Artículo de Investigación

Aibi revista de investigación, administración e ingeniería. Volumen 14, Número 1, Pág 01-13, Enero - Abril de 2026, ISSN 2346-030X

Clasificación de mensajes extorsivos usando modelos de aprendizaje automático y modelo de lenguaje grande

Classifying extortion messages using machine learning models and large language model

Jorge Luis Gutiérrez-Gutiérrez[1]

[1]*Universidad Nacional de Trujillo, Trujillo - Perú*

Recibido: 14 de agosto de 2025.
Aceptado: 11 de diciembre de 2025.
Publicado: 01 de enero de 2026.

Resumen- Este trabajo aborda el problema de la clasificación de mensajes extorsivos; los mensajes extorsivos contienen palabras que expresan una amenaza a alguien, están escritos en español con un sin número de errores ortográficos y palabras abreviadas. Se ha puesto en práctica los conceptos de procesamiento de lenguaje natural, aprendizaje automático y aprendizaje profundo. Se construyó un dataset con mensajes extorsivos y no extorsivos; se entrenaron y probaron tres modelos: uno con random forest, otro con una máquina de vectores de soporte; se entrenó un último modelo con BETO, versión BERT en español. Se evaluaron los resultados obtenidos mediante la comparación de los indicadores de la matriz de confusión de cada modelo y con distintas configuraciones; se concluye que los modelos alcanzaron una exactitud muy alta; BETO obtuvo la mejor exactitud y precisión promedio; sin embargo, el entrenamiento con BETO fue costoso en tiempo y dinero, por lo que los modelos tradicionales tienen buena performance en este problema y puede ser utilizados en problemas de clasificación similares, como por ejemplo, en la clasificación de otros delitos como estafas, acoso, análisis de sentimientos en contextos específicos.

Palabras clave: procesamiento de lenguaje natural, clasificación, aprendizaje automático, modelo de lenguaje grande**.**

***Abstract*—** This paper addresses the problem of classifying extortion messages. The extortion messages contain words that express a threat to someone and are written in Spanish with numerous spelling mistakes and abbreviated words. The concepts of natural language processing, machine learning, and deep learning were applied. A dataset with both extortion and no extorsion messages was constructed; three models were trained and validated: one with a random forest and another with a support vector machine; these models were configured with several hyper parameters for a better analysis. These models are considered traditional models and belong to the machine learning framework. A final model of deep learning was trained with BETO, a spanish BERT release. The results obtained from all the models were evaluated by comparing them with the confusion matrix indicators for each model with several settings. It can be concluded that the models achieve very high accuracy. However, BETO achieved the best accuracy and average precision than others; training was time-consuming and expensive, so traditional models perform well on this problem and can be used in similar classification problems, such as the classification of other crimes such as fraud, harassment, and sentiment analysis in specific contexts.

Keywords: natural language processing, classifying, machine learning, large language model.

*Autor para correspondencia.
Correo electrónico: jgutierrez@unitru.edu.pe (Jorge Luis Gutiérrez Gutiérrez).
La revisión por pares es responsabilidad de la Universidad de Santander.
Este es un artículo bajo la licencia CC BY (https://creativecommons.org/licenses/by/4.0/).
Como citar este artículo: J. L. Gutiérrez-Gutiérrez, "Clasificación de mensajes extorsivos usando modelos de aprendizaje automático y modelo de lenguaje grande", Aibi revista de investigación, administración e ingeniería, vol. 14, no. 1, pp. 01-13 2025, doi: 10.15649/2346030X.5590

I. INTRODUCCIÓN

En esta investigación se presenta un clasificador de mensajes extorsivos mediante la aplicación de técnicas de procesamiento natural y aprendizaje automático. Un mensaje extorsivo es un mensaje que contiene un texto en el cual se lee una amenaza a alguien.

En el Perú existe un nivel de inseguridad ciudadana alta; la ciberdelincuencia se ha incrementado luego de la pandemia. Los delincuentes utilizan la ingeniería social para conseguir datos privados como número de cuentas y contraseñas con la finalidad de cometer delitos en perjuicio de sus víctimas.

Una modalidad conocida es enviar mensajes SMS que son usados como anzuelo para conseguir datos privados. El servicio de mensajes cortos o servicio de mensajes simples, más conocido como SMS (por las siglas del inglés Short Message Service), es un servicio disponible en los teléfonos móviles que permite el envío de mensajes cortos entre teléfonos móviles.

El SMS se diseñó originalmente como parte del estándar GSM de telefonía móvil digital, y actualmente está disponible en una amplia variedad de redes, incluidas las redes 5G [1].

Desde hace 15 años, se viene enviando mensajes extorsivos, se solicita dinero a cambio de no causar daños personales o materiales. Incluso se envían mensajes extorsivos por WhatsApp.

El texto de un mensaje está limitado por el número de caracteres que admite, lo que obliga a usar abreviaturas; como consecuencia aparece en el mensaje palabras mal redactadas, contraídas o en siglas producto de la cultura popular de los delincuentes de la geografía desde donde se escribe.

En los mensajes extorsivos se aprecia un pseudolenguaje donde prima el léxico de los extorsionadores. Así, debemos considerar las palabras en diversas formas para poder reconocerlas y luego clasificarlas en mensajes extorsivos o no extorsivos.

A través de mensajes de texto se comenten delitos como extorsión o estafa; en algunos casos se identifica puntualmente a la víctima, por lo que los mensajes pueden ser enviados de manera individualizada o en masa.

En 2015, en el Departamento de La Libertad las extorsiones a empresarios transportistas era mayormente mediante mensajes de texto amenazantes. Según la Ley de Delitos Informáticos N° 30096, todos estos mensajes son considerados como actos delictivos. En 2024, la mayoría de extorsiones en la ciudad de Trujillo están dirigidos a mineros y comerciantes.

La organización y recuperación de data en textos [2] y la clasificación de mensajes es abordada por diversos tipos de soluciones del área del Machine Learning [3]. Sin embargo, al tener como entrada un mensaje de texto es primordial hacer un preprocesamiento del mensaje a efectos de facilitar su clasificación.

Para tal fin, primero se aplican técnicas de PLN [4] y luego se clasificará mediante un modelo lingüístico grande basado en Transformers [5] y Máquina de Vectores de Soporte [6]. Finalmente, se compararán los resultados de los indicadores de precisión, exactitud, sensibilidad, F1 de la matriz de confusión obtenida de ambos algoritmos.

Existe una necesidad de contar con herramientas capaces de poder detectar mensajes no solamente extorsivos, sino delictivos. El Perú se está convirtiendo en un extorsiona estado; porque cada día se lee más titulares sobre delitos de extorsión y sicariato a dirigentes, comerciantes, mineros, transportistas, dueños de negocios importantes. Una de las formas de perturbar la tranquilidad es mediante el envío de mensajes SMS o a través de WhatsApp; frente a este problema, no se observa que se haya mucho tecnológicamente hablando.

Se tienen aplicaciones para alertar a la policía, para comunicar hechos delictivos a las autoridades; no se observa alguna aplicación, software o servicio en la nube que filtre este tipo de mensaje.

Ante esta situación, se plantea un clasificador de mensajes extorsivos; ante un mensaje recibido el clasificador previamente entrenado debe clasificar correctamente el mensaje en extorsivo o no extorsivo. Esta salida, le permitirá a otra aplicación tomar decisiones si el mensaje es extorsivo. En caso se equivoque, recordar que tiene una precisión menor al 100%, el usuario debe contar con alguna opción que le permita corregir el error y luego reentrenar la red mediante un fine tuning.

La propuesta de estudio tiene como ámbito entrenar y validar el clasificador; no forma parte del alcance del trabajo, el despliegue del clasificador como aplicación o extensión de la aplicación SMS o WhatsApp del teléfono celular. Tampoco las acciones posteriores que haya que hacer una vez clasificado el mensaje como extorsivo, como reportar a alguna central, descartar el mensaje, activar una alarma, entre otras.

