de Kaggle (https://www.kaggle.com/).

A continuación, se describen los conjuntos de datos encontrados y algunas de sus características principales.

### 3.1.1 Comentarios depresivos: suicidio y no suicidio

Este conjunto de textos está compuesto por datos recabados de la red social Reddit y de la base de datos Suicide and Depression Detection de Nikhileswar Komati, es una colección de publicaciones de la comunidad o subreddit r/SuicideWatch. Para colectar las publicaciones se utilizó Pushshift API. Son datos obtenidos del 16 diciembre 2016 del 2008 al 2 enero del 2021. Los datos se encuentran en idioma español (Vásquez.D et al., s.f).

Este conjunto cuenta con 124,692 textos de la categoría suicidio y 124,229 de la categoría de no suicidio, con 248,921 textos en total.

### 3.1.2 Tweets en español que sugieren depresión

Pasando al siguiente conjunto de datos, se ha creado y utilizado para el artículo *Detecting signs of depression in tweets in Spanish: behavioral and linguistic analysis* (Leis et al., 2019). Cuenta con 1000 textos en idioma español, todos ellos de la clase depresivos (Leis et al., s.f).

### 3.1.3 Conjunto de datos de Twitter de depresión

En cuanto al conjunto de datos de Twitter de depresión está clasificado en 2 clases: depresivo y no depresivo con 3,082 textos y 4,687 textos respectivamente en cada clase (Cho, s.f).

Los textos se encuentran en idioma inglés por lo que se llevó a cabo una traducción al idioma español mediante la biblioteca *GoogleTranslator* que tiene una precisión de 94 % para traducciones en idioma inglés a español (Lokalise, 2024).

### 3.1.4 Depresión: conjunto de datos de Reddit (limpio)

Respecto al conjunto de textos Depresión: conjunto de datos de Reddit (limpio) han sido recabados de *subreddits*, contiene 3,831 textos depresivos y 3,900 no depresivos (InFamousCoder, s.f).

Los textos se encuentran en idioma inglés por lo que se llevó a cabo una traducción al idioma español mediante la biblioteca *GoogleTranslator* que tiene una precisión de 94 % para traducciones en idioma inglés a español (lokalise, 2024).

En la Tabla 1 se presentan los datos generales de cada uno de os conjuntos.

Tabla 1. Información general de los conjuntos de datos.

| Cita | Autores | Total de textos | No. De clases | Etiquetas de las clases |
|---|---|---|---|---|
| (Vásquez.D et al., s.f) | Danny Vásquez<br>César Salazar<br>Alexis Cañar<br>Yannela Castro<br>Daniel Patiño | 248,921 | 2 | Suicidio<br>No suicidio |
| (Leis et al., s.f) | Leis, A., Ronzano, F., Mayer, M. A., Furlong, L. I. y Sanz, F. | 1000 | 1 | Depresivo |
| (Cho, s.f) | Hyun Ki Cho | 7,769 | 2 | Depresivo<br>No depresivo |
| (InFamousCoder, s.f) | InFamousCoder | 7,731 | 2 | Depresivo<br>No depresivo |

Para el conjunto de (Vásquez.D et al., s.f) que tiene 248,921 textos, se utiliza únicamente 7,000 textos de cada clase: 7,000 de suicidio y 7,000 de no suicidio para iniciar con la experimentación, esto genera un conjunto de datos al cual se le denomina conjunto de textos de conductas suicidas con 14,000 textos en total.

En el caso de la depresión, se realizó una mezcla de los conjuntos (Leis et al., s.f), (Cho, s.f), (InFamousCoder, s.f) para tener un balance en las clases y trabajar con una mayor cantidad de datos. Esto origina un nuevo conjunto llamado textos depresivos quedando con un total de 16,500 textos, el cual está dividido en 7,913 de clase depresivo y 8,587 no depresivos.

Se obtiene el promedio de palabras por texto en cada uno de los conjuntos. Para el conjunto de textos de conductas suicidas sin limpieza es de 130.02 y con limpieza 64.76. Así mismo, para el conjunto de textos depresivos sin limpieza es de 32.57 y con limpieza 16.35.

## 4 Procesamiento de textos

En esta sección se explican las técnicas de procesamiento de lenguaje natural para la limpieza aplicada a los conjuntos de textos anteriormente mencionados.

Se implementaron métodos en Python para llevar a cabo esta limpieza y se utiliza la biblioteca *nltk* de *Python*.

En la Figura 1 se muestra un texto de los conjuntos para explicar la limpieza aplicada.

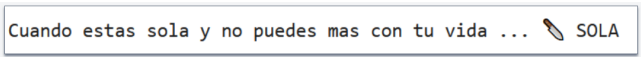

Figura 1. Texto de ejemplo original.

## 4.1 Tokenización

La tokenización consiste en aplicar algún tipo de división de palabras, en este caso se divide por espacios incluyendo la puntuación, esta tarea se realiza con ayuda de *wordpunct_tokenize* de *Python*.

En la Figura 2 se muestra el texto de ejemplo tokenizado.

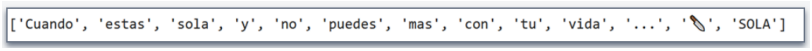

Figura 2. Ejemplo de texto tokenizado.

## 4.2 Normalización

La normalización consiste en transformar todos los textos de depresión y conducta suicida a minúsculas con la finalidad de mantener a un mismo nivel los textos.

En la Figura 3 se muestra el texto normalizado.

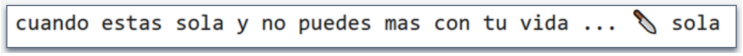

Figura 3. Ejemplo de texto normalizado.

## 4.3 Cambio de emoji a palabras

El cambio de emoji a palabras consiste en cambiar los emojis por el significado. Para llevar a cabo esta etapa se realiza un diccionario de emoji y su significado.

En la Figura 4 se muestra el texto cambiando el emoji por su significado.

Figura 4. Texto con el emoji modificado a palabra.

## 4.4 Eliminación de puntuación

La eliminación de puntuación sirve para quitar aquellos signos de puntuación que no aportan significado al texto.

Se utilizan también, expresiones regulares para eliminar signos consecutivamente repetidos por ejemplo (...).

En la Figura 5 se muestra el texto eliminando la puntuación existente.

```
cuando estas sola y no puedes mas con tu vida  cuchillo sola
```

Figura 5. Texto sin signos de puntuación.

## 4.5 Eliminación de palabras vacías

Se realiza la eliminación de palabras vacías con ayuda de *stopwords* de *Python.* Se obtienen primeramente las palabras vacías en español.

En la Figura 6 se muestra el texto una vez eliminadas las palabras vacías.

```
['sola', 'puedes', 'mas', 'vida', 'cuchillo', 'sola']
```

Figura 6. Texto sin palabras vacías.

# 5 Lematización

Con la limpieza anterior se obtiene una reducción de palabras, pero se experimenta con lematización ya que este proceso permite llevar las palabras a su forma raíz. Para ello se utiliza *spacy* y *es_core_news_sm* de *Python*.

Continuando con el mismo ejemplo del texto. Se muestra en la Figura 7 el texto una vez aplicado la lematización.

```
'solo puede mas vida cuchillo solo'
```

Figura 7. Texto lematizado.

# 6 Extracción de características

Para obtener la extracción de características se utiliza *TF* (*Term Frequency*) que mide la frecuencia con la que se repite un término, en este caso, se aplica a cada uno de los documentos (textos) (J.E Quiroz, 2022).

Para realizar el cálculo de *TF* se hace uso de *sklearn.feature_extraction.text* y *CountVectorizer* de *Python*.

# 7 Algoritmos de aprendizaje automático tradicionales

Se experimenta con 4 algoritmos de aprendizaje automático tradicionales para la clasificación de textos.

A continuación, se abordan cada uno de los algoritmos utilizados.

## 7.1 Naive Bayes

Es un algoritmo de aprendizaje supervisado utilizado para clasificación. Su funcionamiento se basa en el teorema de Bayes que lo que hace es calcular la probabilidad de un evento cuando se tiene información previa sobre otro evento relacionado.

Relaciona la probabilidad de un evento A dado un evento B con la probabilidad de B dado A (F. Cardellino, 2021).

## 7.2 Árboles de decisión

Es un algoritmo de aprendizaje supervisado no paramétrico que es utilizado para tareas de clasificación y regresión. Se tiene una estructura de árbol jerárquica, compuesta por un nodo raíz, ramas, nodos internos (nodos de decisión) y nodos hoja. Implementan la técnica de divide y vencerás. Realiza un proceso recursivo de arriba hacia abajo del árbol hasta que todos o bien la mayoría de los registros hayan sido clasificados mediante las etiquetas específicas (IBM, s.f).

## 7.3 Máquinas de soporte vectorial

*Support vector machines (SVM)* son métodos desarrollados por *Vladimir Vapnik* a mediados del año 1960, en los años noventa fue introducido a la informática.

Es un algoritmo de aprendizaje supervisado que se utiliza para clasificación y regresión, entre otros. La finalidad de este algoritmo es encontrar un hiperplano que divida de la manera más clara dos clases diferentes dados los puntos de datos (R.F Casal, 2021).

Para la implementación de este algoritmo se utilizó *sklearn* y *svm* de *Python*.

## 7.4 Perceptrón

El perceptrón fue inventado por Frank Resonblatt en enero de 1957, en el *Cornell Aeronautical Laboratory* en *Buffalo*, Nueva York. Perceptrón es una red neuronal simple;

son algoritmos diseñados en base a la estructura neuronal del cerebro humano, son de suma importancia debido a la capacidad de entender y aprender a partir de datos. El perceptrón es muy utilizado para clasificaciones binarias es decir donde existen 2 tipos de clases (L. González, 2021).

Para la implementación de este algoritmo se utilizó *sklearn.linear_model* y *Perceptron* de Python.

A continuación, se muestra en la Tabla 2 los parámetros más importantes de los algoritmos utilizados en este trabajo.

Tabla 2. Parámetros de los algoritmos de aprendizaje automático utilizados.

| Algoritmo de clasificación | Parámetros |
|---|---|
| Naive Bayes | **priors:** n_classes, default=None<br>**var_smoothing:** float, default=1e-9 |
| Árboles de decisión | **penalty:** {'l2','l1','elasticnet'}, default=None<br>**alpha:** float, default=0.0001<br>**l1_ratio**float, default=0.15<br>**max_iter:** int, default=1000 |
| Máquinas de soporte vectorial | **C:** float, default=1.0<br>**Kernel:** {'linear', 'poly', 'rbf", 'sigmoid', 'precomputed'}<br>or callable, default='rbf"<br>**degree:** int, default=3<br>**gamma:** {'scale', 'auto'} or float, default='scale' |
| Perceptrón | **alpha:**float, default=0.0001<br>**l1_ratio:** float, default=0.15<br>**fit_intercept:** bool, default=True<br>**max_iter:** int, default=1000 |

# 8 Experimentación y resultados

En esta sección se presentan los conjuntos de textos en español utilizados, así como la segmentación de estos en términos de la cantidad de textos utilizados para la etapa de entrenamiento y pruebas. También, se exponen las configuraciones experimentales llevadas a cabo que consisten en combinar los diversos algoritmos de aprendizaje automático con la tarea de procesamiento de lenguaje natural denominada "lematización" para ambos conjuntos: depresión y conductas suicidas. Finalmente, se presentan los resultados de estos experimentos.

## 8.1 Segmentación de los conjuntos de datos.

Es necesario realizar una segmentación de "Conjunto de textos de conductas suicidas" y "Conjunto de textos depresivos", para el entrenamiento de los algoritmos utilizados en este

trabajo, se segmentan los datos en 2 elementos: entrenamiento y prueba. Esta actividad se realiza con *sklearn* y *train_test_split* de Python. Para el "Conjunto de textos de conductas suicidas" para entrenamiento se utilizan 11,200 textos (80%) y para prueba 2,800 textos (20%). En cuanto a el "Conjunto de textos depresivos" para entrenamiento se utilizan 13,200 textos (80%) y para prueba 3,300 textos (20%).

### 8.2 Configuraciones experimentales

Se realizan un total de 16 experimentaciones dadas a partir de conjuntos de textos: "Conjunto de textos de conductas suicidas" y "Conjunto de textos depresivos" iniciando así una primera etapa de tomar únicamente los textos con la limpieza dada por las técnicas de procesamiento de lenguaje natural aplicadas y posteriormente una segunda etapa que consiste en aplicar lematización a los textos con limpieza. Implementando en ambas etapas los 4 algoritmos de clasificación: Naive Bayes, Árboles de decisión, Máquinas de Soporte Vectorial (SVM) y Perceptrón.

Se utiliza la métrica de exactitud (*accuracy*) para evaluar la clasificación de los textos en los 4 algoritmos, la métrica de exactitud arroja el número de elementos clasificados correctamente en contraste con el número total de textos (Fayrix, 2019), a continuación, se presentan los resultados, para cada conjunto: "Conjunto de textos de conductas suicidas" y "Conjunto de textos depresivos".

### 8.3 Resultados para el conjunto de textos de conductas suicidas

En la Tabla 3 se muestran los resultados obtenidos para el conjunto de textos de conductas suicidas para cada algoritmo experimentado.

Tabla 3. Resultados obtenidos para el conjunto de textos de conductas suicidas

| Algoritmo de clasificación | Limpieza | Limpieza+Lematización |
|---|---|---|
| Naive Bayes | 73.11% | 73.07% |
| Árboles de decisión | 81.07% | 82.75% |
| Máquinas de Soporte Vectorial | 82.73% | **88.82%** |
| Perceptrón | **88.79%** | 88.39% |

Los resultados de suicidio, iniciando únicamente con limpieza el mejor es logrado con Perceptrón con exactitud de 88.79% la simplicidad y eficiencia permite trabajar a perceptrón con grandes cantidades de datos, con respecto a los resultados de suicidio con limpieza y lematizados el mejor resultado se refleja con Máquinas de Soporte Vectorial con exactitud de 88.82%, la limpieza y lematización aplicada a los textos reduce la longitud de estos y una característica de Máquinas de Soporte Vectorial es que este algoritmo puede clasificar de mejor manera con textos más cortos. Como se mencionó anteriormente en la

sección 3, el promedio de palabras con limpieza de los textos del conjunto de suicidio es de 64.76 y del conjunto de textos depresivos es de 16.35.

Para una mejor visualización de los resultados se presenta la Figura 8 con los resultados de este conjunto de textos de suicidio.

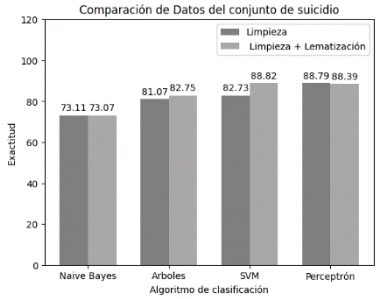


*Figura 8. Resultados del conjunto de textos de suicidio.*

### 8.4 Resultados para el conjunto de textos depresivos

En la Tabla 4 se muestras los resultados obtenidos para el conjunto de textos depresivos para cada algoritmo experimentado.

Tabla 4. Resultados obtenidos para el conjunto de textos depresivos

| Algoritmo de clasificación | Limpieza | Limpieza+Lematización |
|---|---|---|
| Naive Bayes | 70.36% | 67.79% |
| Árboles de decisión | 76.27% | 75.67% |
| Máquinas de Soporte Vectorial | **83.76%** | **82.91%** |
| Perceptrón | 80.00% | 78.48% |

Los resultados para la clasificación de textos depresivos mostrados en la tabla 6, tanto para los textos limpios como limpios y con lematización el mejor resultado se obtiene con el algoritmo de Máquinas de Soporte Vectorial y para limpieza con exactitud de 83.76% y para limpieza+lematización con exactitud de 82.91% esto se debe a que el algoritmo trabaja mejor con textos de menor longitud. Lo que comparado con los textos de suicidio la longitud es mayor. Como se mencionó anteriormente en la sección 3, el promedio de palabras del conjunto de suicidio es de 130.02 y del conjunto de textos depresivos es de 32.57.

Para una mejor visualización se presenta la Figura 9 con los resultados de este conjunto de textos depresivos.

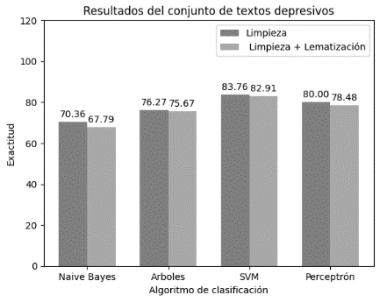


*Figura 9. Resultados del conjunto de textos de depresión.*

# 9 Conclusiones y trabajo a futuro

En este trabajo se presenta una comparación de 4 algoritmos de clasificación: Naive Bayes, Árboles de decisión, Máquinas de Soporte Vectorial (*SVM*) y Perceptrón, sobre los conjuntos: "Conjunto de textos de conductas suicidas" y "Conjunto de textos depresivos", con la extracción de características por medio de *TF* realizando previamente una limpieza con técnicas de procesamiento de lenguaje natural y para una segunda etapa de experimentos realizando lematización a los conjuntos de textos ya con la limpieza. Siendo así la métrica de exactitud, como el criterio de medición de resultados del entrenamiento de los clasificadores. Los mejores resultados para identificación de depresión y conductas suicidas en textos utilizando los conjuntos: "Conjunto de textos de conductas suicidas" y "Conjunto de textos depresivos", se logran con Máquinas de Soporte vectorial. Para el "Conjunto de textos de conductas suicidas" se obtuvo una exactitud de **88.82%** aplicando **limpieza y lematización** y para el "Conjunto de textos depresivos" se logro una exactitud de **83.76%** únicamente con **limpieza**. Con esto se puede demostrar que el algoritmo de Máquinas de soporte vectorial (*SVM*), es un algoritmo eficaz para la identificación de textos depresivos y conductas suicidas.

La salud mental es desafortunadamente un tema de salud que no se aborda tan frecuentemente y que no se le ha dado el seguimiento necesario como lo ha sido con otras enfermedades. Por lo que encontrar datos de depresión y conductas suicidas no es tarea sencilla. Por otro lado, las personas que logran tener un tratamiento, suelen ser tratamientos largos que interrumpen su vida cotidiana. Tener herramientas de apoyo aportará que se puedan atender a más personas o una detección más rápida, de manera que continuar con esta investigación puede llegar a ser muy favorable en auxiliar a expertos de la salud.

Como primer acercamiento a los experimentos para la identificación de depresión y conductas suicidas en textos en español es buen hallazgo para continuar con la investigación, de modo que en un futuro se tenga una herramienta de apoyo para expertos en la salud que basados en su criterio y con ayuda de la herramienta computacional puedan

tomar la decisión acertada y esto tenga un impacto positivo en salud mental y en la detección de depresión así como la prevención de consecuencias graves como el suicidio.

Con respecto a los resultados obtenidos para los 4 clasificadores se puede observar que son datos muy cercanos, tanto en limpieza como en la experimentación con limpieza y lematización, de modo que como trabajos a futuro sería importante experimentar con otras técnicas como *stemming*, etiquetado POS, experimentar con características lingüísticas podría generar un mayor aporte, además, de aplicar ahora *TF-IDF*, también, agregar mayor cantidad de datos y evaluar el rendimiento de los clasificadores con otras métricas como exhaustividad(*recall*), precisión(*precision*), puntuación F1(*score-F1*) y así encontrar características que mejoren el aprendizaje de los modelos. Del mismo modo implementar grandes modelos de lenguaje y estudiar su comportamiento con los textos depresivos y de suicidio.

## Referencias


Salud mental. (2023) *Dos de cada 10 mexicanos tienen depresión*. Recuperado de https://consultorsalud.com/salud-mental-2-cada-10-mexicanos-tienen-depresion/

Cigna Healthcare. (2024). *Depresión*. Recuperado de: https://www.cigna.com/es-us/knowledge-center/hw/temas-de-salud/depresin-hw30709

Cho, H. K. (2021). *Twitter depression dataset [data set]*. Recuperado de https://www.kaggle.com/datasets/hyunkic/twitter-depression-dataset

F. Cardellino. (2021). *Cómo funcionan los clasificadores naive bayes: con ejemplos de código de python*. Recuperado de https://www.freecodecamp.org/espanol/news/como-funcionan-los-clasificadores-naive-bayes-con-ejemplos-de-codigo-de-python/

Fayrix. (2019). *Selección de métricas para aprendizaje automático*. Recuperada de https://fayrix.com/machine-learning-metrics_es

Ghosal, S., Jain, A. (2023). "Depression and suicide risk detection on social media using fasttextembedding and xgboost classifier", *Procedia Computer Science*, vol. 218, pp. 1631–1639.

IBM. (2022). *¿qué es un árbol de decisión?*. Recuperado de https://www.ibm.com/mx-es/topics/decision-trees

InFamousCoder. (2021). *Depression: Reddit dataset (cleaned) [data set]*. Recuperado de https://www.kaggle.com/datasets/infamouscoder/depression-reddit-cleaned

J.E Quiroz. (2022). *Métricas del procesamiento del lenguaje natural (nlp): Tf-idf*. Recuperado de https://iniat.ibero.mx/blog/metricas-del-procesamiento-del-lenguaje-natural-nlp-tf-idf/

L. González. (2021). *¿qué es el perceptrón? perceptrón simple y multicapa – aprende_ia*. Recuperado de https://aprendeia.com/que-es-el-perceptron-simple-y-multicapa/

Lan, X. (2024). *Depression detection on social media with large language models*. Recuperado de https://arxiv.org/abs/2403.10750.

Leis, A., Ronzano, F., Mayer, M., Furlong, L., Sanz, F. (2019). "Detecting signs of depression in tweets in spanish: behavioral and linguistic analysis", *Journal of medical Internet research*, vol. 21, pp. 312-327.

Leis, A. (2021). *Spanish tweets suggesting depression [data set]*. Recuperado de https://www.kaggle.com/datasets/francescoronzano/spanish-tweets-suggesting-depression

Lokalise. (2024). *How accurate is google translate? google translate vs deepl*. Recuperado de: https://lokalise.com/blog/google-translate-accuracy/

Mármol-Romero, A. M., Moreno-Muñoz, A., Plaza-del-Arco, F.M., Molina-González, M.D., Martín-Valdivia, M.T., Ureña-López, L.A., Montejo-Ráez, A. (2023). "Overview of mentalriskes at iberlef 2023: Early detection of mental disorders risk in spanish", *Procesamiento del Lenguaje Natural*, vol. 71, pp. 329–350.

Noguez, L. R. G. (2023). *Identificación de rasgos asociados a la depresión mediante técnicas de aprendizaje automático y procesamiento de lenguaje natural*. Repositorio Institucional DGBSDI-UAQ.

Núnez-Prado, C.-J., Talavera, C., Chanona-Hernández, L., Sidorov, G. (2023). "Aplicación de algoritmos de aprendizaje automático sobre un corpus depresivo digital", *Research in Computing Science*, vol. 152, pp. 89–98.

