IDENTIFICACIÓN DE ATAQUES DE DENEGACIÓN DE SERVICIO DISTRIBUIDO (DDOS) MEDIANTE LA INTEGRACIÓN DE ALGORITMOS DE APRENDIZAJE AUTOMÁTICO Y ARQUITECTURAS DE REDES NEURONALES ARTIFICIALES

Identification of distributed denial of service (DDOS) attacks by integrating machine learning algorithms and artificial neural network architectures

VÍCTOR A. GUZMAN-BRAND[1], LAURA E. GELVEZ-GARCÍA[2]

Recibido: 05 de enero de 2025. Aceptado: 15 de enero de 2025

DOI: https://doi.org/10.21017/rimci.1116

Resumen

El objetivo es identificar los ataques de denegación de servicio distribuido (DDoS) mediante la integración de algoritmos de aprendizaje automático y arquitecturas de redes neuronales artificiales. Metodología: Para estructurar el análisis de datos, se emplea la técnica de Conocimiento del Descubrimiento de Datos (KDD). Este enfoque permite examinar grandes volúmenes de información de diversos tipos, con el objetivo de identificar patrones, correlaciones y producir información valiosa. En cuanto al data set se emplea el conjunto de datos CIC-DDoS2019 desarrollado por el Instituto Canadiense de Ciberseguridad. Resultados: Al entrenar y evaluar los diferentes algoritmos, se observó que los modelos basados en árboles de decisión, como Random Forest y XGBoost, destacaron por alcanzar los mejores resultados en términos de precisión y eficiencia. Por otro lado, en el análisis del desempeño de las redes neuronales las Unidades de Corrientes Cerradas (GRU) sobresalieron al obtener los mejores resultados en exactitud y precisión. Este rendimiento sugiere que las GRU logran un equilibrio óptimo entre la capacidad predictiva y la minimización de falsos positivos y negativos. Discusiones: En la comparación entre modelos tradicionales de aprendizaje automático y redes neuronales para la detección de ataques DDoS, se observa que algoritmos como XGBoost y Random Forest ofrecen un rendimiento similar o superior en términos de precisión y, además, presentan tiempos de ejecución significativamente menores. Por otro lado, redes neuronales como GRU y RNN alcanzan una alta precisión, pero con un alto costo computacional. Conclusiones: XGBoost, demostró un equilibrio óptimo entre precisión (F1-score: 0.9992) y velocidad (11.47s), posicionándose como la alternativa más viable para implementaciones en tiempo real. En el ámbito de las redes neuronales, las Unidades de Corrientes Cerradas (GRU) obtuvieron el mejor rendimiento (accuracy: 0.9992; F1-score: 0.9992), dado la capacidad para procesar dependencias temporales y reducir falsos positivos.

Palabras Clave: Identificar; ataques de denegación de servicio distribuido; DDoS; aprendizaje automático; redes neuronales artificiales.

1 Profesional en Psicología. Especialista en Desarrollo Integral de la Infancia y Adolescencia. Especialista en analítica de datos. Estudiante deIngeniería en sistemas. Corporación Unificada Nacional de Educación Superior (CUN). ORCID: https://orcid.org/0000-0002-6051-3153 Correo electrónico: victora.guzman@cun.edu.co

2 Licenciada en Lengua Castellana. Magíster en Lingüística Española. Doctora en Ciencias de la educación. ORCID: https://orcid.org/0000-0003-0164-2972 Correo electrónico: laura_gelvez@cun.edu.co

Abstract

The Objective is to identify distributed denial of service (DDoS) attacks by integrating machine learning algorithms and artificial neural network architectures. Methodology: To structure the data analysis, the Knowledge Discovery Data (KDD) technique is used. This approach allows examining large volumes of information of various types, with the objective of identifying patterns, correlations and producing valuable information. As for the data set, the CIC-DDoS2019 dataset developed by the Canadian Cybersecurity Institute is used. Results: When training and evaluating the different algorithms, it was observed that the models based on decision trees, such as Random Forest and XGBoost, stood out for achieving the best results in terms of accuracy and efficiency. On the other hand, in the analysis of the performance of the neural networks, the Closed Stream Units (GRU) stood out by obtaining the best results in accuracy and precision. This performance suggests that GRUs achieve an optimal balance between predictive ability and minimization of false positives and negatives. Discussion: In the comparison between traditional machine learning models and neural networks for DDoS attack detection, it is observed that algorithms such as XGBoost and Random Forest offer similar or superior performance in terms of accuracy and exhibit significantly shorter execution times. On the other hand, neural networks such as GRU and RNN achieve high accuracy, but with a high computational cost. Conclusions: XGBoost, demonstrated an optimal balance between accuracy (F1-score: 0.9992) and speed (11.47s), positioning itself as the most viable alternative for real-time implementations. In the field of neural networks, Gated Stream Units (GCU) obtained the best performance (accuracy: 0.9992; F1-score: 0.9992), given the ability to process temporal dependencies and reduce false positives.

Keywords: Identify; distributed denial of service attacks; DDoS; machine learning; artificial neural networks.

I. Introducción

Los ataques de Denegación de Servicio Distribuido (DDoS) representan una grave amenaza para la estabilidad de la red debido a la enorme asimetría de recursos entre los atacantes y la víctima[1]. A diferencia de los ataques de Denegación de Servicio (DoS), que se originan desde una única fuente, los ataques de tipo DDoS son llevados a cabo de manera simultánea y coordinada desde múltiples ubicaciones geográficas[2]. Esta distribución permite a los atacantes generar un tráfico masivo con el propósito de interrumpir el funcionamiento normal de un sistema o red objetivo, lo que complica significativamente su detección y mitigación[3].