Esta investigación tiene como objetivo general: Implementar un clasificador de mensajes extorsivos usando modelos de aprendizaje automático y modelo de lenguaje grande. Los objetivos específicos son: construir un dataset con mensajes extorsivos y no extorsivos, entrenar y validar dos modelos de aprendizaje automático, como son random forest y máquina de vectores de soporte, entrenar y validar un modelo de lenguaje grande BETO mediante fine tuning con el dataset construido, evaluar los resultados obtenidos del entrenamiento y la validación mediante los indicadores obtenidos de una matriz de confusión.

II. MARCO TEÓRICO

El problema de investigación tiene un contexto social derivado del delito de extorsión; se usan servicios digitales mediante mensajes de texto por SMS o WhatsApp para enviar mensajes extorsivos; el gobierno del Perú es lento en tomar decisiones; por ejemplo, luego de una ola de criminalidad ha decidido en 2025 suspender y eliminar líneas de telefonía móvil desde donde se envían este tipo de mensajes.

Las aplicaciones de celular como TrueCaller filtran mensajes spam; dan opciones para bloquear números desconocidos previa confirmación del usuario; almacenan un historial de mensajes.

La clasificación automática de mensajes es un tema importante dentro del aprendizaje automático. Se puede clasificar imágenes y textos. En esta investigación, los mensajes extorsivos vienen en forma de textos, por lo que se requiere en primer lugar realizar una tarea de preprocesamiento de los mensajes.

Tal cual trabaja un traductor, el procesamiento del lenguaje natural puede seguir una secuencia de operaciones para clasificar un mensaje de texto en español.

En el procesamiento de lenguaje natural es importante realizar una serie de operaciones que permitan convertir un texto en lenguaje natural a un registro de un corpus o documento.

Las operaciones son tokenización, borrado de stopwords, identificación de entidades nombradas, conversión de mayúsculas a minúsculas, lematización.

La tokenización separa cada palabra, signo de puntuación, número, símbolo entre otros en un arreglo. El borrado de stopwords elimina del arreglo los signos de puntuación, número, artículos, símbolos, otros textos de menos de 4 caracteres. Es decir, quita aquellos elementos del arreglo que no aportan mayor significado al texto.

Las entidades nombradas, NER, son aquellos nombres propios que deben permanecer en el arreglo; luego se convierte todos los elementos a minúsculas. Finalmente, la lematización es el reemplazo de todos los elementos del arreglo por su lema; lema es cada unidad autónoma con significado léxico raíz de cada elemento; si una palabra está conjugada la reemplaza por su verbo en infinitivo, reemplaza sustantivos en forma simple, singular, sin género.

Bag of words es una técnica para representar textos mediante números en un vector, lo que facilita a una máquina de aprendizaje analizar el texto. Las máquinas de aprendizaje inician su aprendizaje con patrones numéricos. Bag of words representa los textos sin considerar el orden, pero sí la frecuencia de los términos dentro de los textos (documentos); ignora el orden sintáctico, pero guarda la multiplicidad de los términos; para tal fin, calcula métricas de frecuencia como TF (term frequency) o TF-IDF (term frequency – inverse document frequency).

Estas métricas determinan el número de ocurrencias de la palabra en un documento (texto) o en conjunto de documentos (corpus).

Por ejemplo, TF, representa localmente a la frecuencia de un término dentro de un documento. TF-IDF, representa globalmente a un término dentro de un conjunto de documentos.

La técnica bag of words presenta una alta dimensionalidad del vector de números; además presenta dispersión porque en un modelo de muchas palabras; por ejemplo 10 000, un texto de 20 palabras tendrá 19 980 ceros y 20 unos. Además, se señala que tiene una falta de relación con el significado de las palabras.

Los embeddings relacionan la posición de las palabras en el vector con el significado de esta; luego de la construcción de un corpus de textos, se crea un modelo de lenguaje que aprende a predecir palabras; el entrenamiento del modelo genera matrices de embeddings que asocian las palabras con las palabras de su contexto.

Las matrices de embeddings son similares para las palabras que usan contextos similares; es aquí, donde se determina la similitud semántica entre palabras.

Podemos considerar diversos tipos de embeddings como, por ejemplo, embeddings por palabra, embeddings contextuales (por oración) como los que usa ELMo, BERT, GPT embeddings.

Para crear Word embeddings podemos usar algunas técnicas como Word2Vec, GloVe o FastText.

Se pueden entrenar con un corpus de un dominio específico para capturar mejores relaciones o de manera independiente para utilizarlo como codificación inicial para modelos de lenguaje.

La máquina de vectores de soporte es un modelo de aprendizaje automático supervisado que consiste en optimizar la distancia entre los vectores clasificadores de las clases que desea clasificar. Busca calcular la distancia máxima que separa a ambas clases. Originalmente, era un modelo de clasificación binaria; se ha extendido el modelo para clasificar varias clases, multiclase.

Desde otra perspectiva, se puede concebir a las SVM como un modelo de búsqueda; se busca un hiperplano que mejor separe las clases, maximizando el margen entre los puntos más cercanas de cada clase; margen entre los vectores de soporte de las clases. El margen es la medida de qué tan bien separadas están las clases.

Las siglas SVM agrupan tanto a los algoritmos de clasificación, SVC, como a los algoritmos de Regresión, SVR.

Random forest es un algoritmo de aprendizaje supervisado basado en árboles de decisión; fue propuesto como mejora de los árboles individuales como el árbol CART a través de varios árboles que conforman un modelo más robusto, preciso y menos propenso al sobreajuste.

Random forest usa bagging (bootstrap aggregating) y árboles de decisión; bagging tiene como objetivo reducir la varianza sin aumentar significativamente el sesgo; entrena los modelos en paralelo.

Se generan múltiplos subconjuntos aleatorios del conjunto de entrenamiento original con reemplazo; cada subconjunto debe ser entrenado por un árbol de decisión; se considera un número de características para decidir la división óptima.

Si el propósito es con fines de clasificación, se hace una votación mayoritaria entre las salidas del modelo; si el propósito es regresión, se obtiene un promedio.

El Análisis de Componentes Principales (PCA, por sus siglas en inglés Principal Component Analysis) es una técnica estadística utilizada para reducir la dimensionalidad de un conjunto de datos, manteniendo la mayor cantidad posible de variabilidad presente en los datos originales. Su objetivo es transformar un conjunto de variables posiblemente correlacionadas en un conjunto más pequeño de variables llamadas componentes principales, que son combinaciones lineales de las originales.

Las componentes principales son nuevas variables obtenidas como combinaciones lineales de las variables originales, que capturan la mayor cantidad de variabilidad posible en los datos. Se calculan aplicando el álgebra lineal a partir de la matriz de covarianza del conjunto de datos, obteniendo sus autovectores (direcciones de máxima varianza) y autovalores (magnitud de varianza explicada). El primer componente principal es la dirección en la que los datos presentan mayor dispersión; el segundo es ortogonal al primero y representa la siguiente mayor varianza, y así sucesivamente. Estas componentes permiten proyectar los datos en un nuevo espacio de menor dimensión con mínima pérdida de información.

PCA resulta especialmente útil cuando se trabaja con datos de alta dimensión, como en problemas de procesamiento de texto, imágenes o bioinformática. Al proyectar los datos sobre un nuevo sistema de coordenadas formado por los componentes principales, se logra una representación más compacta que facilita el análisis, reduce el ruido, mejora el rendimiento de los algoritmos de aprendizaje automático y permite una mejor visualización de los patrones o agrupamientos presentes en los datos.

En la práctica, PCA se aplica después de centrar los datos (restar la media de cada variable) y, en muchos casos, normalizarlos. Luego se elige el número de componentes principales que se desea conservar, lo cual puede depender de un umbral de varianza explicada (por ejemplo, conservar suficientes componentes que expliquen al menos el 90% de la varianza total). Los datos originales se proyectan en este nuevo espacio de menor dimensión, manteniendo la estructura relevante y descartando redundancias o ruido.