Organización Mundial de la Salud. (2023). *Suicidio*. Recuperado de: https://www.who.int/es/news-room/fact-sheets/detail/depression

Pool-Cen, J., Carlos-Martínez, H., Hernández-Chan, G., Sánchez-Siordia, O. (2023). "Detection of depression-related tweets in mexico using crosslingual schemes and knowledge distillation", *Healthcare*, vol. 11, pp. 1-18.

Poświata, R. (2022). Opi@ lt-edi-acl2022: Detecting signs of depression from social media text using roberta pre-trained language models. *Proceedings of the Second Workshop on Language Technology for Equality, Diversity and Inclusion*, pp. 276–282.

Casal, R.F. (2021). *Capítulo 4 máquinas de soporte vectorial*. Recuperado de: https://rubenfcasal.github.io/aprendizaje estadistico/svm.html

Rizwan, M., Mushtaq, M.F., Akram, U., Mehmood, A., Ashraf, I., Sahelices, B. (2022). "Depression classification from tweets using small deep transfer learning language models", *IEEE Access*, vol. 10, pp. 129176–129189.

Tlatelpa, J. d. J. T. (2020). *Detección de depresión en redes sociales considerando información del perfil de los usuarios.* Repositorio Institucional INAOE.

Vásquez. D. (2020). *comentarios depresivos [data set]*. Recuperado de: https://huggingface.co/datasetshackathon-pln-es/comentariosdepresivos

Zogan, H., Razzak, I., Wang, X., Jameel, S., Xu, G. (2022). "Explainable depression detection with multi-aspect features using a hybrid deep learning model on social media", *World Wide Web*, vol. 25, pp. 281–304

# Capítulo 7

## Un análisis comparativo de enfoques de Aprendizaje Automático para identificar relaciones y entidades con conjuntos desbalanceados de textos médicos en español

Nidia K Serafin-Rojas[1], José A. Reyes-Ortiz[1], Maricela Bravo[1], Josué Padilla-Cuevas[1]

[1] Universidad Autónoma Metropolitana, Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería

{al2221800042,jaro,mcbc,jpc}@azc.uam.mx

**Resumen.** El procesamiento automático de información médica es de suma importancia para generar sistemas o herramientas computacionales que consideren información de relaciones y entidades como medicamentos y enfermedades en la toma de decisiones. Por ello, en este artículo se profundiza en técnicas con algoritmos de aprendizaje automático utilizadas para extraer información de textos médicos en español. El estudio se enfoca en la evaluación de algoritmos de aprendizaje automático como Máquinas de Soporte Vectorial (*SVM*), Árboles de Decisión, K-Vecinos más Cercanos (*k-NN*) y Perceptrón. Se analiza cómo estos algoritmos pueden identificar entidades y relaciones en conjuntos de datos desbalanceados, una situación común en textos médicos reales. Los resultados destacan que las Máquinas de Soporte Vectorial superan a los demás algoritmos en la identificación de entidades y relaciones. Además, se concluye que la combinación adecuada de características morfológicas y semánticas mejora significativamente los resultados, alcanzando una medida $F_1$ del 78.72% para la identificación de entidades y del 35.5% para la extracción de relaciones.

**Palabras Clave:** Procesamiento del Lenguaje Natural, Extracción de Entidades y Relaciones, Aprendizaje Automático.

## 1 Introducción

El campo de las Ciencias de la Salud genera una gran cantidad de información como documentos clínicos no estructurados, informes médicos, diagnósticos y ensayos clínicos.

El Procesamiento del Lenguaje Natural (PLN) ha demostrado ser una herramienta con gran valor para extraer entidades y relaciones significativas de esta extensa cantidad de datos. Al automatizar el proceso de extracción de información, el PLN mejora significativamente la eficiencia de la interpretación de datos médicos. Esto, a su vez,

conduce a la innovación en áreas críticas como la investigación clínica, el desarrollo de tratamientos y el cuidado de los pacientes. Por lo tanto, es esencial aplicar técnicas de procesamiento del lenguaje natural para desbloquear el potencial de la información médica no estructurada y avanzar en la comprensión y tratamiento de enfermedades.

Este estudio tiene como objetivo comparar los enfoques tradicionales de aprendizaje automático (Aggarwal, 2018) para extraer información de textos médicos. El proceso implica la recopilación de un conjunto de textos en español con contexto médico, la aplicación de algoritmos tradicionales de aprendizaje automático adaptados a los textos, y la obtención de los resultados. Los resultados obtenidos se comparan para determinar qué algoritmo produce los mejores resultados en términos de extracción de información. Los algoritmos utilizados en este estudio incluyen k-Vecinos más Cercanos, Máquinas de Soporte Vectorial, Árboles de Decisión y Perceptrón.

Este documento está estructurado de la siguiente manera. La Sección 2 proporciona una revisión del trabajo previo relacionado a la identificación de entidades y relaciones; la Sección 3 detalla el conjunto de textos médicos utilizados y la extracción del texto de interés; la Sección 4 presenta los enfoques tradicionales de aprendizaje automático empleados en este estudio, así como el procesamiento de textos asociado; la Sección 5 lleva a cabo un análisis comparativo entre los algoritmos de aprendizaje automático; finalmente, se presentan las conclusiones.

## 2 Estado del arte

La disponibilidad limitada de conjuntos de datos en español etiquetados por expertos presenta un gran desafío, como se destaca en este informe (Dellanzo et al, 2022), que aborda la escasez de trabajos con soluciones usando corpus en español. En respuesta, como parte de su trabajo construyó un corpus compuesto por 513 artículos anotados. El objetivo es realizar tareas de reconocimiento de entidades nombradas y extracción de relaciones en textos de informes de salud sobre brotes en América Latina. Se emplea una combinación de Redes Neuronales Recurrentes y Campos Aleatorios Condicionales, y sus resultados se evalúan por categoría. Para las entidades, el promedio de los resultados varía entre el 52% y el 68% este valor alcanzado por una coincidencia parcial, y el resultado promedio de las relaciones entre el 73% y el 92% en términos de medida $F_1$.

Los textos médicos en español presentan desafíos significativos debido a su naturaleza no estructurada, como se afirma en el trabajo (Mendoza-Urbano et al, 2023). Se desarrolló un módulo de procesamiento del lenguaje natural y se construyó un conjunto de datos utilizando 140 informes de patología oncológica. Los resultados fueron bastante prometedores en comparación con resultados analizados por humanos, empleando métricas como precisión, exactitud, exhaustividad y medida $F_1$.

Mientras que la mayoría de los estudios se han realizado en idiomas dominantes como el inglés, el trabajo (Kaur et al, 2021) destaca por abordar el procesamiento de textos sin estructura en idioma Punjabi. Los investigadores emplearon algoritmos como Modelos

Ocultos de Markov, Principio de Máxima Entropía y Campos Aleatorios Condicionales, logrando medida $F_1$ de 77.61%, 83.65% y 93.21%, respectivamente.

En un estudio reciente centrado en la extracción de información dentro del dominio médico, (Wei et al., 2020) comparó Máquinas de Soporte Vectorial con un enfoque de aprendizaje profundo BiLSTM-CRF. Este enfoque se probó además en (Fang et al., 2022), demostrando que la combinación de BiLSTM-CRF con representaciones de *Transformers* y Campos Aleatorios Condicionales (BERT-BiLSTM-CRF) resultó efectiva para el reconocimiento de entidades nombradas en textos chinos y la extracción de características POS.

Un trabajo realizado por (Al-Smadi et al, 2019), se centró en el análisis de sentimientos a través de diversas técnicas de extracción de información. Esto implicó la identificación de categorías de aspectos utilizando una Máquina de Vectores de Soporte para el entrenamiento, así como la extracción de la expresión objetivo de la opinión aplicando tareas de procesamiento del lenguaje natural y extrayendo características morfológicas, sintácticas y semánticas. El estudio aplicó algoritmos de aprendizaje automático como Máquinas de Vectores de Soporte, Naive Bayes, Árboles de Decisión y Vecino Más Cercano, utilizando reseñas de hoteles en árabe como fuente de datos.

El estudio de (Sanchez-Graillet et al, 2022) analiza la aplicación práctica de la extracción de conceptos clínicos utilizando resúmenes médicos de glaucoma y diabetes mellitus tipo 2 (T2DM). Utilizaron el modelo BERT-base y lograron una medida $F_1$ promedio del 76% para glaucoma y del 77% para diabetes, resaltando su utilidad final en la implementación de ontologías.

Por otro lado, los trabajos (Yang, 2021) y (Monteagudo-García et al, 2021) participaron en el desafío eHealth-KD 2021, empleando enfoques como BERT-CRF, BILSTM-CRF y BILSTM para la tarea A, que consiste en la identificación de entidades, con medidas $F_1$ de 17.3% y 60.7%, respectivamente. En cuanto a la extracción de relaciones, el equipo con el puntaje más alto fue IXA con una medida $F_1$ del 43%.

Este estudio aborda la identificación de entidades y relaciones mediante la comparación de cuatro algoritmos: utilizando aprendizaje automático con algoritmos k-NN, Máquinas de Soporte Vectorial, Árboles de Decisión y Perceptrón. La mejor precisión en la identificación de entidades fue del 78.75% con Máquinas de Soporte Vectorial, mientras que para la identificación de relaciones se obtuvo 35.5%. Es importante destacar que se utilizaron exclusivamente textos en español.

## 3 Conjunto de textos

Este estudio profundiza en el análisis de textos médicos, lo cual requiere la identificación de conjuntos de datos anotados por expertos. Para lograr esto, se utiliza el conjunto de textos proporcionado por el desafío eHealth-KD 2021 (Piad-Morffis et al, 2019), que incluye frases extraídas de diversas fuentes como MedlinePlus, Wikinews y el corpus CORD-19,

todas relacionadas con temas de salud. Esto asegura una amplia variedad de formatos y estructuras.

Dado nuestro enfoque en el idioma español, se modificó el conjunto de texto original eliminando las frases en inglés.

El corpus consta de dos archivos: uno contiene frases de texto con estructuras variadas como se muestra en la Figura 1, mientras que el otro contiene las anotaciones y se puede observar en la Figura 2. En el segundo archivo, las entidades y relaciones se identifican utilizando etiquetas T y R, respectivamente, seguidas de un número consecutivo que indica sus puntos de inicio y fin. Además, el archivo de anotación específica la clase a la que pertenece la entidad o relación identificada.

```
El sistema vascular es la red de vasos sanguíneos del cuerpo.
```

Figura 1. Ejemplo de archivo de texto

```
T1      Concept 3 10;11 19  sistema vascular
T2      Predicate 26 29     red
T3      Concept 33 38;39 49 vasos sanguíneos
T4      Concept 54 60 cuerpo
R0      arg Arg1:T2 Arg2:T3
R1      is-a Arg1:T1 Arg2:T2
```

Figura 2. Ejemplo de archivo de anotación

### 3.1 Entidades

Las entidades (Diccionario de la lengua española, 2024) son elementos valorados o significativos en un contexto específico. En el ámbito de la extracción de información, estas entidades son elementos distintivos identificados y extraídos de un texto o conjunto de datos. Estas entidades pueden incluir personas, organizaciones, lugares, términos médicos y otra información relevante para el dominio analizado. Este estudio se centra en la extracción de entidades generales, clasificadas en cuatro categorías: conceptos, acciones, predicados y referencias. La Tabla 1 a continuación muestra la frecuencia de apariciones por tipo de entidad en el contenido de las sentencias analizadas.

Tabla 1. Frecuencia por tipo de entidad

| Tipo de entidad | Frecuencia |
|---|---|
| Action | 512 |
| Concept | 1388 |
| Predicate | 190 |
| Reference | 55 |
| **Total** | **2145** |

### 3.2 Relaciones

El conjunto de textos utilizado tiene identificadas relaciones, las cuales se entienden como "conexiones o correspondencias entre elementos" (Diccionario de la lengua española, 2024). En este contexto, las relaciones se establecen entre entidades específicas. Este trabajo se centra en la identificación y análisis de doce tipos distintos de estas relaciones. Las cuales se muestran en la Tabla 2.

Tabla 2. Frecuencia por tipo de relaciones

| Tipo de relación | Frecuencia |
|---|---|
| target | 154 |
| in-context | 98 |
| subject | 86 |
| is-a | 75 |
| domain | 40 |
| causes | 38 |
| in-place | 38 |
| arg | 31 |
| in-time | 22 |
| has-property | 14 |
| part-of | 10 |
| entails | 9 |
| **Total** | **615** |

La Tabla 2 muestra un gran desbalance en los conjuntos de textos en español, esto se convierte en un reto para esta investigación desde que hay relaciones que contienen poca

cantidad de instancias lo que puede afectar el desempeño de los algoritmos. El objetivo es medir el comportamiento de los mismos con estos conjuntos de datos desbalanceados.

### 3.3 Extracción de textos de interés

Los textos deben prepararse para ser ingresados a los algoritmos de aprendizaje automático; por lo tanto, en textos no estructurados es necesaria la identificación del texto de interés con el fin de extraer características morfológicas y semánticas para este trabajo.

Para la obtención del texto de interés los archivos son procesados con ayuda de Python como lenguaje de programación, aprovechando su facilidad en el manejo de texto.

Para transformar los textos de interés en objetos manejables, se crea un dataframe para entidades y otro para relaciones en Python. Para las entidades el *dataframe* consta de tres columnas: la primera almacena la entidad identificada, la segunda contiene el texto relevante, que incluye una ventana de cinco palabras antes y cinco palabras después de la entidad y la tercera columna indica el tipo de entidad. Un ejemplo se muestra en la Tabla 3.

Tabla 3. Ejemplo de extracción de texto de interés para entidades

| Entidad | Contexto | Tipo Entidad |
|---|---|---|
| sistema vascular | El sistema vascular es la red de vasos | Concept |
| red | El sistema vascular es la red de vasos sanguíneos del cuerpo. | Predicate |
| vasos sanguíneos | vascular es la red de vasos sanguíneos del cuerpo. | Concept |

En cuanto a las relaciones, se almacena una columna con la relación identificada. La segunda columna contiene el texto que abarca desde la primera entidad identificada hasta la segunda entidad como se muestra en la Tabla 4.

Tabla 4. Ejemplo de extracción de texto de interés para relaciones

| Relación | Tipo Relación | Entidad 1 | Entidad 2 | Oración |
|---|---|---|---|---|
| R0 | arg | red | vasos sanguíneos | red de vasos sanguíneos |

| | | | | |
|---|---|---|---|---|
| R1 | is-a | sistema vascular | red | sistema vascular es la red |
| R2 | part-of | sistema vascular | cuerpo | sistema vascular es la red de vasos sanguíneos del cuerpo. |
| R14 | subject | elevado | columna | columna de cenizas se ha elevado |

### 3.4 Extracción de características

Los algoritmos de aprendizaje automático requieren de entradas en formato numérico, lo que se logra mediante la extracción de características. En este trabajo se experimenta con dos tipos principales: morfológicas y semánticas.

Las características morfológicas se refieren a la asignación de la categoría gramatical de la palabra en una oración, como verbo, sustantivo o adjetivo. Para esta asignación, el procesamiento de lenguaje natural utiliza la tarea de etiquetado Parte de la Oración (POS). En Python, esta tarea se implementa con la librería *SpaCy,* que incluye etiquetas predeterminadas como: ADJ (adjetivo), VERB (verbo), NOUN (sustantivo). Una referencia visual de cómo se presentan los datos está en la Tabla 5 para la identificación de entidades. El mismo procedimiento se aplicó para relaciones.

Tabla 5. Muestra de matriz numérica con características morfológicas

| adj | adp | aux | det | noun | num | punct | verb | Tipo Entidad | Contexto | POS |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 2 | 2 | 0 | 0 | 0 | Concept | El @ent es la red de vasos | DET AUX DET NOUN ADP NOUN |
| 1 | 2 | 1 | 2 | 3 | 0 | 1 | 1 | Predicate | El sistema vascular es la @ent de vasos sanguíneos del cuerpo. | DET NOUN VERB AUX DET ADP NOUN ADJ ADP NOUN PUNCT |

Por otro lado, las características semánticas se refieren al significado de las palabras en el mundo real de la palabra. Estas características se pueden identificar también en Python con Spacy a través del Reconocimiento de Entidades Nombradas (NER) y entre sus

principales etiquetas están LOC (ubicación geográfica) y PER (Nombres de personas). Referencia en Tabla 6.

Tabla 6. Ejemplo de matriz numérica resultado el conteo de frecuencias de las características semánticas.

| loc | misc | org | per | Tipo Entidad | Contexto NER |
|---|---|---|---|---|---|
| 0 | 4 | 0 | 0 | Concept | red->MISC de->MISC vasos->MISC sanguíneos->MISC del-> .-> |

Las características morfológicas y semánticas se extraen del contexto de la entidad y relaciones para realizar una serie de experimentos, aplicando cada tipo de característica por separado y, posteriormente, combinandolas. Para obtener la matriz numérica, se llevó a cabo un conteo de frecuencias de cada etiqueta.

## 4 Algoritmos de aprendizaje automático

En este estudio se emplean diversos algoritmos para categorizar los textos de interés. Entre ellos se incluyen: Vecinos más cercanos (k-NN), Máquinas de Soporte Vectorial (SVM), Árboles de decisión y Perceptrón. Estos algoritmos se implementaron con la biblioteca scikit-learn de Python.

Vecino más cercano (*k*-NN)

Para la implementación de este algoritmo, hay dos parámetros importantes. El primero es el número de vecinos, para el cual se eligió un valor de K=7 basado en experimentos que mostraron mejores resultados. Esto implica que se consideran los siete elementos más cercanos para asignar la etiqueta categórica correspondiente, y este valor puede ajustarse según las necesidades del problema. El segundo parámetro es el tipo de distancia; en este caso, se utilizó el valor predeterminado, que es la distancia Euclidiana.

Máquinas de soporte vectorial (SVM)

Se utilizó el método SVC de la librería scikit-learn de Python, para la ejecución de este algoritmo. Los parámetros configurados para este algoritmo son la regularización =1.0, kernel ‘rbf’ (kernel gaussiano) y grado del kernel degree establecido en 3.

Árboles de decisión

Este algoritmo se implementó utilizando los valores predeterminados, con el clasificador *DecisionTreeClassifier* de scikit-learn. Entre los parámetros del Árbol de Decisión se encuentran el criterio, que determina la función para medir la calidad de la división (puede ser 'gini' o 'entropy'), y el splitter, que define la estrategia para elegir la división en cada nodo.

Perceptrón

Los parámetros utilizados corresponden a los valores predeterminados del método Perceptrón dentro de la misma biblioteca que los algoritmos mencionados anteriormente. Estos valores incluyen: alpha=0.0001, max_iter=1000, eta0=1.0, tol=1e-3 y shuffle=True.

## 5 Análisis comparativo mediante experimentación y resultados

En este estudio, se llevó a cabo un análisis comparativo de diferentes enfoques para la identificación de entidades en textos en español desbalanceados, utilizando diversas combinaciones de características utilizando el etiquetado POS y etiquetado NER tanto de la entidad misma como de la ventana del contexto (tamaño de ventana 5 palabras). Cada algoritmo fue evaluado por la medida $F_1$, en la identificación de entidades cuando se utilizan diferentes combinaciones de características morfológicas y semánticas. A continuación se presentan los resultados obtenidos en la Tabla 7 y la Figura 3 se visualizan los resultados.

Tabla 7. Resultados obtenidos para la identificación de entidades

| **Algoritmo** | **Entidad etiquetado POS** | **Contexto etiquetado POS** | **Contexto etiquetado NER** | **Entidad POS + Contexto POS + Contexto NER** |
|---|---|---|---|---|
| *k*-NN | 76.97% | 52.99% | 52.66% | 72.13% |
| SVM | 77.22% | 48.91% | 53.04% | **78.72%** |
| Árboles de Decisión | 76.43% | 55.42% | 52.70% | 69.51% |
| Perceptrón | 71.14% | 32.12% | 52.20% | 70.49% |

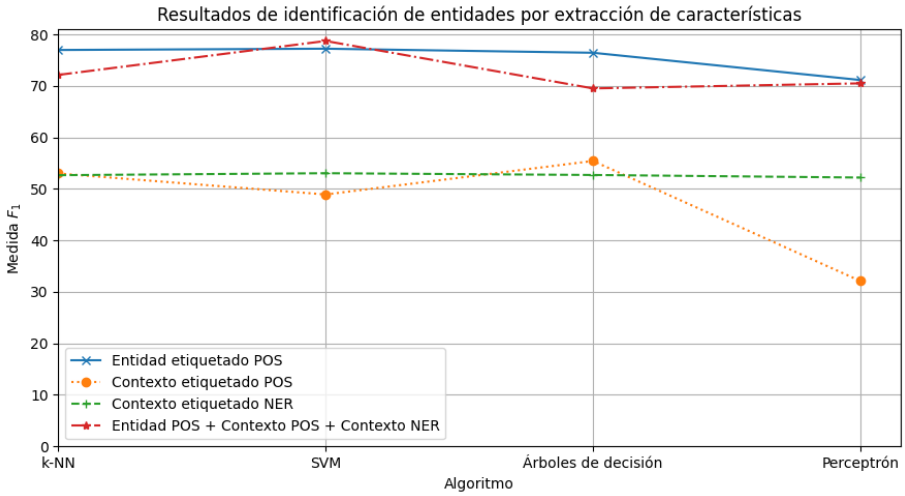


Figura 3. Gráfica de resultados de identificación de entidades

Estos resultados destacan las fortalezas y limitaciones de cada método en la tarea de identificación precisa de entidades, proporcionando una visión integral de cómo diferentes enfoques de procesamiento de lenguaje natural pueden influir en la calidad del análisis de textos médicos en español.

En la Tabla 8 se muestran los resultados de la extracción de relaciones, demostrando que la combinación de características fue el mejor resultado. Para esta comparación se utiliza la información de etiquetado POS y NER del contexto (tamaño de ventana 5 palabras). En la Figura 4 se muestra la gráfica de resultados.

Tabla 8. Resultados obtenidos en la identificación de relaciones

| **Algoritmo** | **Contexto etiquetado POS** | **Contexto etiquetado NER** | **Contexto POS + Contexto NER** |
|---|---|---|---|
| *k*-NN | 28.81% | 10.21% | 34.29% |
| SVM | 26.56% | 17.45% | **35.50%** |
| Árboles de Decisión | 29.12% | 18.38% | 30.96% |
| Perceptrón | 14.20% | 5.40% | 13.23% |

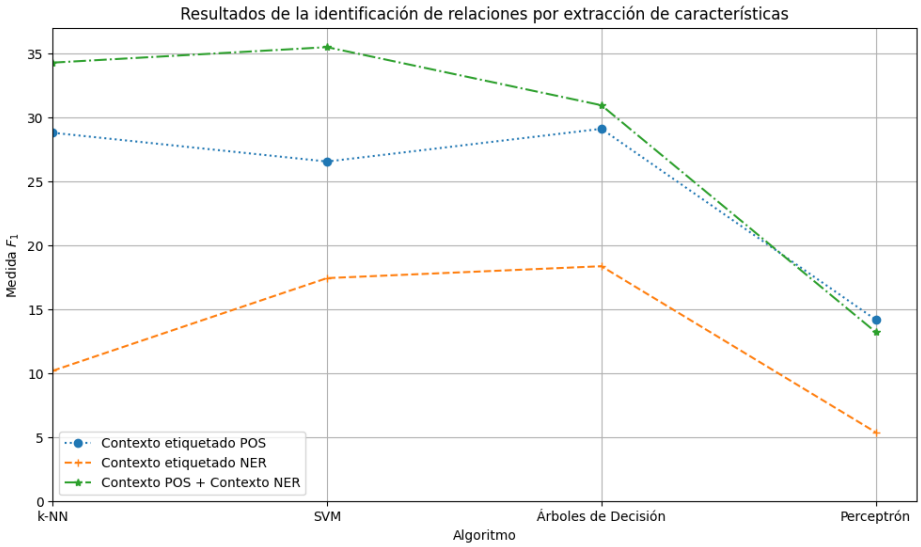


Figura 4. Resultados gráficos de los experimentos para la identificación de relaciones

Para el caso de la identificación automática de entidades, se destaca que las máquinas de soporte vectorial han demostrado ser el algoritmo más efectivo, logrando un 78.72% de exactitud, utilizando información morfológica (etiquetado POS de la entidad y el contexto) y semántica (etiquetado NER del contexto).

Por su parte, la identificación automática de relaciones mostró mejores resultados con máquinas de soporte vectorial, logrando un 35.50% de exactitud, utilizando información morfológica (etiquetado POS del contexto) y semántica (etiquetado NER del contexto).

Si bien los resultados de la extracción de relaciones se pueden observar bajo, este valor compite con los resultados reportados por la literatura para esta tarea de identificación de relaciones a partir de textos en español con conjuntos de textos desbalanceados, que se convierte en un gran reto. El desbalance de los conjuntos afectó de manera evidente en los resultados de la tarea de identificación de relaciones.

## 6 Conclusiones y trabajo a futuro

En este trabajo, se ha implementado una comparación de diversos métodos para la identificación de entidades y relaciones en un conjunto de textos médicos en español desbalanceado. A través de la tokenización etiquetado POS y NER, se logró transformar el texto no estructurado en datos que los algoritmos de aprendizaje automático pueden

procesar. Esta implementación incluye los algoritmos de *k*-NN, SVM, Árboles de Decisión y Perceptrón, ha permitido comparar enfoques y optimizar los resultados.

El algoritmo de Máquinas de Soporte Vectorial mostró un desempeño superior en términos de medida $F_1$, destacándose en la combinación de las características morfológicas y semánticas. Estos resultados se obtuvieron tanto en la identificación de entidades y relaciones, logrando una exactitud de 78.72% y 35.50% respectivamente. Este experimento muestra la importancia de integrar múltiples tipos de características para mejorar el rendimiento de los métodos en la identificación de entidades y relaciones. Se concluye que el desbalance del conjunto de relaciones afectó notablemente el desempeño de los algoritmos de aprendizaje automático.

Como trabajo a futuro, se plantea la implementación de modelos masivos de lenguaje pre entrenados, para experimentar la extracción de conceptos y relaciones en textos médicos desbalanceados. La investigación continua en este ámbito tiene el potencial de enriquecer significativamente el análisis y comprensión de la información médica en español, ofreciendo herramientas más avanzadas para los profesionales de la salud y la investigación clínica.

## Referencias


IberLEF (2021). eHealth Knowledge Discovery Challenge 2021. Recuperado de https://ehealthkd.github.io/2021/resources

Piad-Morffis, A., Guitérrez, Y., Estevez-Velarde, S., & Munoz, R. (2019). A general-purpose annotation model for knowledge discovery: Case study in spanish clinical text. Proceedings of the 2nd Clinical Natural Language Processing Workshop, pp. 79-88.

Aggarwal, C. C. (2018). Machine Learning for Text, Springer Verlag.

Dellanzo, A., Cotik, V., Lozano Barriga, D. Y., Mollapaza Apaza, J. J., Palomino, D., Schiaffino, F., & Ochoa-Luna, J. (2022). "Digital surveillance in Latin American diseases outbreaks: information extraction from a novel Spanish corpus", BMC bioinformatics, vol. 23, pp. 1-22.

Mendoza-Urbano, D. M. (2023) "Extracción automatizada de información en español de texto libre de informes de patología oncológica", Colombia Médica, vol. 54, pp.1-12.

Kaur, A., y Khattar, S. (2021). A systematic exposition of Punjabi Named Entity Recognition using different Machine Learning models. Third International Conference on Inventive Research in Computing Applications (ICIRCA 2021), pp. 1625-1628.

Wei, Q., Ji, Z., Li, Z., Du, J., Wang, J., Xu, J., Xiang, Y., Tiryaki, F., Wu, S., Zhang, Y., Tao, C., y Xu, H. (2020). "A study of deep learning approaches for medication and adverse drug event extraction from clinical text", Journal of the American Medical Informatics Association, vol. 27, pp. 13-21.

Al-Smadi, M., Al-Ayyoub, M., Jararweh, Y., y Qawasmeh, O. (2019). "Enhancing aspect-based sentiment analysis of Arabic hotels' reviews using morphological, syntactic and semantic features". Information Processing & Management, vol. 56, pp. 308-319.

Sanchez-Graillet, O., Witte, C., Grimm, F., & Cimiano, P. (2022). "An annotated corpus of clinical trial publications supporting schema-based relational information extraction", Journal of Biomedical Semantics, vol. 13, pp. 1-18.

Yang, M. (2021). Yunnan-1 at eHealth-KD Challenge 2021: Deep-Learning Methods for Entity Recognition in Medical Text. Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021), pp. 731-736.

Monteagudo-Garcıa, L., Marrero-Santos, A., Fernández-Arias, M. S., & Canizares-Dıaz, H. (2021). Uh-mmm at ehealth-kd challenge 2021. Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021), pp. 692-702

Diccionario de la lengua española (2024) Diccionario de la lengua española. Edición del Tricentenario. Recuperado de https://dle.rae.es/

# Capítulo 8

## Representación ontológica de la arquitectura de control de un robot SCARA utilizando IoT y MATLAB

Gabriel Hurtado Avilés[1], José Manuel Villa Vargas[1], Samyllerick Tapia Hernández[1], Aldair Ángel Rivera Sanabria[1], Jorge Miguel Jaimes Ponce[1]

[1] Universidad Autónoma Metropolitana
Unidad Azcapotzalco
Maestría en Ciencias de la Computación

al2232800343, al2232801439, al2223800675, al2183000368, jjp, @azc.uam.mx

**Resumen.** En el contexto de la Industria 4.0, donde la automatización y la interconexión de dispositivos son fundamentales, surge la necesidad de desarrollar sistemas que permitan el control eficiente de robots en entornos industriales. Este trabajo aborda el desafío de gestionar y controlar un robot *SCARA* mediante la integración de tecnologías de *Internet de las Cosas (IoT)*, lo que permite su operación remota. La complejidad de estos sistemas, que involucran múltiples componentes y relaciones, requiere una representación clara y estructurada del conocimiento. Por ello, se propone una ontología bien definida que facilite la comprensión del sistema y la interacción entre sus elementos. A través de la modelación de esta ontología, se describen los componentes esenciales del sistema y se detalla su implementación en el software *Protégé*, proporcionando así una herramienta que mejora la interoperabilidad y el intercambio de información en el ámbito de la robótica y el *IoT*.

## 1 Introducción

Actualmente la Industria 4.0 (que alude a la 4ta revolución industrial) se enfoca en la creación de fábricas inteligentes en donde cada uno de los elementos que las conforman y que estén involucrados en los procesos de producción puedan estar siempre comunicados unos con otros (L. S. Dalenogare et al, 2018), esto tiene como objetivo optimizar los procesos de producción, reducir costos y disminuir la carga laboral de los empleados (D. Chikurtev et al, 2023).

Siguiendo este argumento de la automatización y conectividad en la industria 4.0, los robots se vuelven cada vez más importantes, ya que son una herramienta flexible de automatización que permite realizar labores repetitivas y peligrosas, permitiendo disminuir los riesgos laborales y el cansancio acumulados de los trabajadores humanos al mismo

tiempo que se logra elevar la consistencia y eficiencia de la producción (S. H. Tay et al, 2022).

Los robots se enfocan en trabajos de operaciones de manejo y colocación de materiales (*pick & place*), en donde los criterios más destacables de estos robots son que tengan certeza de colocación y repetibilidad de procesos, así como una correcta orientación en sus movimientos y que cuenten con tiempos de estabilización y rigidez estática suficientes para desempeñar de la mejor manera sus funciones (CHANAL, Hélène et al, 2022). Es dentro de este contexto que los robots de tipo *SCARA* han tomado relevancia por su diseño y versatilidad.

Originalmente creados por el profesor Hiroshi Makino en 1979 (Barbosa et al, 2020), los robots de tipo *SCARA* son una buena opción que cumple con los requerimientos de la industria 4.0, ya que son de tamaño pequeño, lo cual los vuelve más accesibles. Además, estos pueden ser montados tanto en bases o pedestales como en el suelo (D. Chikurtev et al, 2023) y realizan trabajos con una alta precisión y certeza, todo esto manteniendo una alta eficiencia energética ya que solo 1 de sus articulaciones trabaja en contra de la gravedad (S. H. Tay et al, 2022).

Algunas de las labores en las que se utilizan robots *SCARA* es en el ensamblaje de partes electrónicas, para el manejo de materiales, en el embalaje y en procesos de atornillado (Sonakshi Pradhan et al, 2018). Para alcanzar una alta eficiencia, un robot *SCARA* requiere de un control avanzado que le permita manejar y regular el comportamiento dinámico *no linear* del sistema (S. H. Tay et al, 2022). Es aquí donde el *IoT* cumple una parte fundamental para lograr un control avanzado, en tiempo real y dinámico que se adapte a las distintas funciones que deba cumplir el robot *SCARA*.

Como se define en el artículo de L.A. Grieco et al. (2014), el IoT tiene como principal objetivo conectar dispositivos y personas en cualquier lugar y en cualquier momento. En este contexto, algunos autores han introducido el término *Internet de los robots (IoR)* (L.A. Grieco et al., 2014), que integra las aplicaciones de la robótica con las oportunidades que ofrece el uso del *IoT*. Esto permite la conexión de los sistemas robóticos a Internet, facilitando la obtención de datos sobre el desempeño de estas máquinas (Sonakshi Pradhan et al., 2018).

Finalmente, para definir el concepto de ontología, es útil referirse a las definiciones iniciales con un enfoque filosófico, que se podrían resumir como la especificación formal y explícita de una conceptualización compartida (Gruber, 1993). Ahora bien, aterrizando el concepto de ontología en el campo de la tecnología, se puede usar la pionera definición proporcionada por Neches et al. (1991), quienes definen a la ontología en el campo de las Ciencias de la Computación como la especificación explícita de una conceptualización (Gruber, 1993).

Buscando una definición que sea más comprensible en términos de nuestro campo de interés *(Robótica e IoT*), podemos usar la definición proporcionada por Sharma y Kanjilal (2023), que especifica a una ontología como "*el conocimiento compartido de un campo específico de estudio que puede ser aplicado como un marco unificador para mejorar el intercambio de información, la interoperabilidad y la reutilización entre humanos y máquinas*".

Siguiendo con la utilidad de tener una Ontología acerca de un tema en específico, tenemos que no solo ayudan en la representación eficiente del conocimiento y en la obtención de información, sino que también, ayudan a mapear todo el conocimiento oculto que hay acerca de un tema (Sharma y Kanjilal, 2023).

Teniendo en mente todos estos conceptos, es que en este trabajo se busca unificar las áreas de la *Robótica*, el control mediante *IoT* y la definición de este sistema creando una *Ontología* que lo describa y permita facilitar el compartir el conocimiento generado en este proyecto mediante una estructura clara, universal y abierta.

## 2 Revisión del estado del arte

Respecto a la revisión y estudio del estado del arte para este trabajo, podemos encontrar los trabajos de Tesis de Grado de J.P. Albarez (2023) y de I. Darío et al (2023) en donde de manera eficiente y optimizando los recursos, se puede encontrar el diseño, construcción e implementación de un *robot SCARA*, que además es controlado mediante *IoT*. Este control se logra usando las herramientas *Node-RED*, el protocolo *MQTT* (a través del software *Mosquitto*), una *Raspberry Pi* y usando internet como medio de comunicación.

En el artículo de Chanal et al (2021), se aborda el método para modelar el comportamiento geométrico de *robots SCARA*, con el objetivo de mejorar su precisión final, todo esto después de un proceso de identificación clásico del problema.

En el artículo de Grieco et al (2014) podemos revisar un amplio compendio relacionado al estado del arte del *IoT*, que abarca y define términos como: redes de comunicaciones, seguridad en redes, aplicaciones robóticas en entornos distribuidos y generalizados, diseño de sistemas orientados a la semántica y protocolos de acuerdo basados en la semántica.

En el artículo de Karina Ojo-Gonzalez (2021), podemos encontrar una revisión literaria apoyada en la norma ISO/IEC 25010 para identificar los requerimientos no funcionales prioritarios en cuatro dominios de aplicación: ciudad, hogar, agricultura y fábricas inteligentes. El estudio destaca la importancia de la interoperabilidad, escalabilidad, seguridad y sensibilidad al entorno en los sistemas *IoT*, y resalta cómo estos requerimientos varían según el dominio de aplicación, proporcionando una guía para el diseño y construcción de sistemas *IoT* de alta calidad.

El artículo de Haridy et al (2023), se propone una metodología mejorada para el desarrollo de ontologías, combinando el modelado conceptual impulsado por ontologías *(ODCM)* y el *Procesamiento de Lenguaje Natural (PLN).* Esta metodología, aplicada al dominio del turismo, facilita la construcción de ontologías mediante la identificación de requisitos, conceptualización, formalización e implementación. Además, incluye un módulo de enriquecimiento que sugiere clases y relaciones adicionales utilizando técnicas de *PLN*. La ontología resultante se evalúa mediante preguntas de competencia, demostrando mejoras en concisión y comprensibilidad en comparación con otras ontologías existentes.

Finalmente, en el artículo de Kovács y Csizmás (2018) se identifican diversas propiedades que caracterizan el campo de la ontología aplicada a la ingeniería. Estas

propiedades incluyen modelos de información orientados a conceptos, el apoyo a conceptos abstractos y a nivel de instancia, la inclusión de un sistema lógico para tareas de inferencia y validación, la deducción de nuevos hechos, el uso de un lenguaje de descripción del modelo estándar, la provisión de servicios estándar de gestión del conocimiento, el acceso compartido a los servicios y la utilización de un lenguaje formal de gestión.

También en el texto de Kovács y Csizmás (2018), se mencionan los niveles de las ontologías, siendo la Superior la que cubre conceptos generales en muchas áreas de aplicación, mientras que el *Dominio* se refiere a conceptos específicos usados solo en algunos pocos dominios de aplicación. Adicionalmente, se manifiesta que, con la aplicación de una ontología, los agentes que trabajan mediante *IoT* se pueden comunicar entre ellos usando mensajes generales, preguntando por elementos de información arbitraria.

En el artículo de Sharma y Kanjilal (2023), se establece que la herramienta *Protégé* es uno de los editores más usados para el desarrollo de una ontología, presentando ventajas como ser un editor de licencia de Software-Libre, que fue creada por la Universidad de Stanford y que su interfaz gráfica permite a varios tipos de usuarios construir ontologías sofisticadas, sin la necesidad de un software propietario.

## 3 Especificación del robot *SCARA* como fundamento del modelo ontológico

El modelo ontológico para la Arquitectura de Control de un robot *SCARA* utilizando *IoT* y *MATLAB* se basó en las características de un robot real construido en la UAM Azcapotzalco. La construcción de la ontología se basó en la metodología propuesta por Haridy et al (2023). Este proceso incluyó la medición precisa de parámetros como las longitudes de los eslabones, los ángulos de rotación y los límites de movimiento. Para la integración con el hardware del robot, se realizó un análisis geométrico que implica entender la estructura física del robot y como sus componentes se conectan y se mueven entre sí.

En algunos casos, puede haber múltiples conjuntos de ángulos articulares que resulten en la misma posición y orientación del efector final. Esto se conoce como el problema de la "degeneración cinemática". Además, la cinemática inversa no toma en cuenta las limitaciones físicas del robot, como los límites de rango de movimiento de las articulaciones. Por lo tanto, es importante verificar que las soluciones obtenidas sean válidas y físicamente alcanzables, algo donde se necesita inteligencia, aunque, tal y como Minsky expresó,'El problema de la inteligencia' aún no está completamente resuelto. *"El problema de la Inteligencia Artificial es la naturaleza de la misma inteligencia, un tema que nadie comprende muy bien".* (Minsky M. *"Robotics"*. Omni Publications International. New York, 1985)

El método Denavit-Hartenberg (DH) es una convención que se utiliza para definir de manera consistente los sistemas de coordenadas en un robot para facilitar la derivación de sus ecuaciones cinemáticas. Este método es particularmente útil para simplificar la

descripción de la geometría de un robot, y es ampliamente utilizado en muchas librerías de robótica, incluyendo la librería *Robotics Toolbox* de Peter Corke.

La librería *Robotics Toolbox* de Peter Corke utiliza el método Denavit-Hartenberg (DH) para representar la cinemática de manipuladores de enlace serial como objetos de *MATLAB*. En la librería, cada eslabón del robot se define mediante una serie de parámetros DH, que luego se utilizan para calcular la matriz de transformación homogénea para ese eslabón. Esta matriz de transformación se utiliza para calcular la posición y orientación del eslabón en el espacio.

La cinemática inversa juega un papel fundamental en el sistema de control del robot *SCARA*. Los métodos de cinemática inversa, como el método Denavit-Hartenberg o el método geométrico, se utilizan para calcular los ángulos articulares necesarios para que el efector final del robot alcance una posición y orientación deseadas.

Esta información de ángulos articulares se utiliza posteriormente por el controlador del robot para enviar las señales de control a los motores, permitiendo así el movimiento del robot.

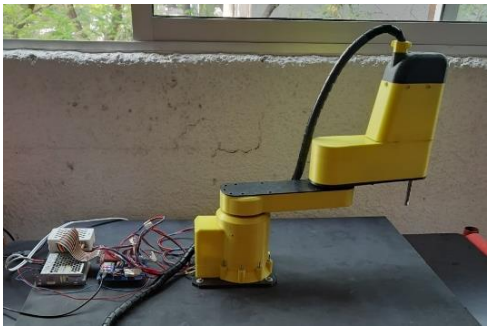

Figura 1. Robot de tipo *SCARA* utilizado en el proyecto

Supongamos que se desea que el efector final del robot *SCARA* alcance una posición específica en el espacio. El *Dashboard* de *Node-RED* (Ver Figura 2) recibe y envía esta información a través del protocolo *MQTT*. El *ESP32* interpreta la información y controla los motores del robot para que el efector final alcance la posición deseada, logrando así el movimiento del robot.


Figura 2. *Dashboard* utilizado en el proyecto

El sistema completo de control para el robot *SCARA* incluye los siguientes componentes:

- **Microcontrolador *ESP32*:** El *ESP32*, un microcontrolador versátil con conectividad *WiFi* y *Bluetooth*, se utiliza para enviar y recibir comandos de control al robot.
- ***Raspberry Pi*:** La *Raspberry Pi* actúa como servidor e intermediario entre el *ESP32* y el software de control, facilitando la comunicación y el procesamiento de datos.
- **Software *Node-RED*:** *Node-RED*, una herramienta de programación basada en flujo, se utiliza para diseñar y gestionar el flujo de datos entre el *ESP32*, la *Raspberry Pi* y el *dashboard*.
- **Protocolo *MQTT* a través de *Mosquitto*:** El protocolo *MQTT*, un protocolo de mensajería ligero ideal para dispositivos con recursos limitados, se utiliza en conjunto con el *broker Mosquitto* para facilitar la comunicación entre los componentes del sistema.
- ***Dashboard* de *Node-RED*:** El *dashboard* gráfico de *Node-RED* proporciona una interfaz visual para el control y monitoreo en tiempo real del robot *SCARA*.
- **Zona *WiFi* en *Windows 10*:** Se configura una zona *WiFi* en un sistema operativo Windows 10 para conectar de forma inalámbrica el *ESP32* y la *Raspberry Pi*, asegurando la comunicación entre estos dispositivos.

La complejidad del sistema de control del robot *SCARA* radica en la interacción de múltiples componentes, cada uno con sus propias funciones y características. Además, la integración de la cinemática inversa añade una capa de complejidad adicional, ya que involucra cálculos matemáticos y conceptos específicos de robótica.

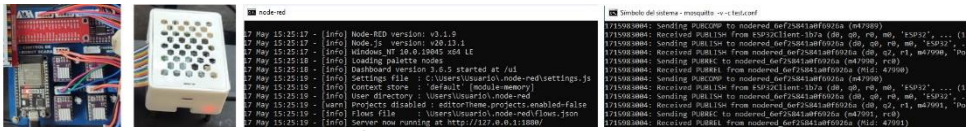

Figura 3. De izq. a der., imágenes de Esp32, Raspberry Pi, Node-RED y Mosquitto

Una ontología no se debe confundir con un Mapa Mental, pues *"Una ontología define los términos básicos y las relaciones que comprenden el vocabulario de una área o tópico, así como las reglas para combinar los términos y las relaciones que definen las extensiones al vocabulario". (Neches et al., 1991).* Por ello, el uso de una ontología para explicar el sistema de control del robot *SCARA* permite representar el conocimiento de manera formal. Al formalizar el conocimiento sobre el sistema, la ontología facilita la navegación entre los conceptos y la comprensión de las relaciones entre ellos. La ontología puede ser utilizada para realizar razonamiento automático sobre el sistema, permitiendo análisis de escenarios hipotéticos y la identificación de posibles problemas.

# 4 Construcción de la ontología para el sistema de control del robot *SCARA* utilizando *IoT* y *MATLAB*

Una ontología es una representación formal y explícita de un dominio de conocimiento. En este caso, se presenta una ontología para describir el sistema de control de un robot *SCARA* utilizando *IoT* y *MATLAB,* incluyendo sus componentes, relaciones y propiedades.

Tabla 1. Etiquetas de la Ontología en Protégé

| Clase | Relación | Clase Relacionada |
|---|---|---|
| **RobotSCARA** | isControlledBy | ControlRemoto |
| **RobotSCARA** | communicatesWith | ESP32 |
| **ESP32** | communicatesWith | RaspberryPi, LANWiFi, MQTT |
| **RaspberryPi** | communicatesWith | LANWiFi |
| **RaspberryPi** | hasOS | RaspberryPiOS |
| **MQTT** | communicatesWith | Mosquitto |
| **NodeRED** | communicatesWith | Mosquitto,Dashboard, MATLAB |
| **MATLAB** | communicatesWith | BrokerMensajeria |
| **MATLAB** | hasOS | SistemaOperativo |
| **SistemaOperativo** | communicatesWith | Computadora, |
| **Computadora** | communicatesWith | BrokerMensajeria |

La ontología (Ver Figura 4) utiliza el lenguaje *OWL* (*Web Ontology Language*) para definir las clases propiedades y relaciones entre los elementos del sistema. A su vez, proporciona una base para comprender el funcionamiento del sistema de control del robot.