PCA es una herramienta muy utilizada en ciencia de datos y procesamiento de lenguaje natural (PLN) para la visualización de datos, compresión y mejora del rendimiento de modelos. Por ejemplo, en tareas de clasificación de texto, los vectores TF-IDF o embeddings pueden tener miles de dimensiones; aplicar PCA permite reducir ese espacio vectorial sin perder información clave, facilitando así el entrenamiento de modelos de clasificación como Random Forest o SVC.

Un modelo de lenguaje grande es un modelo de aprendizaje profundo que consta de una red neuronal que aprende grandes cantidades de textos sin etiquetar mediante aprendizaje auto supervisado; esta característica es aplicada en el procesamiento de lenguaje natural para diversas tareas como el análisis de sentimientos, el razonamiento matemático, la clasificación de textos, la predicción de textos, entre otras.

Entre modelos de lenguaje grande famosos tenemos: BERT, GPT-4, OPT, LLaMA, Gemini, DeepSeek LLM, entre otros.

BERT, Bidirectional Encoder Representations from Transformers, es un modelo preentrenado de representación de lenguaje desarrollado por Google. Este modelo tiene capacidad de captar la semántica de una palabra observando simultáneamente el contexto que la predice y el que la sucede; el modelo original de BERT se entrenó con corpus de habla inglesa: BookCorpus y Wikipedia en inglés.

BERT usa la arquitectura encoder-only transformer. Los encoders de los transformadores usan mecanismos de atención que les permite procesar texto en paralelo.

La arquitectura de BERT consiste en 4 módulos:

- Tokenizer, convierte una secuencia de texto en una secuencia de enteros (tokens).
- Embedding, convierte la secuencia de tokens en un arreglo de vectores de valores reales; representa la conversión de tipos discretos de token en un espacio euclidiano de baja dimensión.
- Encoder, una pila de bloques transformer con auto atención sin máscara causal.
- Task head, convierte la representación final en un token codificado tipo one-hot a través de la producción de una distribución de probabilidad sobre los tipos de token; puede concebirse como una capa de un-embedding.

BERT fue lanzado en 2018 y se convirtió rápidamente en el modelo de referencia en procesamiento de lenguaje natural. Sin embargo, las versiones multilingües como mBERT no lograban alcanzar el mismo nivel de precisión en tareas específicas de lenguas distintas al inglés; esto motivó a los investigadores a entrenar una versión monolingüe optimizada exclusivamente para el español, lo cual dio origen a BETO.

BETO replica la arquitectura de BERT base (12 capas, 768 unidades ocultas, 12 cabezas de atención, y 110 millones de parámetros), pero con una diferencia sustancial: ha sido entrenado completamente en corpus en español. Los datos utilizados para su entrenamiento provienen de fuentes como Wikipedia en español, noticias, libros y textos públicos recopilados en el corpus de OSCAR, lo que garantiza una cobertura lingüística rica y representativa del idioma.

Los Transformers o transformadores es una arquitectura de aprendizaje profundo desarrollado por Google; mejora las limitaciones de las redes neuronales recurrentes como lo es la falta de procesamiento paralelo y el olvido o pérdida de información de etapas muy anteriores o contextos muy lejanos.

Fine-tuning es el proceso de ajustar pesos de un modelo preentrenado como un transformer; para lo cual debemos usar datos etiquetados; no hay que entrenar el modelo de nuevo; prácticamente es el aprendizaje de cuestiones específicas.

Fine-tuning supera en precisión a TF-IDF y Máquina de vectores de soporte, también a TF-IDF y Random Forest; demanda pocos miles de ejemplos.

Puede adaptarse a múltiples tareas como la clasificación, extracción de características, análisis de sentimiento, entre otras.

No es necesario usar fine-tuning, cuando se tienen pocos recursos computacionales o las tareas son sencillas.

Si se hace fine-tuning de BETO con pequeños datasets, el modelo resultante puede ser aplicado en tareas específicas como la inferencia de lenguaje natural y la clasificación de textos, puede entender intenciones implícitas, amenazas indirectas, variaciones de tono; también puede ser aplicado en la generación de lenguaje basado en secuencia a secuencia, por ejemplo, en chatbots o en aplicaciones de pregunta y respuesta.

Hay muchos algoritmos de clasificación de textos; se han identificado seis algoritmos: árboles de decisión, vecinos más cercanos, retroalimentación relevante, clasificador ingenuo, máquina de vectores de soporte, ensamble. Los resultados de comparar dos de estos algoritmos como kNN y Naïve Bayes, arrojaron como resultado que kNN tiene un valor F1 de 55.17 para k=13; mientras que Naïve Bayes arrojó un F1=39.01. Se recomienda usar un valor grande k superior a 10 [7].

Se propuso un modelo informático para el análisis de sentimiento de textos manuscritos y electrónicos a través de algoritmos de aprendizaje automático; está diseñado para clasificar emociones en diversas redes sociales. Se hace un preprocesamiento de texto, se convierte la imagen a texto mediante OCR, se selecciona las características, se clasifica el texto, se analiza los resultados [8].

En una investigación sobre análisis de sentimiento, se realizó una limpieza de datos en un set recolectado de twitter, sirvieron de entrada para una máquina de vectores de soporte que clasifique la polaridad de estos mensajes provenientes de una comunidad de usuarios del videojuego Liga de Leyendas. El clasificador alcanzó 65.08% de exactitud [9].

En un estudio se compara diversos modelos de clasificación de mensajes de odio con diferentes dataset; algunos están desbalanceados por lo que se recomienda balancearlos a efectos de que el algoritmo clasificador no prefiera la clase dominante en cuanto al número de registros. Se aplicaron cuatro modelos: modelo de red neuronal simple (SNN), red neuronal convolucional (CNN), memoria de corto-largo alcance (LSTM) y BETO. Se obtuvo una exactitud para BETO y DL de 89% y 87%; modelos que ofrecieron los mejores resultados en varios datasets balanceados [10].

Fraud-BERT es un marco contextual basado en transformadores, usa aprendizaje por transferencia y lo evalúa en un conjunto de datos de ofertas de empleo falsas altamente desequilibrado conocido como Employment Scam Aegean Dataset disponible en kaggle.com.

Varios modelos de aprendizaje automático han usado métodos de extracción de características como Bag of Words y TF-IDF; sin embargo, estos métodos no son contextuales, muestran asimetría en los resultados debido a desequilibrios en la distribución de datos. Este marco maneja datos desequilibrados con una puntuación F1 de 0,93 y una exactitud del 99% [11].

Se comparó tres algoritmos para la clasificación de textos sobre opiniones de estudiantes sobre el desempeño de un profesor. Se consideran tres clases: positivo, negativo, neutro. En la investigación se comparó tres algoritmos: SVC, que es una variante de máquina de vectores de soporte, MultinomialNB, que es una variante de Naïve Bayes, Pasivo agresivo; los resultados fueron comparados usando métricas de bag of words y TF-IDF; se observa que el algoritmo pasivo agresivo obtuvo los mejores resultados con parámetro C=0.03 [12].

Se destaca la robustez de BERT en la predicción de propiedades químicas y su aplicación en la quimioinformàtica y bioinformàtica; el estudio implicó el preentrenamiento y el ajuste del modelo BERT en datos relacionadas a la COVID-19 y malaria utilizando representaciones SMILES y DeepSMILES. Se configuraron diversos entrenamientos con diferentes embeddings posicionales para aplicaciones de clasificación y regresión [13].

Se propone un IDS de red basado en BERT; se convierte los datos del tráfico de red en representaciones textuales; el modelo captura información textual en secuencias; luego de la extracción de característica se usa GRU para procesar eficientemente los datos de secuencia y reducir la complejidad computacional, captura además las dependencias temporales. Al entrenar el clasificador con las características se distingue con precisión el tráfico normal del anormal. Al comparar resultados con otros métodos tradicionales de aprendizaje automático y aprendizaje profundo, se observa que el modelo BERT mejora la precisión de detección y captura eficaz de patrones de ataque complejo [14].

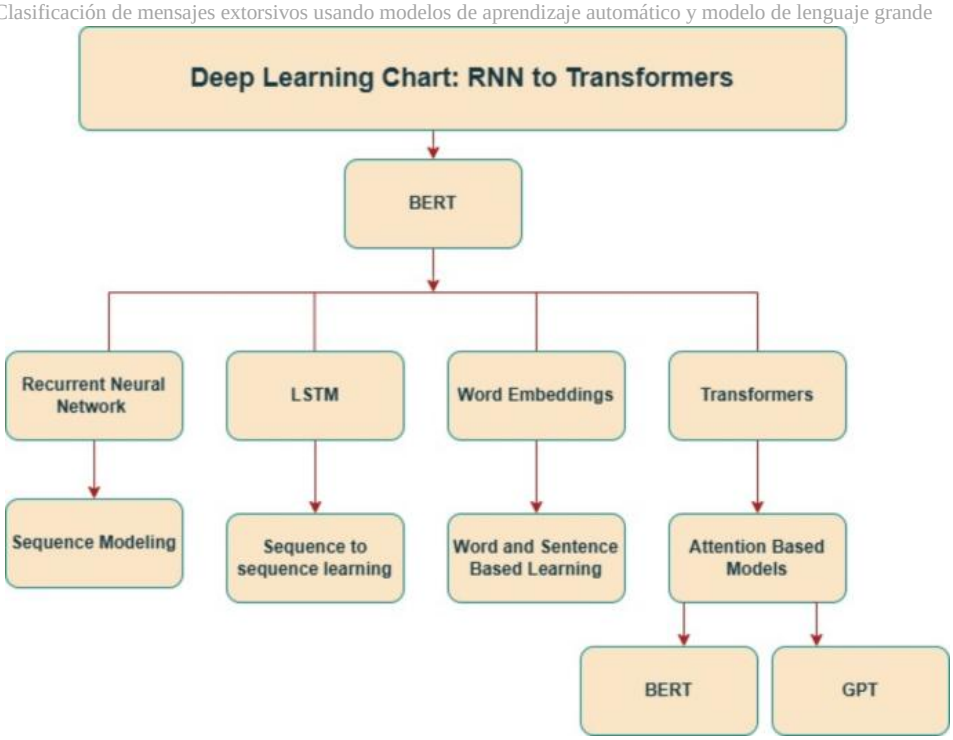

Fig. 1: Evolución las redes neuronales profundas [15].

Diversos estudios han intentado distinguir el contenido textual generado por IA del trabajo de autoría humana; se compiló un conjunto de datos que consta de contenido escrito por humanos y generado por IA (ChatGPT); se utilizó este dataset para entrenar y evaluar varios modelos de aprendizaje automático y aprendizaje profundo. Los resultados experimentales demuestran que el modelo de aprendizaje profundo personalizado basado en RoBERTa logró una puntuación F1 de 0,992 y una precisión de 0,991, seguido de DistilBERT, que tuvo una puntuación F1 de 0,988 y una precisión de 0,988 [16].

Se comparó RoBerTa con diversos modelos de procesamiento de lenguaje natural como VADER; siendo más conveniente para una investigación en análisis de sentimiento en procesamiento cuántico, toda vez que tiene la capacidad de análisis del contexto similar o lo ofrecido por Lambeq [17].

Se usaron técnicas basadas en BERT para detectar rumores en redes sociales, principalmente de twitter; la propuesta fue denominada CE-BERT, BERT conciso y eficiente; reduce costos computacionales al reducir el número de capas de BERT de 12 a 4, emplea solo 53 millones de parámetros en vez de los 110 millones que usa BERT, mejora la performance exhibida en BERT y RoBERTa al haber evaluado sobre tres datasets PHEME, Twitter15 y Twitter16; el modelo propuesto consta de tres pasos: capas de selección efectiva, fine-tunning eficiente, y clasificación [5].

Las redes sociales como Reddit han aprovechado las prestaciones que presentan para compartir recomendaciones de salud y tendencias en enfermedades; puesta a prueba sobre todo en la pandemia de la COVID-19. En la investigación descrita se diseñó una aplicación que permite a los usuarios hacer preguntas y obtener respuestas en español sobre temas relacionados con la salud en el sistema mexicano y argentino, derivados de los comentarios de Reddit.

Para tal fin, usa algoritmos de PLN como el transformador de transferencia de texto a texto (T5) y las representaciones de BERT, ambas tecnologías desarrolladas por Google. Dado una pregunta como entrada, el sistema recuperará un comentario de Reddit usando BERT preentrenado y la búsqueda de similitud de Facebook AI (FAISS) [18].

III. METODOLOGÍA O PROCEDIMIENTOS

Luego de revisar el estado del arte de la clasificación automática de textos, se requirió construir un dataset con mensajes extorsivos y no extorsivos; se recolectó mensajes extorsivos reales de una dependencia policial en la ciudad de Trujillo, además del Poder Judicial, redes socales, diarios digitales; se extrajo los tokens principales, se categorizaron. Se consiguió obtener 240 mensajes en total.

Aplicando la regla 80-20, se dividió el corpus en dos partes; la primera parte contó con mensajes para el entrenamiento de los modelos de aprendizaje automático y de aprendizaje profundo; la segunda, contó con mensajes que sirvieron para la prueba. Se ha asegurado que no haya mensajes repetidos en el entrenamiento y la prueba.

A continuación, se determinaron los modelos de aprendizaje automático y de aprendizaje profundo necesarios para aprender el corpus. Los modelos de aprendizaje automático fueorn random forest y máquina de vectores de soporte; el modelo de aprendizaje profundo se implementó mediante un modelo fine tuning basado en BETO. Puesto que los modelos utilizan aprendizaje supervisado, se requiere etiquetar las clases del dataset; además, para reducir el sesgo, el dataset será balanceado.

Finalmente, se aplicó una matriz de confusión para comparar los tres modelos mediante los indicadores de exactitud, precisión, recall, F1 de todos los modelos entrenados. También se comparó los tiempos de entrenamiento y prueba.

Una limitación del estudio ha sido el acceso a una mayor cantidad de mensajes reales; no se cuenta con repositorios abiertos para obtener mensajes reales motivo de esta investigación.

Esta investigación se ha ejecutado siguiendo los principios éticos que rigen la investigación científica, guardando la confidencialidad y el anonimato de las personas involucradas en el envío y recepción de mensajes.

IV. RESULTADOS, ANÁLISIS E INTERPRETACIÓN

a. Construcción de dataset

Se recogieron 240 mensajes, de los cuales 120 mensajes fueron extorsivos, recolectados de una dependencia policial en la ciudad de Trujillo; como se observa en la siguiente tabla, los mensajes tienen errores ortográficos, palabras contraídas.

En el análisis de los mensajes se ha podido identificar las siguientes componentes estructurales:

- Insultos fuertes; palabras soeces.
- Verbos sobre pagos; pagar, dar plata, colaborar, arreglar.
- Sustantivos sobre dinero; plata, dinero, cuota, tarifa, deuda.
- Sujetos amenazados; la víctima, su familia, sus seres queridos.
- Verbos de amenaza; morir, matar, destripar, meter bala, descuartizar, enfriar, quemar.
- Advertencias; vigilancia, conocimiento de rutina, conocimiento de domicilio, ubicación.
- Elementos de presión; plazo, fecha, demora.
- Jergas, tombos, chibolo, jato, soplón, joder, cagar.
- Afirmaciones de seriedad; no jugamos, chocando fuerte, la víctima decide, hablamos en serio.

Se analizaron los mensajes no extorsivos; se hizo un análisis lingüístico de los mensajes, el cual consistió en un análisis léxico del idioma español mediante la tokenización, la identificación de n-gramas (unigramas y bigramas) y el conteo de longitud. Se reportó uso del idioma informal y cotidiano.