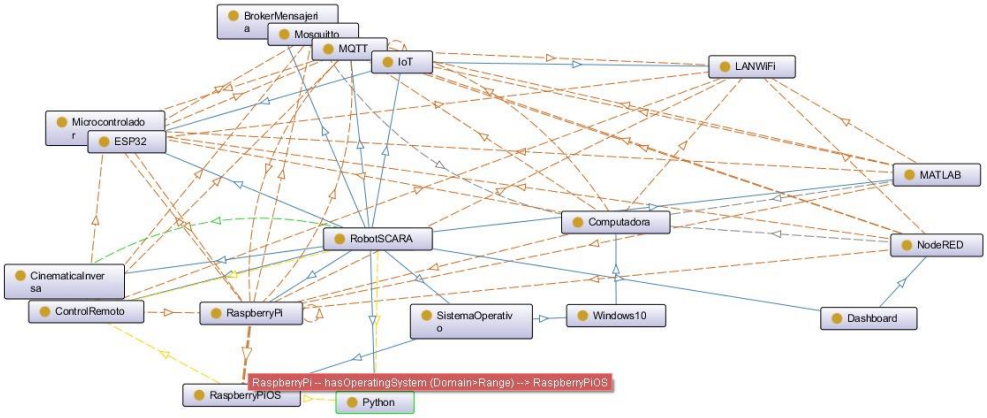


Figura 4. Diagrama de la Ontología en Protégé

La ontología (Ver Figura 5) define subclases (Ver Tabla 2, 3 y 4) para representar relaciones jerárquicas entre clases y propiedades.

Tabla 2. Clases principales de la Ontología en Protégé

| Clase | Descripción |
|---|---|
| **RobotSCARA** | Representa el robot *SCARA* completo. |
| **Microcontrolador** | Representa un microcontrolador genérico. |
| **ESP32** | Representa el microcontrolador *ESP32* utilizado en el sistema. |
| **RaspberryPi** | Representa la computadora *Raspberry Pi* utilizada en el sistema. |
| **MQTT** | Representa el protocolo de mensajería *MQTT*. |
| **BrokerMensajeria** | Representa un *broker* de mensajería utilizado para el protocolo *MQTT* (*Mosquitto* en este caso). |
| **Dashboard** | Representa una interfaz gráfica de usuario (*dashboard*). |
| **NodeRED** | Representa el software *Node-RED* utilizado para programar el flujo de datos. |
| **MATLAB** | Representa el software de cómputo técnico *MATLAB*. |
| **CinematicaInversa** | Representa el concepto de cinemática inversa. |
| **SistemaOperativo** | Representa un sistema operativo genérico. |
| **Windows10** | Representa el sistema operativo Windows 10. |
| **RaspberryPiOS** | Representa el sistema operativo *Raspberry Pi OS*. |
| **LANWiFi** | Representa una red de área local inalámbrica (*WiFi*). |
| **IoT** | Representa el *Internet de las Cosas* (*IoT*). |
| **ControlRemoto** | Representa un control remoto para el robot *SCARA*. |

Tabla 3. Clases principales de la Ontología en Protégé

| Subclase | Superclase |
|---|---|
| **Mosquitto** | BrokerMensajeria |
| **NodeRED** | Dashboard |
| **RaspberryPiOS** | SistemaOperativo |
| **Windows10** | SistemaOperativo |
| **Computadora** | Windows10 |
| **CinemáticaInversa** | RobotSCARA |
| **LANWiFi** | IoT |
| **ESP32** | IoT |

Tabla 4. Propiedades principales de la Ontología en Protégé

| Propiedad | Dominio | Rango | Descripción |
|---|---|---|---|
| **isControlledBy** | RobotSCARA | ControlRemoto | Indica como se controla el robot *SCARA*. |
| **communicatesWith** | Clase | Clase | Indica que dos clases se pueden comunicar entre sí. |

| | | | |
|---|---|---|---|
| **hasOS** | Computadora | SistemaOperativo | Indica el sistema operativo que se ejecuta en una computadora. |

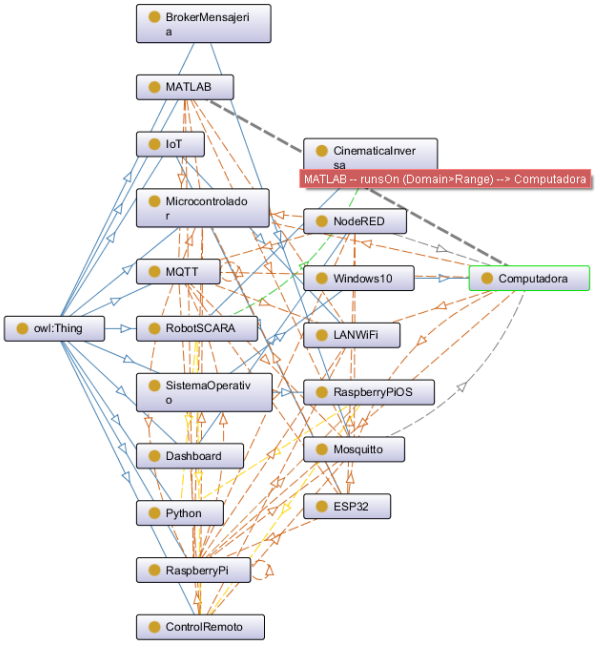


Figura 5. Vista alternativa del diagrama de la Ontología

Por otro lado, la ontología define restricciones en el uso de las propiedades objeto para asegurar la coherencia del modelo:

- **isControlledBy:** Solo los robots *SCARA* pueden ser controlados remotamente.
- **communicatesWith:**
  - El *ESP32* se comunica con la *Raspberry Pi*.
  - El *ESP32* se comunica con otro microcontrolador.
  - El protocolo *MQTT* se comunica con un *broker* de mensajería (*Mosquitto*).
  - *NodeRED* se comunica con un *broker* de mensajería (*Mosquitto*).
  - *MATLAB* se comunica con un *broker* de mensajería (*Mosquitto*).
  - Una computadora se comunica con un broker de mensajería (*Mosquitto*).
  - La *Raspberry Pi* se comunica con una red *WiFi* (*LAN*).
  - El *ESP32* se comunica con una red *WiFi* (*LAN*).
  - Un control remoto se comunica con un *broker* de mensajería (*MQTT*).
- **hasOperatingSystem:** Solo las computadoras pueden tener un sistema operativo.

Al navegar por la jerarquía de clases, las relaciones entre objetos y las propiedades asociadas, se puede obtener una imagen más completa de cómo interactúan los diferentes componentes

## 5 Evaluación de la ontología

La evaluación del modelo ontológico se centra en la extensibilidad y adaptación del sistema, utilizando cinco escenarios donde se medirá el razonamiento e inferencia a través de preguntas formuladas a la ontología. A continuación, se presentan las preguntas realizadas, junto con sus respuestas correspondientes.

Tabla 5. Preguntas de evaluación de la ontología

| Categoría | Pregunta | Respuesta |
|---|---|---|
| **Clasificación y Subsunción** | ¿Es una RaspberryPi un tipo de Computadora? | Sí, ya que RaspberryPi es una subclase de Computadora. |
| | ¿Es Mosquitto un tipo de Broker de Mensajería? | Sí, ya que Mosquitto es una subclase de BrokerMensajeria. |
| **Propiedades de Objeto** | ¿Quién controla al RobotSCARA? | ControlRemoto, ya que el RobotSCARA es controlado mediante ControlRemoto. |
| | ¿Con qué se comunica el NodeRED? | Mosquitto, ya que NodeRED se comunica con Mosquitto. |
| | ¿Qué sistema operativo tiene una RaspberryPi? | RaspberryPiOS, ya que la RaspberryPi es una computadora que utiliza el sistema operativo RaspberryPiOS. |
| **Propiedades de Datos** | ¿Cuál es la dimensión de un RobotSCARA? | Un valor entero, ya que un RobotSCARA tiene dimensiones que son números enteros. |
| | ¿Tiene un RobotSCARA limitaciones en las articulaciones? | Un valor booleano, ya que el RobotSCARA si tiene limitaciones en sus articulaciones. |
| | ¿Cuál es la versión del sistema operativo de una Computadora? | Un valor de cadena, ya que una Computadora tiene una versión especifica de algún sistema operativo. |
| **Inferencia y Razonamiento** | Si un dispositivo se comunica con Mosquitto, ¿se puede | Sí, ya que Mosquitto es una subclase de BrokerMensajeria, y |

| | | |
|---|---|---|
| | inferir que es un dispositivo IoT? | BrokerMensajeria está relacionado con IoT. |
| | Si un RobotSCARA tiene cinemática inversa, ¿se puede inferir que es capaz de realizar movimientos? | Sí, se puede inferir que un RobotSCARA con cinemática inversa tiene la capacidad de realizar movimientos, ya que esta técnica permite calcular las posiciones necesarias para alcanzar un objetivo específico. |
| **Consulta y Coincidencia de Patrones** | Encuentra todos los dispositivos que pueden ser controlados de forma remota. | Robot SCARA, RaspberryPi, Control Remoto. |
| | Encuentra todos los componentes que forman parte de la arquitectura de control del Robot SCARA. | Raspberry Pi, ESP32, NodeRED, Mosquitto, MATLAB, Control Remoto. |

# 6 Aplicaciones a futuro del modelo construido de un robot SCARA

El modelo ontológico del sistema de control del robot *SCARA* no solo permite comprender el funcionamiento del sistema actual, sino también sienta las bases para explorar futuras aplicaciones del robot.

El diseño de la ontología puede seguir desarrollándose en el futuro para realizar otras tareas que realice el robot *SCARA*, como el dibujo de letras del alfabeto. Actualmente, esta tarea se puede realizar en simulación mediante herramientas como *MATLAB*, que permite implementar diferentes métodos de cinemática de robots (directa e inversa) para que el robot dibuje letras con precisión (ver Figura 7).

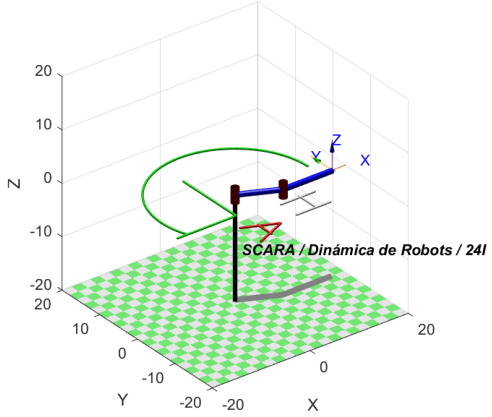


Figura 7. Simulación del dibujo de letras con un Robot *SCARA* en *MATLAB*

Una de las áreas con mayor potencial es la integración de la *Inteligencia Artificial (IA)* en las funcionalidades del robot. La *IA* podría dotar al robot de la capacidad de percibir y comprender el entorno, identificar objetos, reconocer sus características y planificar acciones para manipularlos. La utilización de ontologías podría formalizar este entendimiento y aplicarlo en tareas específicas, como el ensamblaje de piezas, la clasificación de objetos o la asistencia en actividades domésticas.

La integración de la *Inteligencia Artificial* en el modelo de robot *SCARA* descrito abre un abanico de posibilidades para el desarrollo de nuevas aplicaciones en diversos campos. La capacidad del robot para percibir, aprender, razonar y actuar de manera autónoma lo posiciona como una herramienta eficaz para abordar problemas complejos.

## 7 Conclusiones

En este trabajo, se logró con éxito diseñar una ontología que describe el sistema que controla un robot *SCARA* real utilizando una combinación de tecnologías modernas que trabajan sobre *IoT (ESP32, Raspberry Pi, Node-RED, Mosquitto)*. La interconexión de estos componentes permitió una comunicación fluida y en tiempo real para la manipulación eficiente del robot *SCARA*.

A partir de este diseño presentado *(SCARA-IoT*), se desarrolló una ontología que describe detalladamente el sistema de interconexión y componentes de los elementos involucrados en el control del robot *SCARA*.

La ontología desarrollada no solo facilitará la representación y gestión del conocimiento, sino que también promoverá la colaboración y el avance en estos campos tecnológicos, además de lograr un mayor entendimiento de la propuesta presentada para personas que no estén directamente involucradas en los campos de la robótica y del *IoT*. Contar con una ontología como la creada facilita el intercambio de información, la interoperabilidad, la comprensión del sistema y la reutilización de los datos generados en un modelo como este.

## Referencias


Albarez Flores, J. P. (2023). *"Diseño e implementación de un brazo robótico, basado en IoT, para el Laboratorio de la Carrera de Ingeniería Electrónica"*. [Tesis de grado]. Universidad Mayor de San Andrés. Recuperado de https://repositorio.umsa.bo/bitstream/handle/123456789/32147/PG-8428.pdf?sequence=1&isAllowed=y

Barbosa, W. S., Gioia, M. M., Natividade, V. G., Wanderley, R. F., Chaves, M. R., Gouvea, F. C. y Gonçalves, F. M. (2020). *"Industry 4.0: examples of the use of the robotic arm for digital manufacturing processes."* International Journal on Interactive Design and Manufacturing (IJIDeM), 14, 1569-1575. DOI: https://doi.org/10.1007/s12008-020-00714-4

Bhatia, M. P. S., Kumar, A., Beniwal, R. y Malik, T. (2020). Ontology driven software development for automatic detection and updation of software requirement specifications. Journal of Discrete

Mathematical Sciences and Cryptography, 23(1), 197-208. DOI: https://doi.org/10.1080/09720529.2020.1721884

Chanal, H., Guyon, J. B., Koessler, A., Dechambre, Q., Boudon, B., Blaysat, B. y Bouton, N. (2021). *"Geometrical defect identification of a SCARA robot from a vector modeling of kinematic joints invariants."* Mechanism and Machine Theory, 162, 104339. DOI: https://doi.org/10.1016/j.mechmachtheory.2021.104339

Chikurtev, D., Al-Moghrabi, N. y Chikurteva, A. (2023, julio). *"Control of SCARA Robot and Rotary Table in ROS and Gazebo as Part of a Cyber-Physical System"*. En 2023 9th International Conference on Control, Decision and Information Technologies (CoDIT) (pp. 1355-1360). IEEE. https://doi.org/10.1109/CoDIT58514.2023.10284442

Dalenogare, L. S., Benitez, G. B., Ayala, N. F. y Frank, A. G. (2018). *"The expected contribution of Industry 4.0 technologies for industrial performance."* International Journal of Production Economics, 204, 383-394. DOI: https://doi.org/10.1016/j.ijpe.2018.08.019

Darío, I., Peñafiel Sánchez, A. R. y Alulema Flores, D. O. (2023). *"Diseño e implementación de una celda de manufactura remota basada en un robot SCARA con aplicación de visión artificial para su aplicación en los laboratorios remotos"* [Tesis de grado]. Universidad de las Fuerzas Armadas. Recuperado de https://repositorio.espe.edu.ec/bitstream/21000/35441/1/T-ESPE-052667.pdf

Grieco, L. A., Rizzo, A., Colucci, S., Sicari, S., Piro, G., Di Paola, D. y Boggia, G. (2014). *"IoT-aided robotics applications: Technological implications, target domains and open issues"*. Computer Communications, 54, 32-47. DOI: https://doi.org/10.1016/j.comcom.2014.07.013

Haridy, S., Ismail, R. M., Badr, N., & Hashem, M. (2023). An ontology development methodology based on ontology-driven conceptual modeling and natural language processing: Tourism case study. Big Data and Cognitive Computing, 7(2), 101. https://doi.org/10.3390/bdcc7020101

Kovács, L. y Csizmás, E. (2018). *"Lightweight ontology in IoT architecture"*. En 2018 IEEE International Conference on Future IoT Technologies (Future IoT), Eger, Hungría (pp. 1-6). https://doi.org/10.1109/FIOT.2018.8325591

Minsky, M. (1985). *"Our Robotized Future"*. En M. Minsky (Ed.), *Robotics* (pp. 286-307). Omni Publications International.

Neches R, Fikes RE, Finin T, Gruber T, Patil R, Senator T, et al. *"Enabling technology for knowledge sharing"*. AI Mag. 1991;12(3):36. https://doi.org/10.1609/aimag.v12i3.902

Ojo-Gonzalez, K., & Bonilla-Morales, B. (2021). Requerimientos no funcionales para sistemas basados en el Internet de las cosas (IoT): Una revisión. I+D Tecnológico, 17(2), 30-40. https://doi.org/10.33412/idt.v17.2.3303

Pradhan, S., Rajarajan, K. y Shetty, A. S. (2018). *"Prototype, emulation, implementation and evaluation of SCARA Robot in industrial environment"*. Procedia Computer Science, 133, 331-337. DOI: https://doi.org/10.1016/j.procs.2018.07.041

Sharma, R. y Kanjilal, U. (2023). *"Developing Energy Access Ontology using Protege Tool."* DESIDOC Journal of Library & Information Technology, 43(3), 169-175. https://doi.org/10.14429/djlit.43.03.18379

Tay, S. H., Choong, W. H. y Yoong, H. P. (2022). *"A Review of SCARA Robot Control System"*. En 2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET), Kota Kinabalu, Malasia (pp. 1-6). https://doi.org/10.1109/IICAIET55139.2022.9936755

# Capítulo 9

## Aprendizaje por refuerzo aplicado en la cultura física

Edgar Sahit Aguilar Hernández [1]

[1] Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación

edgar.aguilarh@alumno.buap.mx

**Resumen.** En el presente trabajo se hace una revisión de cómo se ha utilizado la inteligencia artificial y otras técnicas en el ámbito de la cultura física para maximizar la hipertrofia muscular. Se describen los sistemas de aprendizaje por refuerzo, como Q-learning y Deep Q-learning, empleados para diseñar y entrenar un agente que optimiza los programas de entrenamiento. Se presenta un modelo donde un agente, representando el estado de la musculatura del bíceps y tríceps, progresa desde un estado inicial a uno más desarrollado, medido en centímetros del brazo. El modelo considera 16 estados diferentes del brazo y 4 acciones correspondientes a ejercicios específicos. Cada ejercicio proporciona una recompensa basada en un puntaje extraído de un conjunto de datos de ejercicios para distintos grupos musculares. Este enfoque permite simular, personalizar y mejorar los programas de entrenamiento para lograr una hipertrofia muscular eficiente y efectiva.

**Palabras Clave:** Q-learning, Deep Q-learning, Aprendizaje por refuerzo, simulación, hipertrofia muscular.

## 1 Introducción

La inclusión de herramientas de inteligencia artificial en el ámbito de la cultura física es necesaria debido a su potencial para revolucionar la simulación, personalización y optimización de los programas de entrenamiento. Tradicionalmente, la creación de rutinas de ejercicio eficaces ha dependido de la experiencia y el juicio humano, lo cual, aunque valioso, puede ser limitado por la subjetividad y la variabilidad individual. La inteligencia artificial, específicamente el aprendizaje por refuerzo ofrece una solución a estos desafíos al permitir la creación de sistemas de entrenamiento adaptativos que pueden ajustarse dinámicamente a las necesidades y progresos individuales. Esta capacidad de adaptación y personalización no solo mejora la eficiencia y efectividad de los programas de entrenamiento, sino que también puede acelerar los resultados deseados, como la hipertrofia muscular. Además, la inteligencia artificial puede analizar grandes volúmenes de datos para identificar patrones y tendencias que podrían pasar desapercibidos para los entrenadores humanos, ofreciendo puntos de vista más profundos y basados en evidencia para la toma de

decisiones. Por lo tanto, la implementación de estas tecnologías en la cultura física no solo es un avance lógico, sino también una necesidad para maximizar los beneficios del entrenamiento físico en una era donde los datos y la tecnología juegan un papel crucial en todas las facetas de la vida cotidiana.

En particular, el aprendizaje por refuerzo ha emergido como una técnica poderosa para diseñar y simular sistemas de entrenamiento adaptativos. Algoritmos como Q-learning y Deep Q-learning permiten entrenar agentes que toman decisiones basadas en el estado actual de un sistema y las posibles recompensas de diferentes acciones.

En este estudio, se implementa un modelo donde un agente representa a una persona y el estado de la musculatura del bíceps, con el objetivo de progresar desde un estado inicial hasta uno más desarrollado, medido en centímetros del brazo. Se consideran 16 estados diferentes del bíceps y 4 acciones correspondientes a ejercicios específicos, cada uno proporcionando una recompensa basada en un puntaje extraído de un conjunto de datos de ejercicios para distintos grupos musculares. Este enfoque promete mejorar la eficiencia y efectividad de los programas de entrenamiento para lograr una hipertrofia muscular óptima, demostrando cómo las herramientas de inteligencia artificial, específicamente el aprendizaje por refuerzo, pueden contribuir a explorar diversos escenarios en los que un sujeto se puede encontrar durante el entrenamiento muscular. Aunque en este estudio se utiliza el bíceps como ejemplo, las técnicas de aprendizaje por refuerzo presentadas son aplicables para el entrenamiento de diferentes músculos o grupos musculares, evidenciando cómo estas tecnologías pueden ser utilizadas para simular y analizar situaciones diversas, facilitando un mayor número de exploraciones y optimizando las estrategias de entrenamiento.

A continuación, se describe la organización de este documento. En la sección 1 se da una introducción al problema de estudio. En la sección 2 se describen los conceptos básicos requeridos para plantear el trabajo de investigación en el diseño de los algoritmos de aprendizaje por refuerzo para la simulación de la hipertrofia muscular. En la sección 3 se presenta el trabajo relacionado con la investigación de la hipertrofia muscular, así como el uso de la inteligencia artificial para el área de la cultura física. En la sección 4 se presenta el diseño del entorno en el cual nuestro agente va a interactuar, así como la descripción de cada uno de los algoritmos utilizados. En la sección 5 se presentan los resultados obtenidos después de entrenar a nuestro agente. En la sección 6 se presentan las conclusiones del trabajo desarrollado.

## 2 Marco teórico

En esta sección del marco teórico, se presentan los fundamentos de los algoritmos de aprendizaje por refuerzo, específicamente Q-learning y Deep Q-learning, así como los principios expuestos en el "Libro Negro de los Secretos del Entrenamiento para la Hipertrofia Muscular" (Christian Thibaudeau, 2007). Se describirán los principios básicos de cada algoritmo y sus mecanismos de funcionamiento. Esta comprensión teórica es

crucial para contextualizar el uso de estas tecnologías en el diseño de sistemas de entrenamiento adaptativos y eficientes.

### 2.1 Hipertrofia muscular

La hipertrofia muscular es el aumento del tamaño de las fibras musculares, lo cual resulta en un crecimiento del músculo. Este fenómeno ocurre principalmente como respuesta a la sobrecarga mecánica y el estrés metabólico que se generan durante el entrenamiento de resistencia y el levantamiento de pesas (Christian Thibaudeau, 2007).

### 2.2 Tensión intramuscular y Tiempo total bajo tensión

El libro "El Libro Negro de Los Secretos de Entrenamiento" (Christian Thibaudeau, 2007) nos maneja dos conceptos importantes para la hipertrofia, Tensión intramuscular y Tiempo total bajo tensión. La tensión intramuscular se refiere al esfuerzo necesario del músculo para generar una cierta producción de fuerza. Como ya es conocido la fuerza es igual a masa por aceleración, así que debe resultar también evidente que la tensión intramuscular se verá influida por la magnitud de la carga y la aceleración que uno tiene que transmitir a la resistencia. En palabras más simples, se puede aumentar la tensión intramuscular aumentando el peso, la aceleración o ambos. La importancia de la tensión presente en el músculo es el principal factor responsable de la calidad de las ganancias estimuladas, cuanto más alta la tensión intramuscular, mayor estimulación de hipertrofia funcional habrá. Además, una alta tensión intramuscular aumenta la tasa de degradación proteica y la subsiguiente recepción de aminoácidos por parte de los músculos. TBT-Tiempo total bajo tensión es el factor responsable principal para la cantidad de hipertrofia estimulada. Un mayor volumen de trabajo estimulará más hipertrofia (mientras el estímulo no exceda la capacidad de recuperarse). Idealmente se debe maximizar la tensión utilizando o bien una carga pesada, o bien levantando la carga tan rápido como sea posible al tiempo que la baja lentamente. Si se escoge una carga con la que puede realizar 1-5 reps, se debe hacer más series para conseguir un fuerte estímulo de crecimiento (Christian Thibaudeau, 2007).

### 2.3 Terminología clave para Q-Learning y Deep Q-Learning

Antes de entrar de lleno en cómo funciona el Q-learning y Deep Q-learning, necesitamos aprender algunas terminologías útiles (Abid Ali Awan, 2024).

**Estado(*s*)**: posición actual del agente en el entorno.

**Acción(*a*)**: paso dado por el agente en un estado determinado.

**Recompensas**: por cada acción, el agente recibe una recompensa y una penalización.

**Episodios**: el final de la etapa, donde los agentes no pueden emprender nuevas acciones. Se produce cuando el agente ha alcanzado el objetivo o ha fracasado.

### 2.4 Q-Learning

Q-learning es una forma sencilla para que los agentes aprendan cómo actuar de manera óptima en dominios markovianos controlados. Equivale a un método incremental para la programación dinámica que impone demandas computacionales limitadas. Funciona mejorando sucesivamente sus evaluaciones de la calidad de acciones particulares en estados particulares (Christopher j.c.h. watkins, 1992).

Un agente intenta una acción en un estado particular y evalúa sus consecuencias en términos de la recompensa o penalización inmediata que recibe y su estimación del valor del estado al que es llevado. Al intentar todas las acciones en todos los estados repetidamente, aprende cuáles son las mejores en general, a juzgar por la recompensa con descuento a largo plazo (Christopher j.c.h. watkins, 1992).

En el Q-learning, la experiencia del agente consiste en una secuencia de distintas etapas o episodios. En el *n* episodio, el agente (Christopher j.c.h. watkins, 1992):

- observa su estado actual $x_n$,
- selecciona y realiza una acción $a_n$,
- observa el estado posterior $Y_n$,
- recibe una recompensa inmediata $r_n$ y
- ajusta sus valores $Q_{n-1}$ usando un factor de aprendizaje.

Ahora este algoritmo va actualizando los *Q* valores de la tabla, en función de la Ecuación 1:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)] \quad (1)$$

Donde:

$\alpha$ es la tasa de aprendizaje,
$\gamma$ es el factor de descuento,
$r$ es la recompensa obtenida después de realizar la acción a en el estado s,
$s'$ es el nuevo estado resultante de la acción a,
$a'$ es la acción que maximiza la función Q en el estado siguiente.

### 2.5 Deep Q-Network

Deep Q-Learning es una técnica de aprendizaje por refuerzo que combina Q-Learning, un algoritmo para aprender acciones óptimas en un entorno, con redes neuronales profundas. Su objetivo es permitir que los agentes aprendan acciones óptimas en entornos

complejos y de alta dimensión. Al utilizar una red neuronal para aproximar la función *Q*, que estima la recompensa acumulada esperada por cada acción en un estado determinado, Deep Q-Learning puede manejar entornos con grandes espacios de estado. La red se actualiza de forma iterativa a través de episodios, utilizando una combinación de estrategias de exploración y explotación. Sin embargo, se debe tener cuidado para mitigar la inestabilidad causada por problemas de no estacionariedad y divergencia, que generalmente se abordan mediante la repetición de experiencias y las redes de destino (Ankit Choudhary,2023)

## 3 Estado del arte

En esta sección se examina el trabajo relacionado con la investigación de la hipertrofia muscular y el uso de la inteligencia artificial en el ámbito de la cultura física. Esta sección proporciona una revisión detallada de estudios previos y enfoques actuales que han aplicado distintas técnicas, así como el uso de inteligencia artificial para optimizar los programas de entrenamiento, destacando cómo estas tecnologías están transformando las estrategias de ejercicio y mejorando los resultados en el desarrollo muscular.

En la actualidad, existen numerosas técnicas avanzadas para el monitoreo de las ganancias musculares en individuos que entrenan con el objetivo de alcanzar la hipertrofia muscular. Identificar los métodos y rutinas de entrenamiento más eficaces para lograr este objetivo es fundamental. A continuación, exploraremos algunas de estas técnicas y los métodos utilizados.

El estudio titulado “Efectos de la frecuencia semanal de entrenamiento sobre la hipertrofia y la fuerza muscular en individuos entrenados” se realizó con el objetivo de comparar los efectos sobre la hipertrofia y la fuerza muscular de dos rutinas de entrenamiento con el mismo volumen y diferente frecuencia de entrenamiento semanal en el cual catorce sujetos entrenados (22 ± 2 años) se dividieron en 2 grupos experimentales que entrenaron durante 6 semanas. Un grupo (*G1*) entrenó cada grupo muscular una vez por semana con una rutina Split de entrenamiento, y el otro grupo (*G2*) entrenó cada grupo muscular tres veces por semana con una rutina *Fullbody*. Se realizó un volumen semanal de nueve series de repeticiones por cada grupo muscular. Se midió la composición corporal, los perímetros musculares y la fuerza máxima mediante test de una repetición máxima (1RM) antes y después del entrenamiento. Este estudio encontró que se produjo un aumento de la hipertrofia muscular en ambos grupos, aunque sin diferencias significativas entre ellos. En relación con la fuerza muscular, *G1* obtuvo mayores mejoras que *G2* en las pruebas de 1RM de press de banca (11.2 ± 4.3 vs. 7.3 ± 3.6 kg), curl de bíceps (3.1 ± 0.8 vs. 2.2 ± 0.7 kg) y elevaciones laterales (2.9 ± 1 vs. 1.6 ± 0.8 kg). Por tanto, se llegó a la conclusión de que ambas rutinas de entrenamiento con un mismo volumen semanal produjeron efectos similares sobre la hipertrofia muscular, sin embargo, la rutina en Split resultó más efectiva que la rutina Fullbody para producir aumentos de la fuerza muscular del tren superior (R. Timón, 2017).

En otro estudio titulado “Entrenamiento de fuerza y resistencia en hipoxia: efecto en la hipertrofia muscular” se encontró que la sinergia del entrenamiento de fuerza y resistencia y la hipoxia normobárica produce mejores y mayores adaptaciones, ganancias y cambios fisiológicos beneficiosos en el tejido muscular, lo cual genera cambios fenotípicos favorables, como la hipertrofia del músculo esquelético. Esta revisión permitió concluir que la sinergia del entrenamiento y la hipoxia normobárica produce mejores y mayores adaptaciones, ganancias y cambios fisiológicos beneficiosos en el tejido muscular, lo cual trae consigo transformaciones fenotípicas favorables como la hipertrofia del músculo esquelético cuando se garantizan condiciones óptimas de entrenamiento e hipoxia. Para ello deben considerarse la dosis de hipoxia, la duración de las sesiones, la intensidad del ejercicio, la frecuencia semanal, las semanas de tratamiento y el número de sesiones. Además, deben contemplarse los aspectos dietéticos y nutricionales, así como la administración de los nutrientes necesarios (Diego Fernández, 2019).

En el estudio titulado “Revisión de los procesos de hipertrofia muscular inducida por el entrenamiento de fuerza oclusivo” el American College of Sports Medicine recomienda el uso de intensidades superiores al 70% con 1 repetición máxima (RM) para inducir hipertrofia a través del entrenamiento con resistencias. El entrenamiento de alta intensidad produce una activación máxima de la musculatura, tanto a nivel neurológico como mecánico. En cambio, el ejercicio con resistencias de intensidad baja (20-50% con 1 RM), en combinación con la restricción del flujo sanguíneo en la parte proximal de la articulación que se quiere entrenar, ha demostrado ser capaz de inducir también hipertrofia muscular. Este hecho se ha demostrado en poblaciones de individuos sanos sedentarios, físicamente activos y deportistas, así como en ancianos y pacientes en período de rehabilitación. El entrenamiento en isquemia es un método novedoso y alternativo para poblaciones que no sean capaces de movilizar cargas de alta intensidad. Aún no se han esclarecido los mecanismos a través de los cuales este tipo de entrenamiento es capaz de inducir hipertrofia muscular, aunque la acumulación metabólica inducida por la restricción del aclarado venoso y el incremento de la tasa de síntesis proteica parecen ser los mecanismos más probables (Martín-Hernández, 2011).

La Visión artificial es un campo de la Inteligencia Artificial que, mediante el uso de diversas técnicas, permite la obtención, procesamiento y análisis de diversos tipos de información a través de imágenes digitales. El objetivo del proyecto de tesis “Sistema de Visión Artificial para la Detección y Corrección de Posturas en Ejercicios realizados por Fisicoculturistas” (Martínez José, 2018) es desarrollar un sistema de Visión Artificial interactivo, que permita reconocer y seguir los movimientos de los fisicoculturistas para determinar si sus posturas son correctas en tiempo real, mediante la posición de sus articulaciones en el sistema de ejes de tres dimensiones. Las pruebas se realizaron con distintas condiciones ambientales con especial énfasis en la iluminación, además de la ropa que se utilizó, distancias y altura en la posición del sensor Kinect. Teniendo la información anterior en cuenta se pudo observar que los resultados variaron acorde al tipo de iluminación del entorno, adaptándose de una forma satisfactoria al ambiente que cuenta tanto con iluminación natural como artificial (Martínez José, 2018).

Después de analizar exhaustivamente la literatura existente, no se encontraron evidencias del uso de aprendizaje por refuerzo para el estudio y simulación relacionados no solo con la hipertrofia o fuerza, sino de manera general para determinar la forma óptima de entrenamiento para un individuo que busca estas características. Por lo tanto, en este estudio se explorará cómo utilizar algoritmos de aprendizaje automático como Q-learning y Deep Q-learning, para simular y analizar situaciones con el fin de obtener mejores resultados en el ámbito de la cultura física. Este estudio busca demostrar cómo estas herramientas podrían mejorar significativamente los estudios y prácticas en esta área, ofreciendo nuevos enfoques y metodologías para optimizar los programas de entrenamiento.

# 4 Propuesta de solución

Para abordar el problema de optimizar la rutina de ejercicios para la ganancia muscular en los brazos, proponemos la implementación de algoritmos de aprendizaje por refuerzo, específicamente Q-Learning y Deep Q-Learning. El objetivo es que un agente aprenda la política óptima de ejercicios que maximicen el crecimiento muscular, medido en términos de aumento del tamaño del brazo.

## 4.1 Definición de Estados y Acciones

El sistema considera un total de 16 estados discretos que representan diferentes tamaños de brazo, comenzando desde 35.6 cm hasta 38.6 cm, con incrementos de 0.2 cm. Estos estados son:

Estado 1: 35.6 cm
Estado 2: 35.8 cm
...
Estado 16: 38.6 cm

Las acciones disponibles para el agente son cuatro ejercicios, dos para bíceps y dos para tríceps:

Curl predicador (bíceps)
Curl martillo (bíceps)
Extensión polea con lazo (tríceps)
Incline DB powerbombs (tríceps)

## 4.2 Recompensas

Basándonos en un conjunto de datos recuperado de la plataforma Kaggle (Kaggle, 2023), la cual ofrece una amplia variedad de conjuntos de datos de diferentes áreas para su análisis,

hemos seleccionado un conjunto de datos que contiene información sobre ejercicios para distintos grupos musculares, incluyendo bíceps y tríceps. Cada ejercicio en este conjunto de datos está acompañado de una calificación (*rating*), que se utilizará para definir la recompensa asociada a cada acción (*ejercicio*) que el agente realice. Esta calificación servirá como una métrica para evaluar la efectividad de cada ejercicio, permitiendo al agente ajustar su política de selección de ejercicios de manera óptima, toda esta información la podemos ver reflejada en la Tabla 1.

Tabla 1. Ejercicios y sus respectivas recompensas

| Número | Ejercicio | Recompensa |
|---|---|---|
| 1 | Curl predicador (bíceps) | Recompensa de 8.8 |
| 2 | Curl martillo (bíceps) | Recompensa de 9.1 |
| 3 | Extensión polea con lazo (tríceps) | Recompensa de 8.7 |
| 4 | Incline DB powerbombs (tríceps) | Recompensa de 8.9 |

El objetivo final es que el agente aprenda la política óptima, que en este contexto es la mejor secuencia de ejercicios para maximizar el crecimiento muscular hasta alcanzar el estado deseado, que es el estado 16 (brazo de 38.6 cm).

### 4.3 Algoritmo de Q-Learning

En el algoritmo que apreciamos en el Pseudocodigo 1 de Q-Learning, el agente explora el espacio de estados y acciones para aprender una función de valor *Q (s, a)*, que estima la recompensa esperada de realizar una acción *a* en un estado *s* y siguiendo la política óptima a partir de entonces. El proceso de actualización de Q-Learning es en función de la Ecuación 1. En el Pseudocódigo 1 se presenta el algoritmo de Q-Learning.

**Pseudocodigo 1.** Algoritmo de Q-learning

```
# Definir parámetros del Q-Learning
num_estados = 16
num_acciones = 4
num_episodios = 100
epsilon = 0.1  # Probabilidad de exploración
alpha = 0.1  # Tasa de aprendizaje
gamma = 0.9  # Factor de descuento
meta_estado = 15
meta_recompensa = 10

# Inicializar la matriz Q con valores arbitrarios
Q = matriz de ceros de tamaño (num_estados, num_acciones)

# Definir las recompensas para cada acción
recompensas = [8.8, 9.1, 8.7, 8.9]

# Definir la función de selección de acción
función seleccionar_accion(estado):
```

```
    si número aleatorio entre 0 y 1 < epsilon:
        retornar una acción aleatoria entre 0 y num_acciones - 1
    de lo contrario:
        retornar la acción con el valor máximo en Q[estado]

# Definir la función de actualización de Q
función actualizar_Q(estado, accion, siguiente_estado, recompensa):
    mejor_siguiente_accion = acción con el valor máximo en Q[siguiente_estado]
    Q[estado, accion] = Q[estado, accion] + alpha * (recompensa + gamma * Q[siguiente_estado, mejor_siguiente_accion]    -
Q[estado, accion])

# Ejecutar el algoritmo de Q-Learning
para episodio en rango(num_episodios):
    estado_actual = estado aleatorio entre 0 y num_estados - 1
    mientras estado_actual != meta_estado:
        accion = seleccionar_accion(estado_actual)
        siguiente_estado = estado_actual + 1
        siguiente_estado = mínimo(siguiente_estado, meta_estado)
        recompensa = recompensas[accion] si siguiente_estado != meta_estado, de lo contrario meta_recompensa
        actualizar_Q(estado_actual, accion, siguiente_estado, recompensa)
        estado_actual = siguiente_estado

# Imprimir la matriz Q
imprimir "Matriz Q:"
```

### 4.4 Algoritmo de Deep Q-Learning

Para mejorar la capacidad de generalización y manejar la dimensionalidad más alta, se implementa el algoritmo Deep Q-Learning. En lugar de una tabla de valores *Q*, se utiliza una red neuronal profunda para aproximar la Función *Q*. La estructura de la red incluye una capa de entrada que toma el estado actual, una serie de capas ocultas y una capa de salida que proporciona un valor *Q* para cada acción posible.

## 5 Resultados experimentales

Para evaluar la efectividad de los algoritmos, se realizaron simulaciones donde el agente repetidamente hace la selección de ejercicios, mejorando su política a través de la experiencia.

En relación con el algoritmo de Q-Learning, la Figura 1 muestra una matriz resultante de tamaño estados por acciones, es decir, de 16 por 4. Inicialmente, esta matriz se inicializa con ceros y se actualiza utilizando la ecuación de Q-Learning (*ver Ecuación 1*). Con el tiempo y a medida que transcurren más episodios, el agente va ajustando y mejorando sus valores *Q*. En la Figura 1 se presentan los resultados obtenidos mediante este algoritmo, limitado a 10 episodios.

Es evidente que el tiempo limitado no fue suficiente para que el agente explorara todas las opciones disponibles, lo que se refleja en algunos valores de la tabla que permanecen en cero. Esto indica que el agente no pudo experimentar todas las combinaciones posibles de estados y acciones, lo que limita su capacidad para tomar decisiones óptimas.

Para una mejor evaluación del desempeño del algoritmo Q-Learning, sería necesario aumentar el número de episodios, que permitan al agente explorar el espacio de estados y acciones de manera más exhaustiva.

```
Matriz Q:
[[0.         0.         0.         0.        ]
 [0.88       0.         0.         0.        ]
 [1.04466472 0.         0.         0.        ]
 [3.60092313 0.         0.         0.        ]
 [3.54502755 1.14523192 0.         0.        ]
 [4.33236698 0.         0.         0.        ]
 [5.38615826 0.         0.         0.        ]
 [5.64079044 0.         0.         0.        ]
 [4.60420733 1.2220796  1.10523192 0.        ]
 [5.93341732 0.         0.         0.        ]
 [5.43112451 1.37383514 0.         0.        ]
 [6.80670189 0.         0.         0.        ]
 [6.00858292 0.         0.         1.35843635]
 [6.03389626 1.3317031  0.         0.        ]
 [5.6953279  0.         1.         0.        ]
 [0.         0.         0.         0.        ]]
```

Figura 1. Resultados del algoritmo de Q-learning

Al incrementar el número de episodios, por ejemplo, a 100 como se muestra en la Figura 2, podemos apreciar una exploración más completa de los valores en nuestra tabla Q. Al analizar esta tabla, observamos que cuando nos encontramos en el estado número 1 (la primera fila), la mejor acción para realizar es la acción 1, con un puntaje de 6.85, mientras que las demás acciones tienen un puntaje de 0. Esto nos sugiere que la mejor estrategia a seguir cuando el brazo tiene una longitud de 35.6 cm es realizar el ejercicio de Curl predicador.

Este resultado resalta la importancia del número de episodios en el proceso de aprendizaje del agente. A través de un mayor número de episodios, el agente tiene la oportunidad de explorar más exhaustivamente el espacio de estados y acciones, lo que conduce a una mejor comprensión y selección de las acciones óptimas en diferentes situaciones.

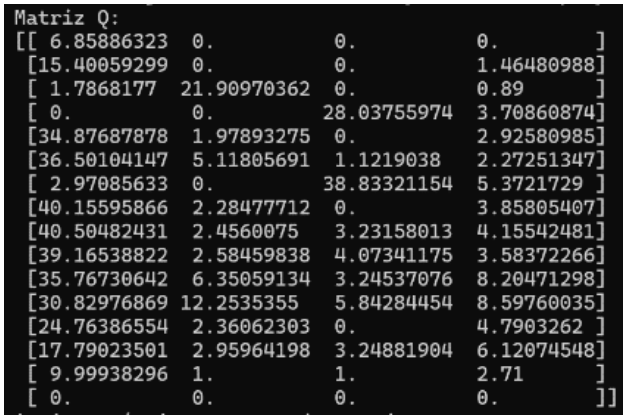


Figura 2. Tabla de Q-learning

En cuanto al algoritmo de Deep Q-Learning, se puede observar en la Figura 3 que, tras el entrenamiento de nuestra red neuronal, al consultarle cuál sería la mejor acción para un determinado estado, como en el ejemplo del estado 14, se determina que la mejor acción a tomar es la acción número 3, que corresponde a la extensión de la polea con lazo.

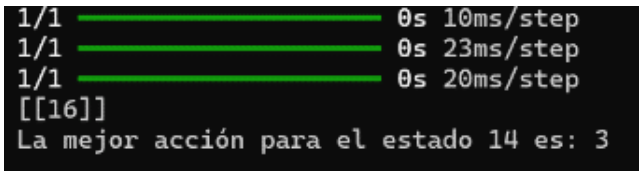


Figura 3. Resultado de Deep Q-learning

En comparación con el algoritmo de Q-Learning, el algoritmo de Deep Q-Learning nos proporciona un resultado distinto. Esto se debe al número de episodios utilizados en el algoritmo de Q-Learning, pero ambos algoritmos son igual de buenos para este escenario.

## 6 Conclusiones y trabajo a futuro

En conclusión, el presente estudio ha demostrado cómo la inteligencia artificial, y en particular los algoritmos de aprendizaje por refuerzo como Q-learning y Deep Q-learning, pueden ser aplicados para simular y optimizar los programas de entrenamiento físico con el objetivo de maximizar la hipertrofia muscular. Mediante la implementación de un modelo en el cual un agente progresa desde un estado inicial hasta uno más desarrollado de la musculatura del bíceps, se ha evidenciado la capacidad de estos algoritmos para personalizar y mejorar los programas de ejercicio de manera eficiente y efectiva.

Aunque el ejemplo utilizado en este estudio es relativamente simple y se enfoca únicamente en el bíceps y tríceps, las técnicas presentadas tienen un amplio potencial de aplicación en el ámbito de la cultura física. Estas herramientas de aprendizaje por refuerzo

pueden ser adaptadas para diferentes músculos o grupos musculares y diferentes áreas de las ciencias de la cultura física, permitiendo la simulación y análisis de una variedad de escenarios de entrenamiento. Esta capacidad de exploración y optimización representa una ventaja significativa en la personalización de rutinas de ejercicio, superando las limitaciones de la subjetividad y la variabilidad individual que caracterizan los métodos tradicionales.