Los mensajes tienen verbos en modo imperativo. Se identificó las temáticas del mensaje. Se advirtió los errores ortográficos que representan una forma de escritura informal. Se precisó la longitud del mensaje en palabras, se observó mensajes en minúsculas sin puntuación formal.

Se hizo un análisis de estilo y tono mediante la aplicación de un modelo de análisis morfosintáctico basado en reglas. Se usó un agrupamiento semántico para agrupar mensajes según tema (avisos, saludos, amenazas, tareas, servicios, entre otros).

Los mensajes están etiquetados; se ha establecido arbitrariamente 0 como etiqueta para los mensajes no extorsivos y 1 para los mensajes extorsivos. Para tal fin el programa randomiza los diversos tokens extraídos del análisis de los mensajes iniciales.

Ahora corresponde hacer un preprocesamiento del texto, toda vez que los mensajes son oraciones en algunos casos no bien escritas, con algunas letras mayúsculas, con espacios en blanco. La conversión a minúsculas no se requiere hacerlo porque los métodos que se usarán como TfidfVectorizer ya lo hace por defecto. Lo mismo pasa con las entidades nombradas, NER, porque no se requiere analizar personas, lugares, organizaciones de manera explícita.

El preprocesamiento consistió en:

- Tokenizar, importante porque los textos tienen errores ortográficos, abreviaturas, contracciones, lenguaje informal.
- Eliminar Stopwords, reduce el ruido; pueden ser útiles para comprender el contexto o la intensidad de la amenaza.
- Lematizar, todas las conjugaciones amenazantes se abstraen con un solo lema; por ejemplo, mataré, matarán, mataremos, se lematizan en un solo lema, matar.

Fig. 2: Preprocesamiento de dataset adoptado.

Para tal fin, se requiere usar en primer lugar una tokenización avanzada; el uso de algunas librerías como spacy o nltk mejoran los resultados. Se usó también esta librería para lematizar.

Se ha considerado algunos registros del dataset a efectos de ejemplificar el pipeline propuesto para PLN. Por ejemplo, se muestra a continuación el preprocesamiento para un mensaje de entrada.

"si no pagas vas a morir y tu familia se va contigo uno por uno estas advertido"
Luego de la tokenización, queda separado en palabras:
['si', 'no', 'pagas', 'vas', 'a', 'morir', 'y', 'tu', 'familia', 'se', 'va', 'contigo', 'uno', 'por', 'uno', 'estas', 'advertido']
Luego de la eliminación de stopwords, se quitaron palabras como si, a, y, tu, se, por; queda:
['pagas', 'vas', 'morir', 'familia', 'va', 'contigo', 'uno', 'uno', 'estas', 'advertido']
Luego de la lematización, queda:
['pagar', 'ir', 'morir', 'familia', 'ir', 'contigo', 'uno', 'uno', 'estar', 'advertir']

Aibi revista de investigación, administración e ingeniería. Volumen 14, Número 1, Enero - Abril de 2026, ISSN 2346-030X PP 01-13.

b. Entrenamiento y validación de los modelos

Para entrenar estos modelos es necesario vectorizar los lemas del dataset; la vectorización permitió convertir los textos (lemas) en números que serán las entradas requeridas por el modelo random forest y el clasificador de vectores de soporte, SVC.

Hay diversas maneras de vectorizar como bag of words, TF, o TF-IDF; en esta investigación se usó TF-IDF, porque resalta las palabras que son representativas de un texto específico y no las que aparecen en todos los textos (documentos). Para estos modelos es adecuado, puesto que no requiere un entrenamiento complejo; TF-IDF se presta también porque el vocabulario es limitado; además, palabras distintivas de un mensaje extorsivo como matar, pagar, entre otras, son más valoradas por TF-IDF lo que favorece la clasificación.

Se ha configurado el hiperparámetro n_estimators con valor 50, 100, 250 para ejecutar diversas ejecuciones de random forest; significa que random forest usará estas cantidades de árboles por cada entrenamiento. A continuación, se muestra los resultados de la matriz de confusión para todos estos valores de n_estimators.

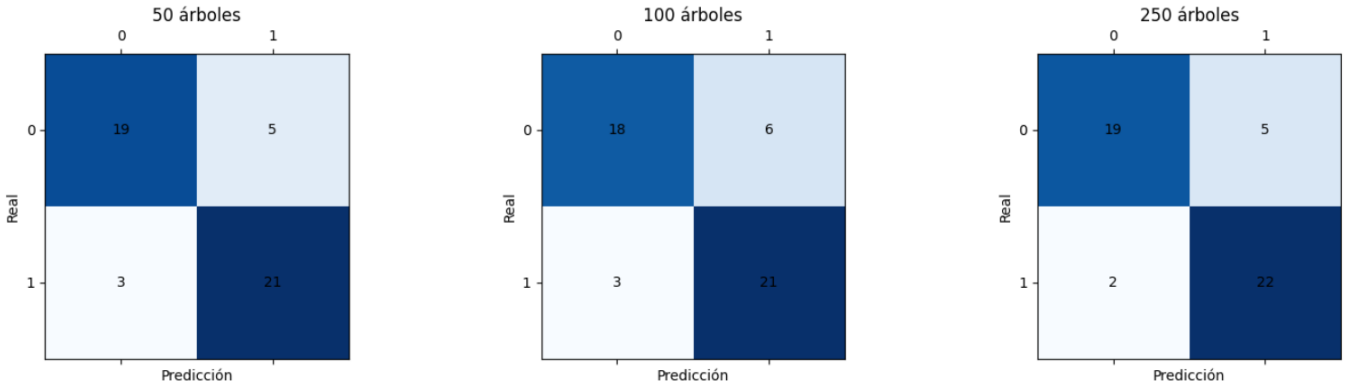

Fig. 3: Matriz de confusión para 50, 100 y 250 árboles.

Por experiencia propia, 100 es un valor recomendado en vez de tener pocos árboles; permita obtener mayor exactitud (accuracy) sin reñir con el sobreajuste. Además, aumentar el número de árboles reduce el sobreajuste de cada árbol, por ende, aumenta la generalización; esto es bueno porque mejora la precisión.

TABLA I
REPORTE DE CLASIFICACIÓN – RANDOM FOREST

N_estimators	accuracy	precision	recall	f1-score
50	0.8333	0.8636	0.7917	0.8261
100	0.8125	0.8571	0.7500	0.8000
250	0.8542	0.9048	0.7917	0.8444

Sin embargo, se debe tener un límite razonable porque muchos árboles aumentan el costo computacional generando un tiempo excesivo de entrenamiento, sin hallar una mejor precisión de la ya obtenida; podemos verificar en la tabla los resultados del tiempo de entrenamiento y el tiempo de predicción para cada bosque aleatorio entrenado.

TABLA II
COMPARACIÓN DE TIEMPOS DE ENTRENAMIENTO Y PRUEBA

N_estimators	Tiempo de entrenamiento (segundos)	Tiempo de prueba (segundos)
50	0.1033	0.0041
100	0.2107	0.0088
250	0.5298	0.0174

SVC requiere que las clases estén balanceadas, se hizo previamente. SVC es muy sensible al ajuste de los hiperparámetros; los valores por defecto establecen función de bases radiales para el kernel, gamma de tipo escala, y C = 1.0. C aumenta la penalización por errores; un valor alto de C hace que el margen de error sea pequeño; lo que puede sobreajustar el entrenamiento. Por el contrario, un valor muy bajo, relaja el modelo.

Se hizo un análisis de sensibilidad probando con diversos valores de C; por ejemplo, con 0.1, 1.0, 10; y con estos dos valores para el kernel: rbf (función de base radial) y linear. El tipo de frontera de decisión para el kernel linear es una línea; demanda menor complejidad computacional por lo que es más rápido el entrenamiento; es más fácil interpretar los resultados y la probabilidad de sobreajuste es menor si C no es muy alto.

Se justifica la consideración de la evaluación del kernel linear porque los textos suelen estar en espacios vectoriales de alta dimensión y puede ser eficiente. No obstante, se evaluará el modelo con ambos kernels. El número de registros para el entrenamiento es 80% y para la validación es 20%.

La validación cruzada favorece la evaluación del modelo, toda vez que asegura que sea más confiable y generalizable. Se optó por 5 fold que consiste en dividir el dataset en cinco partes, con cuatro de ellas se entrena y con una se prueba. Este proceso se repite 5 veces, así cada una de las partes se usará para prueba; esto permite evaluar todas las partes del dataset y obtener una visión más robusta; finalmente se promedia el resultado para obtener una estimación más realista.

Se presentan las matrices de confusión acumulada para diversos valores de C, con kernel linear y luego con kernel rbf.

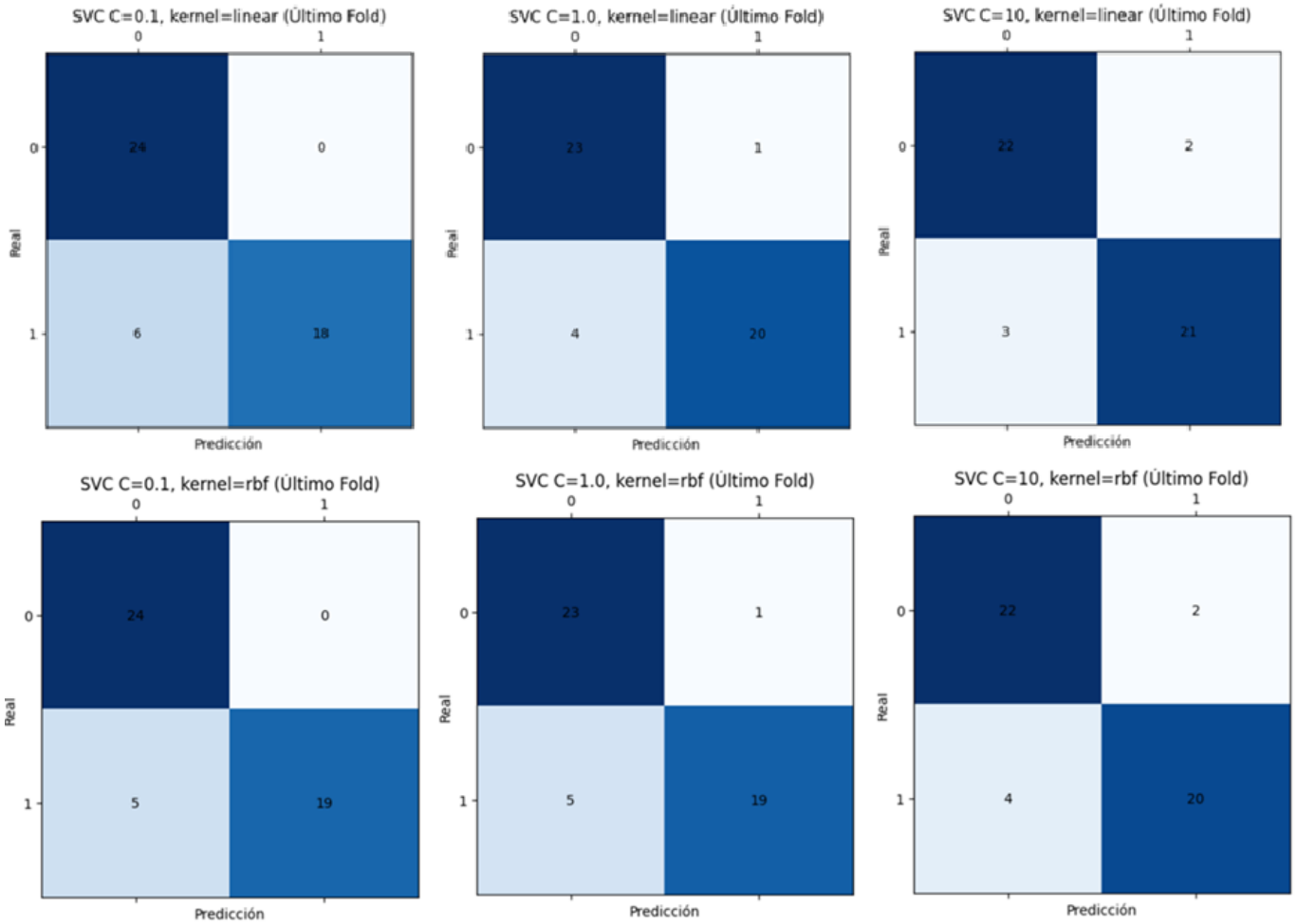

Fig. 4: Matriz de confusión para SVC, C= 0.1, 1.0, 10, kernel linear y rbf.

TABLA III
REPORTE DE CLASIFICACIÓN – SVC

C	kernel	accuracy	precision	recall	f1-score
0.1	Linear	0.8875	0.9157	0.8583	0.8831
1.0	Linear	0.9083	0.9171	0.9000	0.9075
10	Linear	0.8917	0.8995	0.8833	0.8909
0.1	Rbf	0.9083	0.9079	0.9167	0.9084
1.0	Rbf	0.9083	0.9250	0.8917	0.9064
10	Rbf	0.9083	0.9085	0.9083	0.9079

TABLA IV
TIEMPOS DE ENTRENAMIENTO DE SVC CON DIVERSOS VALORES DE C Y KERNEL

C	kernel	Tiempo entrenamiento (segundos)	Tiempo predicción (segundos)
0.1	Linear	0.01	0.0017
1.0	Linear	0.01	0.0015
10	Linear	0.01	0.0015
0.1	Rbf	0.01	0.0017
1.0	Rbf	0.01	0.0017
10	Rbf	0.01	0.0017

Para entrenar BETO, modelo de lenguaje grande, es necesario primero tener el dataset preprocesado; esto es, sin stopwords, textos en minúsculas. Luego, se requiere tokenizar, lo que divide el texto en sub palabras o tokens. A continuación, se vectoriza, lo que permite convertir los tokens en identificadores del vocabulario de BETO.

Los identificadores se transforman en tensores para ser usados por el modelo. Estos tensores son los embeddings que requiere el entrenamiento del modelo.

El modelo BETO procesa los embeddings para predecir las etiquetas de las clases de los mensajes que se les presenta; para tal fin, usa su capa transformer para procesar contexto y relaciones entre tokens; su capa de salida clasifica o predice la etiqueta de la clase de mensaje.

Fine-tuning es necesario porque se desea que un modelo preentrenado como BETO aprenda características más específicas del problema en donde se aplica; en este caso, aplicado a clasificar mensajes extorsivos.

Se dividió el dataset con la regla 80% - 20%; 80% para el entrenamiento y 20% para la evaluación. Después, se cargó el modelo de clasificación con dos etiquetas. Finalmente, se entrena el modelo con 4 épocas, fue configurado para calcular los mismos indicadores que los modelos de aprendizaje automático.

Para el entrenamiento se usó el servicio Amazon Sage Maker Studio Lab. También se autenticó el token necesario desde una cuenta de Hagging Face para que el servicio de computación en la nube de Amazon pueda acceder a BETO (BERT basado en español); disponible en https://huggingface.co/dccuchile/bert-base-spanish-wwm-uncased.

La configuración de BETO tiene en cuenta los siguientes hiperparámetros:

- Learning rate; es la tasa de aprendizaje; es el hiperparámetro más crítico; controla la actualización de pesos en cada paso. Se ha considerado 2e-5 para el presente entrenamiento.
- Batch size; es el número de ejemplos procesados antes de actualizar los pesos; se ha considerado 8 para el entrenamiento y para las pruebas. Tamaños mayores demandan mayor memoria.
- Número de épocas; se ha considerado solamente 4 por ser un proceso de fine-tuning. Demasiadas épocas pueden causar sobreajuste con dataset pequeños como el del presente entrenamiento.