Como trabajo a futuro, se planea mejorar el modelo mediante la incorporación de una mayor cantidad de parámetros que incrementen la precisión en la medición de la masa muscular. Entre estos parámetros, se incluirá el índice de masa corporal (IMC), el cual permitirá una evaluación más integral y personalizada de los individuos. Además, se considera la integración de medidas adicionales como la circunferencia de la cintura y la cadera, el porcentaje de grasa corporal y el nivel de actividad física. Estos parámetros complementarios no solo contribuirán a una estimación más exacta de la masa muscular, sino que también proporcionarán un panorama más completo del estado de salud general del sujeto.

## Referencias


R. Timón, B. Bugarín, I. Martínez-Guardado, M. Marcos, M. Camacho-Cardeñosa, G. Olcina. (2017). "Efectos de la frecuencia semanal de entrenamiento sobre la hipertrofia y la fuerza muscular en individuos entrenados", *Rev Andal Med Deporte,* vol.10 no.4 Sevilla dic. 2017.

Diego Fernández-Lázaro, Joseba Díaz, Alberto Caballero, Alfredo Córdova4. (2019). "Entrenamiento de fuerza y resistencia en hipoxia: efecto en la hipertrofia muscular", *Biomédica,* vol.39 no.1 Bogotá Jan/Mar: 2019.

Martín-Hernández, J.; Marín, P.J.; Herrero, A.J. (2011). "Revisión de los procesos de hipertrofia muscular inducida por el entrenamiento de fuerza oclusivo", *Revista Andaluza de Medicina del Deporte*, vol. 4, núm. 4, diciembre, 2011, pp. 152-157.

Christian Thibaudeau. (2007). " El Libro Negro de Los Secretos de Entrenamiento" Editorial F.Lepine ISBN 978-0-9783194-5-8

Christopher j.c.h. Watkins. (1992). "Q-Learning", *Kluwer Academic Publishers*, Boston. Manufactured in The Netherlands, Machine Learning, 8, 279-292.

Ankit Choudhary. (2023). *A Hands-On Introduction to Deep Q-Learning using OpenAI Gym in Python*. Recuperado de https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/

Martínez José, Juan Manuel. (2018). "Sistema de Visión Artificial para la Detección y Corrección de Posturas en Ejercicios realizados por Fisicoculturistas", Recuperado de http://ri.uaemex.mx/handle/20.500.11799/95190

Abid Ali Awan. (2024). "Introducción al Q-Learning: Tutorial para principiantes" Recuperado de https://www.datacamp.com/es/tutorial/introduction-q-learning-beginner-tutorial

Ambarish deb. Gym Exercises Dataset. (2023). Citibike Data Analysis. Kaggle. https://www.kaggle.com/datasets/ambarishdeb/gym-exercises-dataset/data

Miguel Alarcón-Rivera, Luis Benavides-Roca, Cristian Salazar Orellana, Eduardo Guzmán-Muñoz, (2024). "Efectos del entrenamiento cluster sobre la hipertrofia muscular: Una revisión sistemática", Revista en ciencias del Movimiento Humano y Salud, Vol. 21(1), e-ISSN: 1659-097X

# Capítulo 10

## Modelado de flujo de datos en la red Wi-Fi pública de la ciudad de Puebla

Mario Sánchez González[1]

[1] Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación

sg223470505@alm.buap.mx

**Resumen.** El constante flujo de datos y tamaño de una red permite entender que tan confiable es el servicio que esta ofrece a una comunidad, en especial si es una red pública. Este artículo explora la complejidad de la red de Wi-Fi pública en la ciudad de Puebla de Zaragoza, así como el tráfico dentro de la misma en un periodo de tiempo. Usando algoritmos de aprendizaje supervisado y no supervisado se analizará el comportamiento de los dispositivos en la red y la aglomeración de los puntos de acceso que conforman la red.

**Palabras Clave:** Modelado, *DBSCAN*, *K-Means*, *K-NN*, *Random Forest*, Regresión Lineal.

## 1 Introducción

El internet forma parte de la cotidianidad de gran parte del mundo hoy en día, la capacidad de comunicarnos casi al instante, compartir datos alrededor y tener acceso a diversas fuentes de información han ampliado por completo la forma en que el mundo se estructura hoy en día.

En la ciudad de Puebla de Zaragoza existe un programa gubernamental con el objetivo de crear puntos Wi-Fi de acceso público para la población en general (Puebla Capital, 2024a), este programa ha mantenido registros generales de dispositivos conectados a estos puntos de acceso y el número de conexiones realizadas por estos; además de los datos de ubicación de cada uno de los puntos de acceso en la ciudad, tales como coordenadas geográficas, calles y colonia.

No obstante, algunos datos de conexión se han perdido o no fueron registrados en algunos periodos de tiempo, como es el caso del año 2020 y gran parte del año 2022, por lo que será necesario generar una aproximación de los datos de dichos periodos; además, es necesario identificar las zonas de la ciudad donde ha existido una mayor demanda para la

implementación de estos puntos Wi-Fi, con lo cual se podrá ver cómo ha sido el crecimiento de la red y el uso de la misma, lo que nos dará una perspectiva de la fiabilidad del programa y su continuación en años posteriores.

Tomando esto en cuenta, en este documento se hará un modelado de los datos de la red de internet pública de la ciudad de Puebla de Zaragoza en el estado de Puebla, con el objetivo de entender la magnitud y complejidad de la red misma en tiempos recientes, así como el uso que se le ha dado a esta en un periodo de tiempo que va desde finales del 2019 a finales del 2023.

El siguiente artículo se estructura de la siguiente manera; en la sección 2 exponemos trabajos relacionados con el campo de investigación en el que se enfoca este artículo; en la sección 3 se plasmarán todo lo relacionado a los conceptos que se tocan en este trabajo; la sección 4 plasmará la propuesta de solución que se ha implementado en este artículo; mientras que en la sección 5 se expondrán los resultados preliminares obtenidos por la propuesta de solución; en la sección 6 se darán a conocer las conclusiones relacionadas a lo observado en este trabajo y los comentarios dirigidos a posibles trabajos futuros o aplicaciones relacionadas a este artículo.

## 2 Estado del Arte

El estudio de las redes de internet, así como sus características y desempeño no son nada recientes, el análisis de esta tecnología ha sido objeto de estudio en muchas áreas de la computación, esto con el objetivo de desarrollar nuevas formas de conexión y comunicación que permitan un enlace más estable y fluido entre dispositivos.

En años recientes se han estudiado aspectos como el consumo de energía por parte de este tipo de redes, en ese trabajo realizado por Stepanova et al. (2022) se enfocó en presentar propuestas que permitan ahorrar en el consumo de energía, en este caso se estudian dos propuestas, el TWT (*Target Wake Time*) y el WUR (*Wake-Up Radio*); en el TWT se programan momentos específicos para que el dispositivo se active y escuche el canal, reduciendo el tiempo en que la radio está encendida, mientras tanto el WUR utiliza una radio de baja potencia para escuchar señales de activación, permitiendo que la radio principal se mantenga apagada la mayor parte del tiempo. El objeto de estudio en esas dos propuestas es un problema llamado efecto de la deriva del reloj, que es una desviación aleatoria en el reloj del dispositivo, puede afectar la eficiencia de TWT y WUR. Por este hecho se hace un estudio al comportamiento y desempeño de estos conceptos evaluando estos mecanismos de ahorro de energía usando la plataforma de simulación ns-3, lo que proporciona una base sólida para comparar estos métodos en condiciones controladas.

El ámbito de la seguridad en las redes Wi-Fi también ha sido tema de estudio desde el inicio de estas, el cifrado de la información ha sido una pieza clave para ello y entre tantos estudios la distribución de claves cuánticas ha sido parte de ellos; durante ese estudio

realizado por Escanez-Exposito et al. (2023) se plasma la comparación de los métodos tradicionales de cifrado con la QKD(siglas en inglés para distribución de claves cuánticas), la implementación entre dos protocolos QDK, los protocolos de cifrado E91 y B92. Además, el estudio realiza un análisis preliminar del potencial de aplicar estos protocolos QKD de manera híbrida en redes Wi-Fi, basadas en el estándar IEEE 802.11. Este análisis es relevante para comprender las posibilidades de aumentar la seguridad en redes inalámbricas utilizando tecnologías cuánticas.

Las aplicaciones del internet han progresado considerablemente en la última década, la interconexión de dispositivos inteligentes ha innovado en el ámbito de la tecnología actualmente, entre estos avances está la conectividad Wi-Fi en vuelo (IFC por sus siglas en inglés), lo cual se ha abordado en un estudio elaborado por Gurumekala y Gandhi (2022) para solucionar un problema relacionado con los protocolos de enrutamiento existentes, esto debido a que generan sobrecarga de paquetes y retraso debido al método de *beaconing* tradicional. En ese trabajo se hace una propuesta de solución vinculada a redes Ad hoc aeronáuticas (AANET en inglés): Un nuevo tipo de redes móviles Ad hoc diseñadas para ofrecer IFC sobre áreas remotas u oceánicas. Para ello se propuso modelar la movilidad de los aviones utilizando los estándares de separación de la Organización de Aviación Civil Internacional para evitar colisiones.

Además, los puntos de acceso Wi-Fi públicos también han sido estudiados con un enfoque en la generación de ingresos a través de la difusión de anuncios, en ese estudio obra de Xu et al. (2021) se hace un análisis hacia la tolerancia de los usuarios hacia los anuncios influye en su efectividad y en las ganancias a largo plazo.

Las ciudades inteligentes, siendo parte de las maravillas de la nueva década, demuestran ser el foco de atención en el ámbito de la investigación, este estudio elaborado por Gandhi y Narmawala (2023) se enfoca en el uso de datos de sensores en tiempo real para optimizar diversos servicios urbanos y mejorar la calidad de vida de sus habitantes; sin embargo, el rápido aumento de datos de sensores plantea importantes retos en estas áreas, especialmente en ciudades densamente pobladas.

Con los artículos observados anteriormente se ha obtenido una perspectiva de los diferentes aspectos del internet que han desglosado una vasta colección de investigaciones con el propósito de mejorar esta tecnología. Por lo tanto, la contribución de este trabajo está enfocado a la implementación de conocimientos relacionados con el aprendizaje supervisado y no supervisado para el modelado del desempeño de una red Wi-Fi pública y la magnitud del crecimiento en esta, ya que estos aspectos pueden contribuir a tener un panorama más claro sobre el desarrollo de más proyectos como este en otras ciudades o la posible descontinuación de estas ideas.

# 3 Conceptos básicos

En esta sección se expondrán las definiciones correspondientes a los métodos y términos utilizados en la elaboración de este trabajo, en este caso temas relacionados con aprendizaje supervisado, tales como regresión lineal (Bishop, 2006), *random forest* (Hastie et al., 2013), *K-NN* (Hastie et al., 2013); y aprendizaje no supervisado, como son *K-Means* (Bishop, 2006) y *DBSCAN* (Witten et al., 2016).

## 3.1 Algoritmos de aprendizaje supervisado

Un algoritmo de aprendizaje supervisado es un tipo de algoritmo de aprendizaje automático que se entrena utilizando un conjunto de datos etiquetados. En este contexto, "etiquetado" significa que cada ejemplo de entrenamiento en el conjunto de datos tiene una entrada y una salida deseada conocida (también llamada etiqueta). El objetivo del algoritmo es aprender una función que mapea las entradas a las salidas deseadas de tal manera que pueda predecir correctamente la salida para nuevas entradas no vistas.

### 3.1.1 Regresión Lineal

La regresión lineal es una técnica de análisis de datos que predice el valor de datos desconocidos mediante el uso de otro valor de datos relacionado y conocido. Modela matemáticamente la variable desconocida o dependiente y la variable conocida o independiente como una ecuación lineal. Este método se representa con la fórmula mostrada en la Ecuación 1.

$$y = mx + b \tag{1}$$

Donde:

$y$ es la variable dependiente que estamos tratando de predecir.

$x$ es la variable independiente que estamos utilizando para hacer la predicción.

$m$ es la pendiente de la línea, que representa el cambio en $y$ por cada cambio unitario en $x$.

$b$ es la intersección en $y$ cuando $x$ es igual a cero.

### 3.1.2 *Random Forest*

Los bosques aleatorios (*Random Forest* en inglés) es un algoritmo de aprendizaje supervisado que se utiliza tanto para clasificación como para regresión. Este método combina múltiples árboles de decisión para mejorar la precisión y evitar el sobreajuste.

Utiliza el método de *bagging* para crear cada árbol de decisión en el bosque. Esto implica tomar múltiples muestras aleatorias con reemplazo del conjunto de datos original y entrenar un árbol de decisión en cada una de estas muestras. Para la regresión, el resultado final es el promedio de las predicciones de todos los árboles. Mientras que, para la clasificación, el resultado final es la clase más votada por los árboles.

### *3.1.3 K-NN*

*K-NN* (*K-Nearest Neighbors* en inglés) es un algoritmo de aprendizaje supervisado utilizado tanto para clasificación como para regresión. Es uno de los algoritmos más simples y efectivos, basado en la idea de que objetos similares estarán cerca en el espacio de características.

*K-NN* clasifica un punto de datos en función de los $K$ puntos de datos más cercanos en el espacio de características. Para regresión, el valor predicho es el promedio de los valores de estos $K$ vecinos más cercanos.

## 3.2 Algoritmos de aprendizaje no supervisado

Un algoritmo de aprendizaje no supervisado es un tipo de algoritmo de aprendizaje automático que se utiliza para extraer patrones o estructuras a partir de datos que no tienen etiquetas ni salidas predefinidas. A diferencia del aprendizaje supervisado, donde el modelo se entrena con ejemplos de entrada-salida, en el aprendizaje no supervisado el objetivo es encontrar estructuras ocultas en los datos sin guía explícita sobre qué resultado se espera.

### *3.2.1 K-Means*

*K-Means* es un algoritmo de *clustering* no supervisado que agrupa un conjunto de datos en $K$ *clusters* o grupos. El objetivo es minimizar la variación dentro de cada *cluster* y maximizar la variación entre *clusters*.

*K-Means* selecciona $K$ centroides iniciales, que pueden ser puntos seleccionados aleatoriamente o utilizando algún método como *K-Means++*. Cada punto de datos se asigna al *cluster* cuyo centroide esté más cercano, según una medida de distancia, típicamente la

distancia euclidiana. Luego los centroides de los *clusters* se recalculan como el promedio de todos los puntos asignados a cada *cluster*. Esto se repite hasta que los centroides ya no cambian significativamente o se alcanza un número máximo de iteraciones.

### *3.2.2 DBSCAN*

*DBSCAN* (*Density-Based Spatial Clustering of Applications with Noise* en inglés) es un algoritmo de *clustering* no supervisado que agrupa puntos de datos que están densamente empaquetados juntos, marcando puntos que están en áreas de baja densidad como ruido.

*DBSCAN* agrupa puntos en *clusters* en función de la densidad de los puntos, los *clusters* se forman donde hay una alta densidad de puntos. Los parámetros principales del algoritmo son $\epsilon$ , que se define como el radio de vecindad de un punto, y *MinPts* significando el número mínimo de puntos necesarios para formar un *cluster*.

## 4 Propuesta de solución

Los datos perdidos correspondientes al promedio de conexiones de dispositivos en los puntos de acceso Wi-Fi pueden ser aproximados a través de algoritmos de aprendizaje supervisado, por lo que se implementarán tres algoritmos distintos, Regresión Lineal, *Random Forest*, *K-NN*; con los cuales se podrá comparar . Mientras tanto, los datos de ubicación de los puntos pueden analizarse con aprendizaje no supervisado, con esto en cuenta se aplicarán los algoritmos *K-Means* y *DBSCAN* para comparar los resultados de estos métodos y tener una perspectiva precisa.

El primer paso en esta propuesta es la inserción de los datos, esta información es ingresada a los métodos usando la biblioteca Pandas de Python; luego se procederá a hacer el preprocesamiento de los datos, donde se identificarán los valores que se desean implementar en el modelo de aprendizaje, se convertirán en el formato necesario para su procesamiento y normalizarán si el método así lo necesita.

Después se hará el entrenamiento del modelo de aprendizaje, ya sea supervisado o no supervisado, que se encargará de procesar la información correspondiente y devolver los resultados según el tipo de modelo, los cuales pueden implementarse usando la biblioteca Scikit-learn de Python, en el caso de los modelos de aprendizaje supervisado se obtendrán las predicciones de los datos procesados y en modelos de aprendizaje no supervisado se mostrarán los agrupamientos de los datos introducidos.

Por último, se imprimirán y guardarán los resultados generados por los modelos correspondientes, para aprendizaje supervisado se exportarán las predicciones de los modelos en archivos de formato .csv para su comparación; y para el aprendizaje no supervisado se plasmará el agrupamiento de los datos de forma geográfica utilizando la

biblioteca Folium, con el propósito de tener una representación fidedigna y clara del agrupamiento de los datos.

# 5 Resultados

Los modelos implementados en la sección 4 han devuelto datos interesantes con respecto a los datos que se plasmarán a continuación, mostrando en el proceso agrupamientos y predicciones que permiten tener una perspectiva más clara de las características analizadas en este trabajo.

## 5.1 Conjunto de datos utilizados

La red Wi-Fi pública de la ciudad de Puebla tiene dos tipos de registros, la ubicación de todos los puntos de acceso repartidos por toda la ciudad (Puebla Capital, 2024c) correspondientes al año 2024 y conformados por un total de 600 puntos de acceso registrados, los cuales se conforman por las siguientes características: nombre del sitio, dirección, latitud, longitud y la colonia donde se ubican; en la Tabla 1 se ve reflejado una muestra de estos datos.

Tabla 1. Muestra de Puntos Wifi tabla mar 2024 (Puebla Capital, 2024c)

| SITIO | Dirección | Latitud | Longitud | Colonia |
|---|---|---|---|---|
| 16 DE SEPTIEMBRE | 74 PTE 5 NORTE | 19.073494 | -98.185175 | 16 DE SEPTIEMBRE NORTE |
| 3 DE MAYO 2001 | AYUNTAMIENTO 10 DE MAYO | 19.086469 | -98.158606 | TRES DE MAYO |
| 3 DE MAYO 2002 | Y 12 DE DICIEMBRE 5 DE ABRIL | 19.08765 | -98.162222 | TRES DE MAYO |
| AGUA AZUL | ENTRE 53 PONIENTE Y PASEO DEL RIO 11 SUR | 19.0257371 | -98.222241 | AGUA AZUL |
| ALTOS DE JALISCO | CALLE ALTOS DE JALISCO ENTRE BACHILLERES | 19.0974678 | -98.198826 | SAN PABLO XOCHIMEHUACAN |

Mientras tanto, el número de dispositivos y conexiones realizadas por estos en toda la red (Puebla Capital, 2024b) son registrados de forma general con las siguientes características: mes del registro, la cantidad de dispositivos conectados a la red y la cantidad de conexiones totales que hubo en la red, es decir, un mismo dispositivo puede conectarse; en la Tabla 2 se ve reflejado una muestra de estos datos.

Tabla 2. Muestra de Promedio_conexiones_2019_2023 (Puebla Capital, 2024b)

| MES | DISPOSITIVOS | CONEXIONES |
|---|---|---|

| | | |
|---|---|---|
| oct-19 | 1,022,277 | 2,528,890 |
| nov-19 | 2,125,517 | 5,053,833 |
| dic-19 | 2,321,599 | 5,279,485 |
| ene-21 | 1,600,921 | 5,837,827 |
| feb-21 | 1,565,848 | 5,585,885 |

En los datos de conexiones de la red Wi-FI pública de la ciudad de Puebla existe un vacío en los registros que datan del periodo de enero de 2020 a diciembre de 2020 y de mayo de 2022 a diciembre de 2022, lo cual hace de estos periodos un objetivo perfecto para su predicción para identificar el desempeño de la red en años más recientes; mientras que los datos de ubicación de los puntos de acceso pueden visualizar las zonas con una gran acumulación de puntos de acceso y ver la dimensión de la red.

### 5.2 Resultados experimentales usando aprendizaje supervisado

Con el uso de los algoritmos de regresión lineal, *Random Forest* y *K-NN* se pueden obtener diferentes predicciones de los valores faltantes en el *dataset*, ya que si se grafican los datos en su estado actual se pueden ver dos vacíos en la progresión de tiempo de estos, como se puede apreciar en la Figura 1.

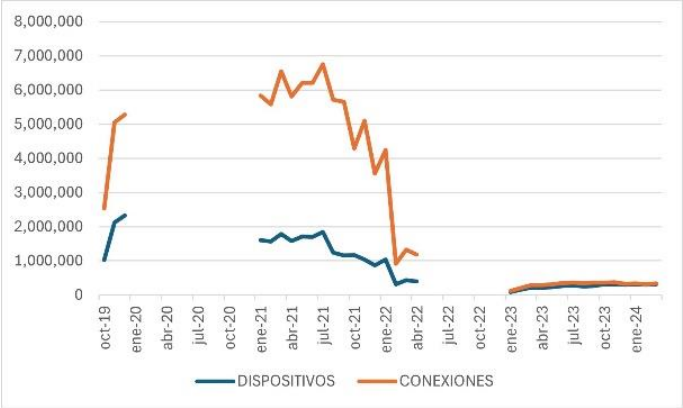


Figura 1. Gráfica de los datos del dataset inicial.

Con la implementación de *K-NN* se puede apreciar como el modelo de entrenamiento da resultados interesantes; como se aprecia en la Figura 2, el modelo de *K-NN* indica que, a partir de enero del 2020 a mayo del 2020, el número de conexiones realizadas en la red se mantienen constantes, tienen un aumento durante junio y julio para volver a mantenerse constantes el resto del 2020, mientras que en el periodo faltante de mayo a diciembre de 2020 se predicen valores similares que varían por muy poco. Mientras que en los números

de dispositivos conectados el comportamiento de los valores predichos en esos periodos es similar, con la diferencia que en el periodo de junio y julio de 2020 los valores decrecen.


Figura 2. Gráfica de datos predichos por *K-NN*.

En caso de *Random Forest* tenemos una predicción que se asemeja a lo visto en *K-NN*, con la diferencia que los valores relacionados a conexiones en la red no son tan bajos como en comparación a los resultados de *K-NN*, además que en el periodo de mayo de 2022 se nota un crecimiento en ese mismo periodo comparado con el método anteriormente explicado, algo que se puede observar en la Figura 3.


Figura 3. Gráfica de datos predichos por *Random Forest*.

Con la regresión lineal se observan resultados completamente diferentes a lo que se muestran en los otros dos métodos implementados, ya que se puede apreciar cómo decrecen los datos de conexiones en la red que inicia desde inicios del 2020 hasta finales del mismo año, tomando en cuenta que en mayo de 2022 se observa un aumento abrupto que va disminuyendo hasta ver una baja drástica al empezar el año 2023; de la misma manera se ve como los dispositivos conectados a la red decrecen de forma similar a las conexiones, algo que se plasma en la Figura 4.


Figura 4. Gráfica de datos predichos por Regresión Lineal.

### 5.3 Resultados experimentales usando aprendizaje no supervisado

Usando los algoritmos de aprendizaje no supervisado han retornado resultados interesantes respecto al agrupamiento de los puntos de acceso en la red, ya que aplicando *K-Means* y *DBSCAN* es posible encontrar patrones interesantes en los puntos plasmados geográficamente de la Figura 5.