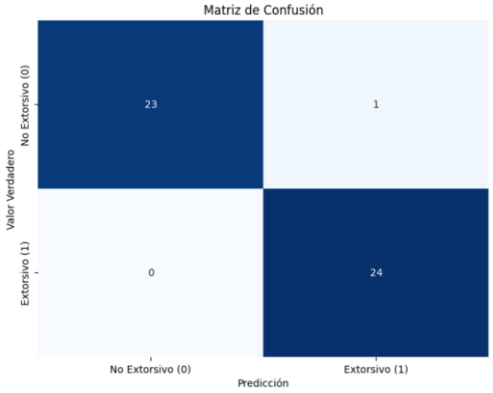

Fig. 5: Matriz de confusión para BETO.

TABLA V
REPORTE DE CLASIFICACIÓN – BETO

Modelo	accuracy	precision	recall	f1-score
BETO	0.9792	0.9600	1.000	0.9796

Los modelos evaluados fueron tres: random forest, máquina de vectores de soporte y BETO (BERT en español); BETO tuvo la mejor exactitud (accuracy), precisión y sensibilidad (recall); seguido de la máquina de soporte de vectores para la clasificación, y en último lugar random forest.

Sin embargo, el entrenamiento y el costo computacional fueron diferentes. Mientras que en los dos modelos de aprendizaje automático (modelos tradicionales) el entrenamiento duró segundos, en el caso de BETO (aprendizaje profundo) duró 10 minutos 40 segundos.

c. Discusión

Los mensajes utilizados en el dataset son representativos de la clase extorsivos y no extorsivos; el sobreajuste (over fitting) no es bueno, puesto que cualquier modelo pierde capacidad de generalización, lo que deviene en mala clasificación cuando se le presenta datos no vistos en el entrenamiento. Para garantizar que el modelo entrenado no está sobreajustado, se verificó si el problema de clasificación es linealmente separable; esto significa que, el modelo se puede clasificar con facilidad mediante un hiperplano, o también que, las características (tokens del mensaje) son muy predictoras.

Para comprobar la hipótesis, se presenta la visualización de datos mediante Análisis de Componentes Principales, PCA. El propósito del PCA es transformar un conjunto de variables correlacionadas en un nuevo conjunto más pequeño de variables no correlacionadas denominadas componentes principales; estos componentes, capturan la mayor parte de la variabilidad presente en los datos originales.

Es así como se aplicó esta técnica PCA para visualizar los datos y comprobar si se trata de un dataset linealmente separable. Se muestra a continuación la visualización obtenida mediante PCA lo que permite confirmar que se trata de un modelo linealmente separable.

Los resultados obtenidos mejoran los obtenidos en [19] quienes entrenaron un modelo basado en Naïve Bayes y alcanzó una exactitud de 0.933.

El problema abordado tiene una complejidad menor que el análisis de sentimientos en mensajes de twitter; por lo que los modelos entrenados tienen una exactitud mejor que los encontrados en [9]; a pesar de que se han usado modelos similares, el análisis de sentimiento demanda un análisis mucho más semántico que la clasificación de mensajes.

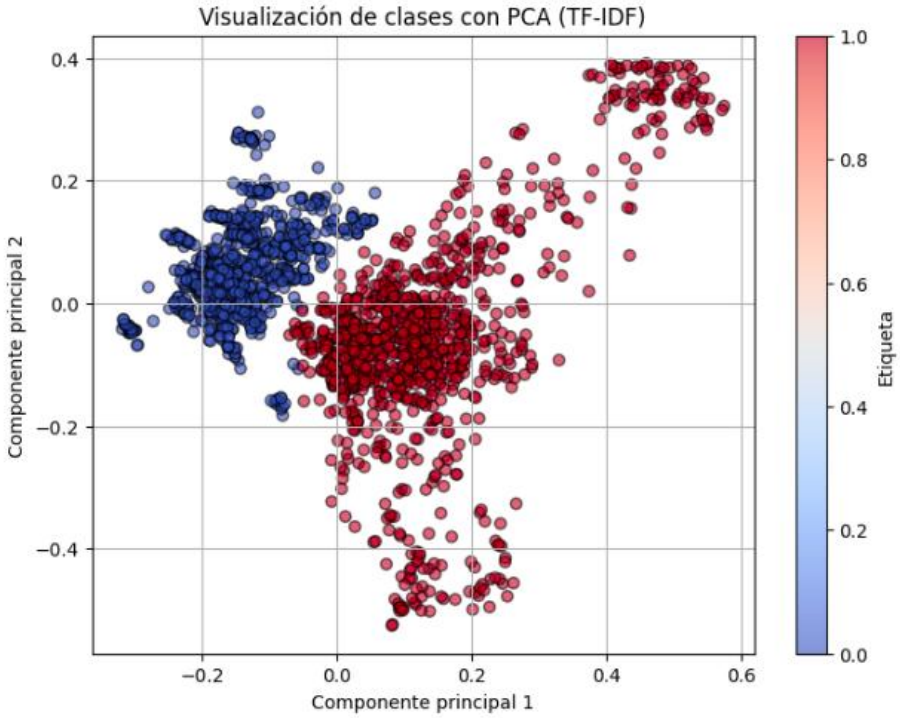

Fig. 6: Visualización de clases mediante Análisis de Componentes Principales.

Como bien se comprueba en [20], el uso de BERT se aplica para clasificaciones donde importa el contexto; en esta investigación el contexto queda reducido a un conjunto pequeño de palabras frecuentemente utilizadas en los mensajes extorsivos; todo lo demás que se requiere para tratar el mensaje extorsivo como el análisis sintáctico o semántico lo maneja muy bien BERT y su derivado en español, BETO.

Este trabajo también tiene implicancias prácticas a tener en cuenta; el uso de modelos tradicionales tiene una ventaja significativa en términos de accesibilidad tecnológica y coste computacional; estos modelo pueden ser entrenados y ejecutados eficientemente en entornos que tienen hardware limitado como laptops o computadoras de escritorio de usuario final, por lo que estos modelos es la opción viable para muchas organizaciones estatales, sin fines de lucro o voluntarias que no cuenten con infraestructura avanzada de computación.

Se ha podido verificar que los tiempos de entrenamiento en random forest y en SVC son menores a 1 segundo, por lo que son mucho más pequeños que los de BETO, que fue de varios minutos; este resultado es importante, lo cual resulta clave en aplicaciones donde es crítica la respuesta rápida. Por otro lado, BETO puede ser mucho más confiable al tratar con volúmenes grandes de textos o con corpus que contengan lenguajes ambiguos; por lo que podemos recomendar su uso, todo depende de las necesidades que se tengan.

V. CONCLUSIONES

Existen muchos trabajos interesantes que proponen diversos modelos capaces de clasificar textos para un contexto específico; algunos van mucho más allá que la clasificación, como los que generan textos, mediante predicción de palabras faltantes o respuestas adecuadas a preguntas planteadas.

Se construyó el dataset balanceado con el mismo número de instancias de ambas clases: extorsivos y no extorsivos. El dataset contiene palabras que provienen de mensajes extorsivos reales y que no se repiten en el entrenamiento. Se dividió el dataset en 80% para el entrenamiento y 20% para la prueba.

Se usó el mismo método de evaluación para comparar tres modelos: random forest, máquina de vectores de soporte, BETO.

Se entrenó random forest con 50, 100 y 250 árboles; la mayor exactitud fue 0.8542, obtenida con 250 árboles; de igual forma, con 250 árboles se obtuvo la mayor precisión promedio, 0.9048 por cada una de las dos clases etiquetadas en todos los entrenamientos.

Se entrenó una máquina de vectores de soporte clasificadora; se hizo validación cruzada de tipo 5 fold y se configuró diversos valores de C para penalizar el aprendizaje como 0.1, 1.0 y 10; además se entrenó el modelo para los kernel rbf y linear; la mayor exactitud obtenida fue 0.9083 con valor C igual a 1.0; la mayor precisión fue de 0.9250 con un valor C igual a 1.0 y con kernel igual a rbf.