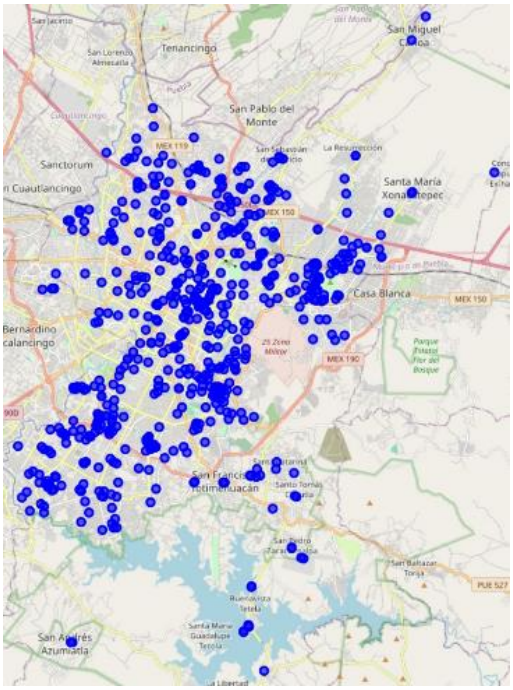


Figura 5. Puntos de acceso Wi-Fi plasmado en un mapa.

El modelo de aprendizaje implementado por *K-Means* ha dado como resultado un agrupamiento de 5 grupos a partir de los datos procesados; esto puede visualizarse en la Figura 6, los cuales representan una acumulación de puntos de acceso en las zonas centro, noroeste y sudoeste de la ciudad, mientras que en la parte noreste y sudeste se distinguen grupos con una menor acumulación de puntos de acceso en comparación con los otros grupos obtenidos.

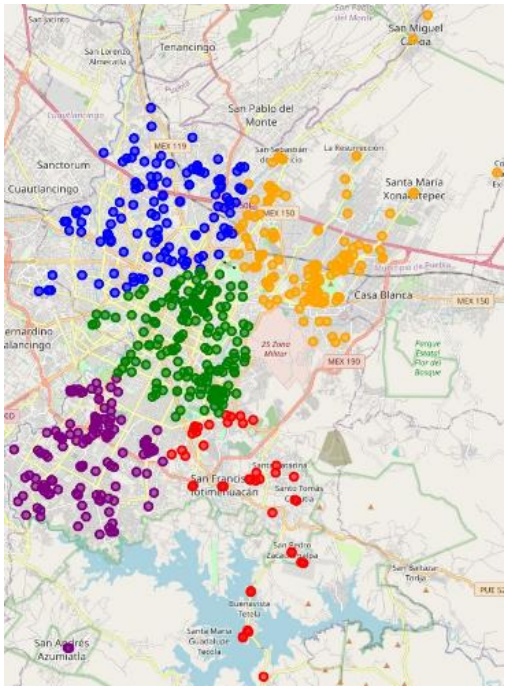


Figura 6. Grupos generados por *K-Means* en el mapa.

Por parte de *DBSCAN* se obtuvieron distintos grupos donde se pueden notar grupos donde la aglomeración es notable de forma más precisa en ciertas zonas de la ciudad, tales como el centro histórico, parques, escuelas, zonas turísticas y zonas habitacionales, esto se ve reflejado en la imagen 7.

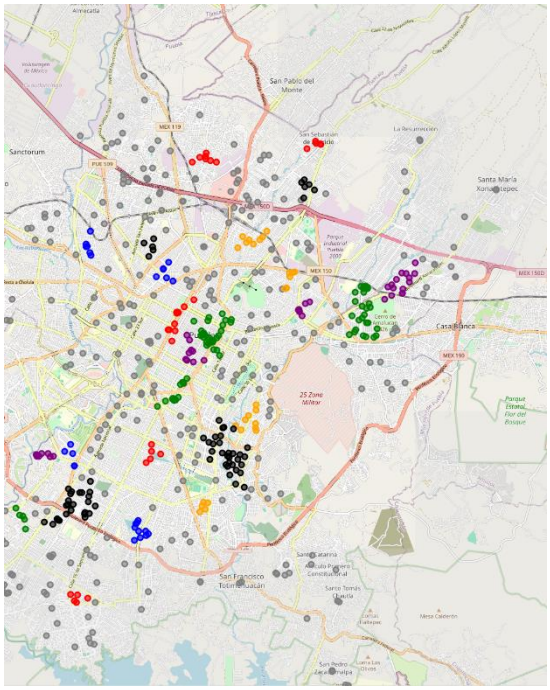


Figura 7. Grupos generados por *DBSCAN* en el mapa.

## 6 Conclusiones

Para concluir, es importante recalcar los aspectos identificables derivados de los resultados obtenidos en este estudio, empezando por el aprendizaje no supervisado. Los datos observados en la sección 5.3 han sido muy interesantes, han expuesto el gran tamaño que el programa de red Wi-Fi pública ha desarrollado hasta la actualidad, así como algunas zonas con un enfoque mayor en la instalación de más puntos de acceso para dichas ubicaciones, lo cual abre la cuestión de investigar si es posible expandir aún más la red hacia más zonas de la ciudad.

En el caso de los resultados presentados por los modelos de aprendizaje supervisado en la sección 5.2 han demostrado como de los tres algoritmos implementados solo uno mantiene un comportamiento completamente diferente a los demás. La regresión lineal devuelve resultados que simulan una decadencia tanto en los dispositivos conectados y las conexiones realizados por estos en los periodos calculados, mientras que *K-NN* y *Random*

*Forest* muestran predicciones muy similares con un rango de diferencia entre los valores generados.

No obstante, el desempeño predicho por *K-NN* y *Random Forest* no concuerda demasiado con el contexto histórico en el que estos registros se desarrollan, la cuarentena iniciada a principios del año 2020 y terminada hasta finales del año 2022, donde las personas se vieron confinadas en sus hogares y aisladas del exterior. Por lo tanto, el comportamiento predicho en la red Wi-Fi por estos algoritmos resulta improbable con este contexto, lo que hace que los resultados proporcionados por el algoritmo de regresión lineal sean más viables.

Se espera que en un futuro este trabajo pueda enriquecerse con datos más detallados, como datos de conexiones de años anteriores a lo obtenido, o de número de conexiones específicas de cada colonia de la ciudad, con el propósito de tener una perspectiva más amplia de lo obtenido hasta el momento.

Además, aunque la sociedad ha vuelto a la normalidad a principios del año 2023, parece que el uso de la red pública por parte de la ciudadanía no ha aumentado en tiempos recientes, lo que demuestra que en caso de seguir disminuyendo o no haber un aumento en los números de conexiones, la cancelación del programa de red pública Wi-Fi sería muy probable en un tiempo cercano.

## Referencias


Puebla Capital. (2024a). Wifi Gratuito. https://www.pueblacapital.gob.mx/wifi-gratuito

Stepanova, E. A., Bankov, D. V., Khorov, E. M., & Lyakhov, A. I. (2022). Influence of the Clock Drift on the Efficiency of the Power Save Mechanisms in Wi-Fi Networks. Journal Of Communications Technology & Electronics, 67(S1), S167-S175. https://doi.org/10.1134/s1064226922130149

Escanez-Exposito, D., Caballero-Gil, P., & Martín-Fernández, F. (2023). Interactive simulation of quantum key distribution protocols and application in Wi-Fi networks. Wireless Networks, 29(8), 3781-3792. https://doi.org/10.1007/s11276-023-03438-x

Gurumekala, T., & Gandhi, S. I. (2022). Toward in-flight Wi-Fi: a neuro-fuzzy based routing approach for Civil Aeronautical Ad hoc Network. Soft Computing, 26(15), 7401-7422. https://doi.org/10.1007/s00500-021-06677-2

Xu, W., Fan, X., Wu, T., Tao, W., Xi, Y., Yang, P., & Tian, C. (2021). Tolerance-Oriented Wi-Fi Advertisement Scheduling: A Near Optimal Study on Accumulative User Interests. Journal On Special Topics In Mobile Networks And Applications/Mobile Networks And Applications, 26(6), 2242-2257. https://doi.org/10.1007/s11036-021-01849-8

Gandhi, J., & Narmawala, Z. (2023). A case study on the estimation of sensor data generation in smart cities and the role of opportunistic networks in sensor data collection. Peer-to-peer Networking And Applications, 17(1), 337-357. https://doi.org/10.1007/s12083-023-01607-5

Witten, I. H., Frank, E., Hall, M. A., & Pal, C. J. (2016). Data Mining, Fourth Edition: Practical Machine Learning Tools and Techniques. https://dl.acm.org/citation.cfm?id=3086818

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer Verlag.

Hastie, T., Tibshirani, R. J., & Friedman, J. (2013). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. http://catalog.lib.kyushu-u.ac.jp/ja/recordID/1416361

Puebla Capital. (2024b). Promedio conectados y conexiones wi-fi. Datos Abiertos. https://datos.pueblacapital.gob.mx/dataset/promedio-conectados-y-conexiones-wi-fi

Puebla Capital. (2024c). Puntos Wi-fi. Datos Abiertos. https://datos.pueblacapital.gob.mx/dataset/puntos-wi-fi

# Capítulo 11

## Una propuesta metaheurística usando Java para el problema de asignación cuadrática

Rogelio González-Velázquez[1], Erika Granillo Martínez[2], María Beatriz Bernabé-Loranca[1], Abraham Sánchez-López[1]
[1] Facultad de Ciencias de la Computación, Benemérita Universidad Autónoma de Puebla, Av. San Claudio, C.P. 72590 Puebla, México
[2] Facultad de Administración, Benemérita Universidad Autónoma de Puebla, Av. San Claudio, C.P. 72590 Puebla, México
rogelio.gzzvzz@gmail.com, erika.granillo76@gmail.com, beatriz.bernabe@gmail.com, abraham.sanchez@correo.buap.mx,

**Resumen.** Uno de los problemas clásicos de optimización combinatoria perteneciente a la clase NP-hard es el problema de asignación cuadrática, el interés en resolver el problema radica en su alta complejidad computacional, así como sus aplicaciones en: logística, circuitos electrónicos, asignación de puertas en aeropuertos, entre otros. En este trabajo se implementó la metaheurística del Procedimiento de Búsqueda Adaptativa Aleatoria Codiciosa para buscar sus soluciones. La principal contribución de este trabajo es la adaptación de una estructura de vecindad contenida en k-intercambios en la fase de post-procesamiento. Las pruebas se realizaron para 29 instancias de gran escala cuyas dimensiones oscilan entre 64 y 254 tomadas del QAPLIB. Las soluciones aproximadas se encontraron a través de una metaheurística que basa su búsqueda en barrios y algoritmos de búsqueda local. Java fue el lenguaje de programación utilizado para la implementación de metaheurísticas, su ejecución permitió equilibrar los parámetros para obtener resultados competitivos respecto a los valores conocidos en la literatura. Los resultados reportados y presentados en este trabajo alcanzaron los objetivos propuestos.

**Palabras clave:** Metaheurística, NP-hard, 2-intercambio, vecindario, optimización combinatoria

## 1 Introducción

El caso más general del problema de optimización se puede escribir como:

$$Optimiza \quad z = f(x) \qquad \textbf{(1)}$$
$$sujeto\ a\ \ x \in \Omega.$$

Donde $\Omega$ es el conjunto de soluciones factibles y $f$ es una función real llamada función objetivo que asocia cada solución factible $x \in \Omega$ con un costo $f(x)$.

La palabra optimizar se puede cambiar para minimizar o maximizar. En el caso del problema de minimización se busca un *mínimo* global, es decir, una solución $x^* \in \Omega$ tal que $f(x^*) \leq f(x), \forall x \in \Omega$ y en el caso del problema de maximización, busca un *máximo* global, es decir, una solución $x^* \in \Omega$ tal que $f(x^*) \geq f(x), \forall x \in \Omega$.

Los problemas de optimización se clasifican comúnmente en dos clases, una cuando el modelo matemático tiene variables continuas que toman valores reales y su espacio de factibilidad es infinito y la otra tiene variables discretas que toman valores enteros y su espacio de factibilidad puede ser de cardinalidad finita.

Se menciona que es un problema de optimización combinatoria (POC), si las variables de decisión sólo admiten valores enteros y su espacio de solución está formado por permutaciones o subconjuntos de números naturales. Por tanto, un POC es un problema de optimización discreto.

Diversos problemas de optimización combinatoria pueden plantearse como problemas de programación entera y resolverse mediante métodos de programación entera exacta como Branch y Bound. Sin embargo, tales formulaciones a veces implican un sin fin variables y restricciones, y aunque varios de estos métodos han demostrado su convergencia teórica, no pueden hacer frente a problemas muy grandes, que generalmente son los que surgen en la práctica de la vida real. Uno de los principales problemas de la programación entera es que la región de viabilidad no es convexa, por lo que no hay garantía de que un problema entero pueda resolverse en un tiempo razonable para una computadora.

Dentro de los problemas de optimización combinatoria existe una clase de problemas de alta complejidad computacional conocidos como NP-hard (Sahni y González, 1976) para los cuales no se conoce ningún algoritmo eficiente que pueda resolverlo en tiempo polinómico y dada la ineficiencia de los métodos exactos para resolver esta clase de problemas se utilizan métodos de aproximación llamados Metaheurísticas (MH). En general, las MH son algoritmos de propósito general que guían una heurística en la búsqueda de soluciones en regiones estratégicas del espacio de factibilidad. Las estrategias de búsqueda más utilizadas en las MH son por vecindarios o por poblaciones con un enfoque inspirado en procesos naturales.

Este trabajo se centra en un método de búsqueda de soluciones mediante vecindades para lo cual se dirá que la vecindad de una solución $x \in \Omega$ es un subconjunto $N(x) = \{x_1, x_2, \cdots x_p\} \subseteq \Omega$, cada solución puede llegar a $N(x)$ desde x mediante un operador llamado mover. Este trabajo se centra en la búsqueda de soluciones al Problema de Asignación Cuadrática (QAP), a través de los conocidos procedimientos metaheurísticos de Búsqueda Adaptativa Aleatoria y Voraz conocidos por sus siglas en inglés Greedy Randomized Adaptive Search Procedure (GRASP), (Resende y Ribeiro, 2016). El QAP consiste en encontrar una asignación óptima entre *n* instalaciones y *n* ubicaciones considerando la distancia entre las ubicaciones y el flujo de materiales entre las instalaciones de manera que se minimice el costo de transporte de materiales. El QAP es un clásico problema de optimización combinatoria y está clasificado como un NP-hard.

El objetivo principal de este trabajo es la adaptación de estructuras de un vecindario en la fase de post-procesamiento de GRASP para comparar el valor mejor conocido (VMC) contra el resultado del valor mejor encontrado (VME). Con los resultados que se pretende encontrar se podrá visualizar y verificar si el procedimiento implementado es considerado una metaheurística robusta para resolver este tipo de problemas. Además, se analizó la eficiencia de GRASP ejecutando dos conjuntos de parámetros, el primero con$\{\alpha = 0.2, \beta = 0.2\}$ y el segundo con $\{\alpha = 0.5, \beta = 0.1\}$.

Las pruebas se realizaron en 29 instancias, 2 de B. Eschermann y H.J. Wunderlich, 8 de Y. Li y P.M. Pardalos, 10 para Skorin Kapov, 6 para E.D. Taillard, uno para U.W.

Thonemann y A. Bölte y 2 para M.R. Wilhelm y T.L. Ward, tomadas de la libreria del QAPLIB (Burkard et al, 2024). El objetivo propuesto es encontrar las soluciones aproximadas a las reportadas en la literatura usando los procedimientos metaheurísticos como una clase de métodos de aproximación diseñados para resolver problemas difíciles de optimización combinatoria (Díaz et al, 1996).

Los problemas de optimización se basan en elegir la mejor configuración entre un conjunto de soluciones factibles para lograr un objetivo (Cela, 1995) y se dividen en dos categorías: problemas con variables continuas y discretas. Para este caso, se tomará la segunda categoría que intenta encontrar un objetivo que se selecciona dentro de un conjunto finito y discreto, un número entero, un conjunto de números enteros, una permutación o un gráfo. Los dos tipos de problemas tienen diferentes métodos de solución; sin embargo, los problemas de optimización combinatoria pertenecen a la segunda categoría (Cela, 1995) como ejemplo, el procedimiento de búsqueda adaptativa aleatoria Voraz (GRASP).

GRASP es un procedimiento iterativo donde cada paso consta de una fase de construcción y una fase de mejora. En la fase de construcción se aplica un procedimiento heurístico constructivo para obtener una buena solución inicial. Esta solución se mejora en la segunda fase mediante un algoritmo de búsqueda local. La mejor de todas las soluciones examinadas es el resultado de (Li et al, 1994) y (Resende, Pardalos y Li, 1996). La fase de mejora podría ser una búsqueda local simplificada o aplicar otra metaheurística para formar algoritmos híbridos. Se pueden utilizan otras metaheurísticas como se mencionó anteriormente como: Búsqueda de Vecindad Variable (BVV), Búsqueda Tabú (BT), algoritmos genéticos (AG) entre otras (Resende y Ribeiro, 2016).

La principal aportación de este trabajo se basa en la experimentación y análisis de los resultados en el método, obteniendo una buena solución inicial y mejorándola posteriormente.

Este artículo está organizado de la siguiente manera: la Sección 1, aborda una introducción al trabajo, en la sección 2, se presenta el trabajo relacionado. En la sección 3, se describe la formulación del problema de asignación cuadrática, en la sección 4, se muestra la metaheurística, en la sección 4.1 se define GRASP, en la sección 4.2 se ubica la implementación de GRASP a QAP. La sección 5 contine los resultados y la discusión, la sección 6, se encuentran las conclusiones y el trabajo futuro. Finalmente, en el apartado 7 referencias.

## 2 Trabajo Relacionado

El Problema de Asignación Cuadrática (QAP) es un problema de optimización combinatoria que consiste en encontrar una asignación óptima de $n$ recursos a $n$ ubicaciones para minimizar el costo de transporte, adicionalmente se necesitan dos matrices, una para los requerimientos de las unidades a transportar y la segunda es el costo del transporte por unidad entre las localidades.

El QAP fue propuesto por (Koopmans y Beckmann, 1957). Posteriormente, dos décadas después se demostró que el QAP es un NP-hard (Sahni y González, 1976). Hasta ahora, se han encontrado soluciones óptimas utilizando métodos exactos para

instancias de tamaño 30 (Roucairol, 1987). El QAP se utiliza en múltiples aplicaciones, tales como: diseño de teclados de computadora, programación de la producción, diseño de terminales de aeropuertos y procesos de comunicación. Se ha utilizado un algoritmo exacto de ramificación y acotamiento con algunas variantes para resolver el QAP que fue presentado por (Roucairol, 1987), sin embargo, recientemente se propusieron soluciones con diferentes técnicas metaheurísticas como en (Zhou,Hao y Duval, 2020), (Hafiz y Abdennour, 2016): algoritmos genéticos, recocido simulado, búsqueda tabú (Skarin-kapov, 1990) y GRASP (Li et al, 1994), (Chmiel et at, 2017). En (Pardalos et al, 1995) implementaron GRASP en paralelo para resolver el QAP.

Otros trabajos relacionados con la búsqueda de soluciones QAP se basan en el algoritmo de enjambre de partículas, la recombinación de operadores para algoritmos genéticos y la cooperativa consciente del estancamiento paralelo a la búsqueda local (Aksan, Dokeroglu y Cosar, 2017), (Tosun, 2014), (Obdelkafi et al, 2016).

El QAP en instancias grandes, es un problema notablemente difícil para encontrar una solución y el rendimiento de los algoritmos metaheurísticos varía, como se menciona en (Saifullah Hussin y Stützle, 2014) donde se comparan dos algoritmos con resultados que dependen del tamaño del problema.

En (Kumar,Sahu y Mitra, 2021) se implementan y comparan los algoritmos de recocido simulado, algoritmos genéticos, búsqueda local iterada, búsqueda tabú para resolver el QAP para instancias grandes. Otro ejemplo de procesamiento paralelo se puede ver en (Dokeroglu, Sevenic y Cosar, 2019) donde la metaheurística se implementa con búsqueda tabú para trabajar con instancias de gran tamaño de QAP.

## 3 Formulación para el Problema de Asignación Cuadrática

El QAP consiste en encontrar una asignación óptima que minimice el costo de transporte de materiales, entre *n* instalaciones en *n* ubicaciones, considerando la distancia entre ubicaciones y el flujo de materiales entre instalaciones. El QAP puede formularse mediante un modelo de optimización combinatoria (Resende, Pardalos y Li, 1996).

Dado un conjunto de $N = \{1,2 \dots, n\}$ y dos matrices asimétricas de tamaño *n* x *n* donde: $F = (f_{ij})$ y $D = (d_{kl})$ , una permutación de $p \in \prod_N$ que se debe encontrarse para minimizar.

$$\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij} d_{p(i)p(j)} \qquad \textbf{(2)}$$

Donde $\prod_N$ es el conjunto de todas las permutaciones de *N* y *F* es la matriz del flujo de materiales entre las instalaciones y *D* es la matriz de distancia entra las ubicaciones $i, j, k, l = 1, \cdots, n.$

# 4 Metaheurística

El principal inconveniente que enfrentan las técnicas heurísticas es la existencia de óptimos locales que no son absolutos. Si durante la búsqueda hay un óptimo local, la heurística no podría continuar el proceso y quedaría "atrapada" en el mismo punto. Para solucionar el problema se recomienda reiniciar la búsqueda desde otra solución inicial y verificar que la nueva búsqueda explore otros caminos (Resende y Ribeiro, 2016).

La mayoría de los problemas de optimización combinatoria son problemas específicos, por lo que un algoritmo de técnica heurística que funciona para un problema a veces no es útil para resolver otros problemas. Sin embargo, en los últimos tiempos se han desarrollado heurísticas de propósito general denominadas metaheurísticas que intentan solucionar los inconvenientes anteriores. La mayoría de las metaheurísticas se desarrollan con métodos de búsqueda de vecindad (Resende y Ribeiro, 2016).

Se acuñó la palabra metaheurística (Díaz et al, 1996) y surgió el término Búsqueda Tabú (1986). Una metaheurística es una estrategia maestra que guía y modifica otras heurísticas para generar mejores soluciones que las que normalmente presentan otros métodos (Mishmast y Gelareh, 2007).

Existen diferentes metaheurísticas exitosas en la resolución de problemas combinatorios. La metaheurística de Greedy Randomized Adaptive Search Problem (GRASP) es una de las técnicas más recientes, fue desarrollada originalmente (Feo y Resendes, 1995) en el momento de estudiar problemas de cobertura de alta complejidad combinatoria (Li et al, 1994). Cada iteración en GRASP generalmente consta de dos pasos: la fase de construcción y el procedimiento de búsqueda local. En una primera etapa se construye una solución inicial que posteriormente se mejora mediante un post-procesamiento para perfeccionar la solución obtenida en la primera etapa hasta obtener un óptimo local.

Hay trabajos donde se aplica esta metaheurística para problemas de optimización en big data (Palmieri et al, 2016). Además, existen otras variantes como: GRASP-path relinking, GRASP-reactivo, GRASP-paralelo, GRASP-híbridos, con algunas otras metaheurísticas cuya búsqueda se basa en vecindades (Festas y Resendes, 2011).

## 4.1 Problema de Búsqueda Adaptativa Aleatoria Voraz (GRASP)

Un GRASP es un proceso iterativo, cada iteración consta de dos fases: la fase de construcción y el procedimiento de búsqueda local. En el primero se construye una solución inicial factible, en el segundo llamado postprocesamiento se implementa mediante una búsqueda local aplicada una permutación inicial obtenida de la construcción, posteriormente se mejora mediante un procedimiento de intercambio hasta obtener un óptimo local. (Koopmans y Beckmann, 1957).

Una vez ejecutadas las dos fases se almacena la solución obtenida y se realiza otra iteración guardando cada vez la mejor solución encontrada en ese momento.

En la figura 1 se muestra un algoritmo que ejemplifica la metaheurística.

```
Procedimiento GRASP
 EntradaInstnacia();
 Mientras (criterio no satisfecho) do
        Fase    1    Construcción    de
    Solución();
        Fase 2 Post-proccesamiento();
        Actualización de Solution();
 Fin
 Regresar (Mejor solucion)
Fin
```

**Fig. 1.** Pseudocódigo Genérico de GRASP. Fuente (Li et al, 1994).

La descripción general de los principales componentes de GRASP son: El componente *Voraz* que utiliza un algoritmo miope para la selección de los componentes que guían la construcción de soluciones, el *Aleatorio* utilizado para las selecciones aleatorias de una lista de candidatos de élite que determinan el camino de la búsqueda, el *Adaptivo* tiene la misión de actualizar cada resultado obtenido de los componentes de la solución que se construye (El Mouayani et al, 2019).

### 4.2 GRASP Para el Problema de Asignación Cuadrática (QAP)

Algunos investigadores han utilizado el diseño de GRASP para resolver el QAP en diferentes casos (Díaz et al, 1996), (Resende, Pardalos y Li, 1996), (Sahni y González, 1976). Cabe señalar que las soluciones son una permutación de longitud *n*, resumiendo el procedimiento de la siguiente manera: Fase de construcción inicial: etapa 1, generación de una lista de candidatos, la cual previamente ha sido restringida por dos parámetros $\alpha$ y $\beta$, luego, aleatoriamente uno es tomado de estos candidatos del que se derivan las dos primeras asignaciones. Etapa 2, se agregan las *n*-2 asignaciones restantes en relación con el procedimiento *Greedy* o voraz, una vez finalizado el proceso, se completa la permutación, produciendo una solución factible. Fase 2 de mejora: la solución generada a partir de la fase 1 se toma como solución inicial de algún procedimiento de búsqueda local, al final del procedimiento se obtendrá una solución óptima local, que también podría ser la óptima global.

La estructura de vecindario usada es: 2-intercambio, que consiste en una solución inicial de $S_0$, generando un vecindario con cardinalidad $C_k^n$. La estructura de vecindario *k*- intercambio ha sido usada en problemas de búsqueda por medio de permutaciones tal como el problema del agente viajero, el problema de particionamiento de grafos, entre otros. La estructura de vecindario 2-intercambio ha alcanzado buenos resultados, si $k \geq 3$ la búsqueda podría tener un costo computacional alto (Resende y Ribeiro, 2016). Se muestra un ejemplo de del vecindario de *n*=5, en la tabla 1:

$$C_2^5 = \frac{5!}{(5-2)!\,2!} = \frac{5*4}{2} = 10 \quad \textbf{(3)}$$

**Tabla 1.** Vecindario 2- intercambio de solución inicial $S_0$ para la matriz Nugent 5

| Vecinadario 2-intercambio relacionado a $S_0$ | | | | | | |
|---|---|---|---|---|---|---|
| | **Permutación** | | | | | **Valor obj.** |
| $S_0$ | 3 | 4 | 1 | 2 | 5 | 35 |
| $S_1$ | 4 | 3 | 1 | 2 | 5 | 31 |
| $S_2$ | 1 | 4 | 3 | 2 | 5 | 34 |
| $S_3$ | 2 | 4 | 1 | 3 | 5 | 35 |
| $S_4$ | 5 | 4 | 1 | 2 | 3 | 39 |
| $S_5$ | 3 | 1 | 4 | 2 | 5 | 39 |
| $S_6$ | 3 | 2 | 1 | 4 | 5 | 30 |
| $S_7$ | 3 | 5 | 1 | 2 | 4 | 37 |
| $S_8$ | 3 | 4 | 2 | 1 | 5 | 35 |
| $S_9$ | 3 | 4 | 5 | 2 | 1 | 30 |
| $S_{10}$ | 3 | 4 | 1 | 5 | 2 | 35 |

En la práctica se toman las matrices de flujo, así como la matriz de distancias y sus elementos se enumeran por separado. los flujos se ordenan de mayor a menor y las distancias de menor a mayor, ambas listas se restringen con un parámetro de $0 < \alpha < 1$ , y se multiplican generando una nueva lista de elementos de la forma $f_{ij}{*}d_{kl}$ que contienen grandes flujos y distancias cortas

Esta lista está restringida con un parámetro $0 < \beta < 1$, con estas operaciones se obtiene una lista restringida de candidatos (LRC), de esta lista se selecciona aleatoriamente un elemento de la forma $f_{ij}{*}d_{kl}$, produciendo el primer par de asignación (*i,k*), (*j,l*), interpretado como la instalación *k* está asignada a la ubicación *i* y la instalación *l* está asignada a la ubicación *j*. Finalmente quedan *n*-2 componentes por asignar, nuevamente con un voraz se calculan los costos $C_{ik}$ respecto de las 2 asignaciones contra las restantes asignaciones posibles, se forma otra LRC y se selecciona uno de estos candidatos con lo cual se genera la tercera asignación y así sucesivamente hasta completar la permutación de *n* componentes llamada solución inicial $S_0$ la cual es sometida a la fase 2, llamada fase de post-procesamiento, completándose una iteración de GRASP (Riffi, Saju y Barkatou, 2017).

## 5 Resultados y discusión

Esta sección muestra los resultados producidos por ejecutar la metaheurística GRASP para 29 instancias extraídas de la librería de QAPLIB (Burkard et al, 2024) para mostrar la eficiencia de la metodología propuesta ya que las instancias varían de tamaño que van desde $n = 64$ hasta $n = 256$, consideradas instancias de gran escala. Para restringir la LRC, primero se ejecutó GRASP con los parámetros de $\alpha = 0.2\ and\ \beta = 0.2$, luego se procesó por segunda vez con los parámetros de $\alpha = 0.5\ and\ \beta = 0.1$ (Li

et al, 1994). El propósito del experimento es observar la eficiencia de GRASP con variaciones en los parámetros de la LRC.

**Tabla 2.** Los resultados que se presentan son para un conjunto de 29 instancias de dimensiones de 60 hasta 256. La primera columna muestra las instancias de prueba, la segunda, con el encabezado MVE contiene el mejor valor encontrado, después, las columnas 3 mejor valor conocido encontrado ($MVCE^1$) y 4 $GAP^1$ corresponde a GRASP con parámetros de $\alpha = 0.2\ y\ \beta = 0.2$, la quinta columna representa el tiempo de cómputo $CPUT^1$. Finalmente, las columnas 6 $MVCE^2$ y 7 $GAP^2$ son de GRASP con parámetros de $\alpha = 0.5\ y\ \beta = 0.1$, el tiempo de cómputo se muestra en la última columna.

| *Instancia* | MVE | $MVCE^1$ | $GAP^1$ | $CPUT^1$ | $MVCE^2$ | $GAP^2$ | $CPUT^2$ |
|---|---|---|---|---|---|---|---|
| esc 64a | 116 | 116 | 0.00 | 16194 | 116 | 0.00 | 17422 |
| esc128 | 64 | 64 | 0.00 | 289358 | 64 | 0.00 | 278097 |
| Lipa 60a | 107218 | 108136 | 0.86 | 31230 | 108132 | 0.85 | 31688 |
| Lipa 60 b | 2520135 | 3007542 | 19.34 | 26586 | 3006630 | 19.30 | 39909 |
| Lipa70 a | 169755 | 170930 | 0.69 | 60449 | 170968 | 0.71 | 60414 |
| Lipa 70 b | 4603200 | 4614766 | 0.25 | 46703 | 4614766 | 0.25 | 50074 |
| Lipa 80 a | 253195 | 254930 | 0.69 | 116748 | 254928 | 0.68 | 96351 |
| Lipa 80 b | 7763962 | 9388572 | 20.93 | 77328 | 9398638 | 21.05 | 152594 |
| Lipa 90 a | 360630 | 360630 | 0.00 | 151700 | 360632 | 0.00 | 152587 |
| Lipa 90 b | 12490441 | 15154028 | 21.33 | 121205 | 15163102 | 21.40 | 129387 |
| sko 64 | 48498 | 49014 | 1.06 | 60200 | 49066 | 1.17 | 69899 |
| Sko72 | 66256 | 66728 | 0.71 | 116687 | 66744 | 0.74 | 114110 |
| Sko 81 | 90998 | 91820 | 0.90 | 455312 | 91970 | 1.07 | 156591 |
| Sko 90 | 115534 | 116654 | 0.97 | 316722 | 116646 | 0.96 | 246604 |
| Sko 100 a | 152002 | 153150 | 0.76 | 402552 | 153282 | 0.84 | 381774 |
| Sko 100 b | 153890 | 155344 | 0.94 | 377645 | 155032 | 0.74 | 379244 |
| Sko 100 c | 147862 | 155032 | 4.85 | 379244 | 149398 | 1.04 | 378249 |
| Sko 100 d | 149576 | 150990 | 0.95 | 369676 | 150272 | 0.47 | 414197 |
| Sko 100 e | 149150 | 150674 | 1.02 | 377337 | 150444 | 0.87 | 377573 |
| Sko 100 f | 149036 | 150430 | 0.94 | 365562 | 150444 | 0.94 | 410407 |
| Tai 60 a | 7205962 | 7438638 | 3.23 | 25455 | 7445042 | 3.32 | 26845 |
| Tai 64c | 1855928 | 1855919 | 0.00 | 18817 | 1855919 | 0.00 | 26845 |
| Tai 80 a | 13499184 | 13865020 | 2.71 | 18817 | 13915442 | 3.08 | 74053 |
| Tai 80 b | 818415043 | 818415043 | 0.00 | 152698 | 818415043 | 0.00 | 152594 |
| Tai 150 | 498896643 | 498896643 | 0.00 | 144817 | 498912784 | 0.003 | 145599 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Tai 256 | 44759294 | 44890864 | 0.29 | 377645 | 44906536 | 0.33 | 379244 |
| tho 150 | 8133398 | 8147865 | 0.18 | 377645 | 8137865 | 0.05 | 383645 |
| Wil 100 | 273038 | 274312 | 0.47 | 466797 | 274312 | 0.47 | 389396 |
| Wil 50 | 48816 | 49010 | 0.40 | 23901 | 49034 | 0.45 | 23351 |

Se puede observar en los resultados dentro de la tabla 1 en la columna GAP1 que para los conjuntos ejecutados con los parámetros de $\alpha = 0.2 , \beta = 0.2$ el porcentaje de error con respecto al MVE y al MVCE[1,2], de 9 instancias fue del 0.0%, 12 casos que van del 0,40% al 0,97%, siendo el error inferior al 1%. Por otro lado, se obtuvo un error del 1% para 2 instancias, para las matrices Sko c y Tai 60 a, 80a, el error oscila entre 2% y 3% finalmente, las instancias Lipa 60b, 80b, 90b, que tienen los porcentajes de error más altos van del 19% al 21% debido a su naturaleza al ser una instancia de mayor escala.

Respecto a los parámetros de $\alpha = 0.5 , \beta = 0.1$ en la columna GAP2, se encontró que nuevamente 9 instancias comparten el porcentaje de error del 0.0%, así mismo, 12 instancias tienen un error menor al 1%, hubo un aumento de dos a tres instancias con un valor del 1%, disminuyendo a dos instancias con porcentajes de error del 3%. Las instancias que quedaron con el mismo porcentaje de error del 19% al 21% fueron Lipa 60b, 80b, 90b.

Para las 29 instancias se observa que el 72% de los conjuntos de prueba tienen un gap de $0 <$GAP$< 1$, el 28% restante incluye el resto de las instancias. Por lo tanto, y según los datos arrojados, se demuestra que la implementación de GRASP es eficiente y a la vez que robusta.

Cabe mencionar que el programa en uso fue diseñado en el lenguaje de programación JAVA por lo que el experimento fue procesado en una computadora Intel Evo Core i5 con un tiempo promedio de cómputo de 189832.7 milisegundos y con una diferencia en el tiempo máximo con relación al mínimo de 450603 milisegundos para el parámetro de $\alpha = 0.2 , \beta = 0.2$. Finalmente, para los parámetros $\alpha = 0.5 , \beta = 0.1$, el tiempo promedio de cómputo fue de 190750 milisegundos con una diferencia entre el tiempo máximo y el mínimo de 396775 milisegundos.

## 6 Conclusiones y trabajo futuro

Como se indicó al inicio de este documento, el objetivo principal de este trabajo es la adaptación de las estructuras de un vecindario en la fase de post-procesamiento de GRASP para comparar el mejor valor encontrado (MVE) contra el resultado del mejor valor conocido encontrado (MVCE). Con los resultados que se pretende obtener se podrá visualizar y verificar si el procedimiento implementado es considerado una metaheurística robusta para resolver este tipo de problemas.

En este trabajo se ha discutido la implementación de una conocida metaheurística GRASP en un lenguaje de programación JAVA para la búsqueda de soluciones aproximadas para un conocido COP clásico, NP-hard QAP. La búsqueda se realiza a través de una estructura vecinal de 2- intercambios. Se realizaron pruebas para una

amplia gama de instancias a gran escala tomadas de QAPLIB (Burkard et al, 2024) con dos conjuntos de parámetros separados para restringir el componente de la LRC de GRASP con el fin de mostrar la variación en la eficiencia de este enfoque.

Para las 29 instancias se observa que el 72% de los conjuntos de prueba tienen un gap de 0 <GAP< 1, el 28% restante incluye a las demás las instancias. Por lo tanto, y según los datos arrojados, se demuestra que la implementación de GRASP es eficiente y al mismo tiempo robusta.

Con base con la discusión de los resultados en la sección 5, afirmamos la naturaleza robusta de este trabajo y que los recortes de la LRC ayudan en la fase de construcción para producir soluciones de alta calidad para minimizar el esfuerzo de la fase de post-procesamiento.

Como trabajo futuro, se planteará introducir una combinación de GRASP con MH bioinspiradas de reciente creación como Gray Wolf Optimizer – (GWO), Firefly Algorithm (FA), entre otros.

## Referencias


Aksan, Y., Dokeroglu, T., y Cosar, A. (2017). "A stagnation-aware cooperative parallel breakout local search algorithm for the quadratic assignment problema", Computers & Industrial Engineering, vol.103, pp. 105-115, https://doi.org/10.1016/j.cie.2016.11.023

Burkard, R.E., Karisc, S.E., y Rendl, F. QAPLIB- A Quadractic Assignment Problem. REcueprado de http://www.imm.dtu.dk/~sk/qaplib/ins.html

Cela, E. (1995) "The Quadratic Assignment Problem: Special Cases and Relatives), Institut für Mathematik B Technische Iniversität Graz. Graz Austria.

Chmiel, W., Kadłuczka, P., Kwiecień, J., y Filipowicz, B. (2017). "A comparison of nature inspired algorithms for the quadratic assignment problema", Bulletin of the Polish Academy of Sciences. Technical Sciences, vol. 65 (4), pp. 513-522, https://doi.org/10.1515/bpasts-2017-0056

Díaz,A.D., Glober,H., Ghaziri,M, Gonzalez,J.L.,Moscato,P., y Tseng,F.T. (1996). "Optimización Heurística y Redes Neuronales en Dirección de Operaciones e Ingeniería"Paraninfo, Madrid.

Dokeroglu, T., Sevenic, E., y Cosar, A. (2019). "Artificial bee colony optimization for the quadratic assignment problema", Applied Soft Computing Journal, vol. 76, pp. 595-60, https://doi.org/10.1016/j.asoc.2019.01.001

El Mouayni, I.,Demesure, G., Bril-El Haouzi, H., Charpentier, P., y Siadat, A. (2019). "Jobs scheduling within Industry 4.0 with consideration of worker's fatigue and reliability using Greedy Randomized Adaptive Search Procedure", IFAC-PapersOnLine., vol. 52(19), pp. 85-90, https://doi.org/10.1016/j.ifacol.2019.12.114

Feo, T.A., y Resendes, M.G.C. (1995). "Greedy Randomized Adaptive Search Procedures", Journal of Global Optimization, vol. 6, pp. 109-133, https://doi.org/10.1007/BF01096763

Festa, P., y Resendes, M.G.C. (2011). "GRASP: basic components and enhancements", Telecommun Syst., vol. 46, pp. 253-271, https://doi.org/10.1007/s11235-010-9289-z

Hafiz, F., y Abdennour, A. (2016). "Particle Swarm Algorithm variants for Quadratic Assignment Problems-A probabilistic learning approach", Expert Systems with Applications. Vol.44, pp. 413-431, https://doi.org/10.1016/j.eswa.2015.09.032

Koopmans,T.C., y Beckmann, M.J. (1957). "Assignment problems and the location of economic activities" Econometrica, vol. 25, pp.53-76. https://doi.org/10.2307/1907742

Kumar, M., Sahu, A., y Mitra, Pinaki. (2021). "A comparison of different metaheuristics for the quadratic assignment problem in accelerated systems". Applied Soft Computing Journal, vol.100, pp. 1-16, https://doi.org/10.1016/j.asoc.2020.106927

Li, Yong., Pardalos, P.M., y Resende, M.G.C. (1994). "A Greedy Randomized Adaptative Search Pro-cedure for the Quadratic Assignment Problem" In P.M. Pardalos and H. Wolkowicz, (eds.): Quadratic assignment and related problems, Vol. 16. Of DIMACS Series on Discrete Mathematics and Theoretical Computer Science. American Mathematical Society, pp.237-261 Rhode Island.

Mishmast, H., y Gelareh, S. (2007). "A Survey of Meta-Heuristic Solution Methods for the Quadratic Assignment Problem", Applied Mathematical Sciences, vol.1 (46), pp. 2293 – 2312.

Obdelkafi, O., Idoumghar, L., y Lepagnot, J. (2016) "Data Exchange Topologies for the DISCO-HITS Algorithm to Solve the QAP", International Conference on Swarm Intelligence Based Optimization, jun 2016, Mulhouse, France. ⟨hal-01665274⟩, DOI: 10.1007/978-3-319-50307-3 4.

Palmieri, F., Fiore, U., Ricciardi, S., y Castiglione, A. (2016). "GRASP- based resourced re-optimization for effective big data access in federated clouds". Future Generation Computer Systems, vol, 54, pp. 168-179, https://doi.org/10.1016/j.future.2015.01.017

Pardalos, P.M., Pitssoulis, L.S., y Resende, M.G.C. (1995). "A Parallel GRASP implementation for the Quadratic Assignment Problem". In A. Ferreira and J. Rolim, editors, Parallel Algorithms for Irregularly Structured Problems – Irregular'94, pp.111-130. Kluwer Academic Publishers, Boston.

Resende,G.C.M., y Riberiro, C. (2016). "*Optimization by GRASP: Greedy Randomize Adaptive Search Procedures"*. Springer, New York.

Resende, M.G.C., Pardalos, P.M., y Li, Yong. (1996). "Algorithm 754: Fortran subroutines for approximate solution of dense quadratic assignment problem using GRASP", ACM Transactions on mathematical software. Vol.22 (1), pp.104-118. https://doi.org/10.1145/225545.225553

Riffi, M. E., Saju, Y., y Barkatou, M. (2017). "Incorporating a modified uniform crossover and 2-exchange neighborhood mechanism in a discrete bat algorithm to solve the quadratic assignment problema", Egyptian Informatics Journal, vol. 18(3), pp. 221-232, https://doi.org/10.1016/j.eij.2017.02.003

Roucairol, C. (1987). "A parallel branch and bound algorithm for the quadratic assignment problema". Discrete Applied Mathematics, vol.18, pp. 211-225, https://doi.org/10.1016/0166-218X(87)90022-9

Sahni, S., y Gonzalez T. (1976). "P-complete aproximations problems", J. Asssoc. Comp. Machine, vol. 23 (3), pp. 555-565, https://dl.acm.org/doi/10.1145/321958.321975

Saifullah Hussin, M., y Stützle, T. (2014). "Tabu search vs. Simulated annealing as a function of the size of quadratic assignment problem instances", Computers & Operations Research, vol. 43, pp.286-291, https://doi.org/10.1016/j.cor.2013.10.007

Skorin-kapov, J. (1990). "Tabu Applied to the Quadratic Assignment Problem", ORSA Journal on Computing, vol. 2(1), pp. 33-45, https://doi.org/10.1287/ijoc.2.1.33

Tosun, U. (2014). "A New Recombination Operator for the Genetic Algorithm Solution of the Quadratic Assignment Problem", Procedia Computer Science, vol. 32, pp. 29–36 https://doi.org/10.1016/j.procs.2014.05.394

Zhou, Y., Hao, J. K., y Duval, B. (2020). "Frequent Pattern-Based Search: A Case Study on the Quadratic Assignment Problem". IEEE Transactions on Systems, Man, and Cybernetics: Systems, pp. 1-13, DOI: 10.1109/TSMC.2020.3027860.

# Índice de autores

# Compiladores

Mireya Tovar Vidal
Ana Laura Lezama Sánchez
Meliza Contreras González

# Revisores

Ana Laura Lezama Sánchez
Cecilia Reyes Peña
Erick Barrios Gónzalez
José David Alanís Urquieta
José Alejandro Reyes Ortiz
José de Jesús Lavalle Martínez
Karina Rosales López
María Auxilio Medina Nieto
Mario Rossainz López
Meliza Contreras González
Mireya Tovar Vidal

# Editores

Mireya Tovar Vidal
Ana Laura Lezama Sánchez
Meliza Contreras Gónzalez

Avances recientes en procesamiento de lenguaje natural y otras áreas afines
Coordinadores de la publicación:
Mireya Tovar Vidal
Ana Laura Lezama Sánchez
Meliza Contreras González
A partir de diciembre de 2024
está disposición en PDF en la página
de la Facultad de Ciencias de la Computación
de la Benemérita Universidad Autónoma de Puebla (BUAP)
https://ontologica.cs.buap.mx/icokg2024/libroICOKG2024.pdf
Peso del archivo: 6.00 MB