Se entrenó el modelo BETO mediante fine tunning con el dataset construido; los resultados de entrenamiento obtenidos como exactitud, precisión, recall y f1-score fueron mejores que los obtenidos con los dos modelos de aprendizaje automático; se obtuvo una exactitud de 0.9792 y una precisión de 0.9600; se obtuvo una sensibilidad de 1.0, lo que significa que BETO idenitificó correctamente todos los mensajes extorsivos; el costo computacional incurrido en el entrenamiento y prueba fue considerablemente más alto que los obtenidos con los dos modelos tradicionales antes expuestos.

VI. RECOMENDACIONES

La clasificación de mensajes extorsivos puede escalar a mensajes delictivos; en este trabajo se hizo énfasis en la caracterización de mensajes extorsivos, los mismos que contenían palabras mal escritas y argot de los extorsionadores; pero los delitos tienen una gran variedad de modalidades como estafas, ingeniería social, acoso, entre otros; por lo que podemos escalar la solución obtenida para estos nuevos problemas.

No solamente se requiere clasificar, muchas veces se requiere entender lo que se dice en estos textos. Por lo que el análisis semántico será una oportunidad para elaborar otras investigaciones. La extracción de características puede potenciarse con nuevos algoritmos. Se puede experimentar con nuevos modelos cada vez más sofisticados, configurando sus parámetros y su arquitectura.

Este trabajo puede servir como guía metodológica a efectos de hacer fine-tuning con otros corpus específicos como por ejemplo terminología propia de algunos deportes, videojuegos, cuestiones legales, de gestión, etc.

La clasificación de mensajes de textos es una tarea importante que puede ser incorporada en muchos agentes inteligentes, útiles en muchas organizaciones porque asiste a la toma de decisiones.

Una compañía de telecomunicaciones o un operador puede tomar los resultados de esta investigación e incorporarlo en aplicaciones que requieran hacer este tipo de clasificación a afectos asistirlos en la toma de decisiones; por ejemplo, productivizar este algoritmo en una aplicación móvil capaz de reconocer mensajes extorsivos y proceder a bloquear al remitente o a reportar a alguna central de denuncias, aumentará la satisfacción de los usuarios respecto a la percepción de seguridad ciudadana.

Para poder tener más mensajes se puede usar data augmentation; esto permite que algunas inteligencias artificiales modifiquen los datos, lo que permite dotar de una mayor variedad de mensajes.

VII. AGRADECIMIENTOS O FINANCIAMIENTO

Deseo expresar mi agradecimiento al Dr Abel Coronado, quien con sus observaciones ha contribuido a esta investigación; la presente investigación es financiado en su totalidad por el autor.

VIII. REFERENCIAS

[1] K. Guan et al., “IEEE Access Special Section Editorial: 5G and beyond mobile wireless communications enabling intelligent mobility,” IEEE Access, vol. 8, pp. 208892–208897, 2020, doi: 10.1109/ACCESS.2020.3037635.
[2] V. dos Santos Riedo and S. A. Carro, “Um modelo para extração, estruturação, indexação e recuperação de casos clínicos publicados na web,” Colloquium Exactarum, vol. 11, no. 2, pp. 91–100, 2019, doi: 10.5747/ce.2019.v11.n2.e278.
[3] E. Elsaeed, O. Ouda, M. M. Elmogy, A. Atwan, and E. El-Daydamony, “Detecting fake news in social media using voting classifier,” IEEE Access, vol. 9, pp. 161909–161925, 2021, doi: 10.1109/ACCESS.2021.3132022.
[4] M. Callejas Cuervo and M. A. Vélez Guerrero, “Predicción electoral usando un modelo híbrido basado en análisis sentimental y seguimiento a encuestas: elecciones presidenciales de Colombia,” Revista Politécnica, vol. 15, no. 30, pp. 94–104, 2019, doi: 10.33571/rpolitec.v15n30a9.
[5] R. Anggrainingsih, G. M. Hassan, and A. Datta, “CE-BERT: Concise and efficient BERT-based model for detecting rumors on Twitter,” IEEE Access, vol. 11, pp. 80207–80217, 2023, doi: 10.1109/ACCESS.2023.3299858.
[6] S. H. Al Sulaimani and A. Starkey, “Short text classification using contextual analysis,” IEEE Access, vol. 9, pp. 149619–149629, 2021, doi: 10.1109/ACCESS.2021.3125768.
[7] Z. E. Rasjid and R. Setiawan, “Performance comparison and optimization of text document classification using k-NN and Naïve Bayes classification techniques,” Procedia Comput. Sci., vol. 116, pp. 107–112, 2017, doi: 10.1016/j.procs.2017.10.017.
[8] R. Ahamad and K. N. Mishra, “Exploring sentiment analysis in handwritten and E-text documents using advanced machine learning techniques: A novel approach,” J. Big Data, vol. 12, no. 1, Art. no. 11, 2025, doi: 10.1186/s40537-025-01064-2.
[9] J. R. Davalos-Miranda, “Análisis del Sentimiento: Exploración de League of Legends y su comunidad en Twitter,” M.S. thesis, Universidad Internacional de La Rioja (UNIR), 2022.
[10] C. Simón-Gallego, “Comparativa de modelos de aprendizaje profundo para la detección de odio en castellano en medios de información social,” M.S. thesis, Universidad Internacional de La Rioja (UNIR), 2023.
[11] K. Taneja, J. Vashishtha, and S. Ratnoo, “Fraud-BERT: transformer based context aware online recruitment fraud detection,” Discover Computing, vol. 28, Art. no. 9, 2025, doi: 10.1007/s10791-025-09502-8.
[12] I. Belzunegui-Gabilondo, “Creación de un clasificador de sentimientos de alumnos mediante minería de opinión,” M.S. thesis, Universidad Internacional de La Rioja (UNIR), 2023.
[13] M. E. Mswahili, J. Hwang, J. C. Rajapakse, K. Jo, and Y.-S. Jeong, “Positional embeddings and zero-shot learning using BERT for molecular-property prediction,” J. Cheminform., vol. 17, no. 1, Art. no. 17, 2025, doi: 10.1186/s13321-025-00959-9.
[14] Y. Yang and X. Peng, “BERT-based network for intrusion detection system,” EURASIP J. Inf. Secur., vol. 2025, no. 1, Art. no. 11, 2025, doi: 10.1186/s13635-025-00191-w.
[15] N. M. Gardazi, A. Daud, M. K. Malik, A. Bukhari, T. Alsahfi, and B. Alshemaimri, “BERT applications in natural language processing: a review,” Artif. Intell. Rev., vol. 58, no. 6, Art. no. 166, 2025, doi: 10.1007/s10462-025-11162-5.
[16] M. Maktabdar Oghaz, L. Babu Saheer, K. Dhame, and G. Singaram, “Detection and classification of ChatGPT-generated content using deep transformer models,” Front. Artif. Intell., vol. 8, Art. no. 1458707, 2025, doi: 10.3389/frai.2025.1458707.
[17] C. Samper-Quinto, “Análisis de sentimiento en procesamiento cuántico del lenguaje natural en un contexto financiero,” M.S. thesis, Universidad Internacional de La Rioja (UNIR), 2023.
[18] O. Elizarraras and J. Chakraborty, “The use of social media in generating health information support for Spanish users,” Procedia Comput. Sci., vol. 256, pp. 1183–1191, 2025, doi: 10.1016/j.procs.2025.02.227.

[19] K. C. Argomedo Pflücker and A. T. Cóndor Ruiz, “Diseño de un sistema inteligente utilizando procesamiento de lenguaje natural (PLN) para el reconocimiento de mensajes extorsivos,” B.S. thesis, Universidad Nacional de Trujillo, Trujillo, Peru, 2015.
[20] C. M. Liapis, A. Karanikola, and S. Kotsiantis, “Enhancing sentiment analysis with distributional emotion embeddings,” Neurocomputing, vol. 634, Art. no. 129822, 2025, doi: 10.1016/j.neucom.2025.129822.