Los ataques de Denegación de Servicio Distribuido (DDoS) son relativamente sencillos de ejecutar, pero su detección y mitigación representan un desafío significativo. Para su implementación, los ciberdelincuentes suelen emplear botnets, es decir, redes de dispositivos comprometidos que generan tráfico malicioso de manera coordinada. Los ataques DDoS más frecuentes logran evadir los mecanismos de detección y aumentar su efectividad mediante la combinación de múltiples tácticas y escenarios de ataque dirigidos a objetivos específicos[4].

Según el nivel del protocolo afectado, los ataques DDoS se clasifican en dos categorías principales. En primer lugar, los ataques de inundación en la capa de transporte o red explotan paquetes TCP, UDP, ICMP y DNS para saturar el ancho de banda y afectar la conectividad en redes basadas en SDN. En segundo lugar, los ataques de inundación en la capa de aplicación comprometen la disponibilidad de servicios en línea al agotar los recursos del servidor que los aloja, impidiendo el acceso legítimo de los usuarios[5].

La aplicación de modelos de aprendizaje automático en la detección de ataques de Denegación de Servicio Distribuido (DDoS) representa una oportunidad clave para optimizar su identificación, mejorar la precisión de los sistemas de seguridad y agilizar la respuesta ante estas amenazas[6]. Entre los antecedentes sobre el tema están[7],[8],[9], donde se comparan algoritmos de aprendizaje automático y profundo para la detección del ataque. Asimismo, se encuentra estudios que han aplicado modelos como KNN[10], Decision Tree[11],[12], XGBoost[13], Random Forest[14], red neuronal artificial (ANN)[15],[16], en busca de los mejores resultados.

El presente proyecto investigativo tiene como objetivo identificar los ataques de denegación de servicio distribuido (DDoS) mediante la integración de algoritmos de aprendizaje automático y arquitecturas de redes neuronales artificiales, empleando el data set el conjunto de datos CIC-DDoS2019, desarrollado por el Instituto Canadiense de Ciberseguridad y planificado dentro del proceso del Conocimiento del Descubrimiento de Datos (KDD).

II. Metodología

Para estructurar el análisis de datos, se utiliza el descubrimiento de datos, un proceso que se enmarca dentro del Conocimiento del Descubrimiento de Datos (KDD). Este enfoque permite examinar grandes volúmenes de información de diversos tipos, con el objetivo de identificar patrones, correlaciones y otros insights relevantes[17],[18]. Este proceso abarca varias etapas clave: la selección de los datos, su limpieza para eliminar inconsistencias, la transformación y proyección de los datos, la extracción de patrones y modelos significativos, y finalmente, la evaluación e interpretación de estos patrones para transformarlos en conocimiento útil[19].

Fase uno: selección de los datos

Se ha seleccionado el conjunto de datos CIC-DDoS2019, desarrollado por el Instituto Canadiense de Ciberseguridad. Este dataset está compuesto por 88 características y 7 registros, los cuales incluyen tanto tráfico de red normal como diversos tipos de ataques de denegación de servicio distribuido (DDoS). Su principal aplicación es la clasificación, permitiendo diferenciar y reconocer múltiples tipos de ataques en contraste con el tráfico legítimo[16]. Además, el CIC-DDoS2019 cubre 18 tipos de ataques, entre los que se destacan aquellos basados en reflexión y explotación, lo que lo convierte en una herramienta valiosa para la investigación en ciberseguridad[20].

Fase dos: preprocesamiento de datos

La clasificación es un proceso fundamental en el análisis de datos que consiste en identificar un modelo capaz de describir clases o conceptos dentro de un conjunto de datos. El objetivo principal de este modelo es predecir la clase de objetos cuyas etiquetas son desconocidas, basándose en el análisis de conjuntos de datos de entrenamiento[21]. Sin embargo, no toda la información almacenada es útil para construir el modelo; por lo tanto, es crucial identificar y extraer el subconjunto relevante de datos que contribuya de manera significativa a la elaboración del modelo[22].

Con el objetivo de asegurar la calidad y relevancia de los datos, se implementa un proceso de preprocesamiento estructurado que abarca las siguientes etapas:

- Identificación y eliminación de datos nulos o faltantes: Se detectan y eliminan los valores ausentes en la base de datos para evitar sesgos o errores en el análisis.

- Organización de etiquetas: Se eliminan los espacios en blanco en las etiquetas para asegurar la consistencia en el formato de los datos.

- Conversión de valores categóricos a numéricos: Utilizando la clase LabelEncoder, se transforman las variables categóricas en valores numéricos, lo que facilita su procesamiento y reduce el consumo de memoria en comparación con el uso de cadenas de texto.
- Equilibrio de clases: Se aplica la técnica SMOTE (Synthetic Minority Oversampling Technique), la cual genera muestras sintéticas de la clase minoritaria para equilibrar la distribución de las clases. Esto evita el sesgo hacia la clase mayoritaria y mejora el rendimiento del modelo.

- Normalización de características: Las características de los datos se escalan a un rango común para asegurar que todas contribuyan de manera equitativa al modelo.

- Eliminación de características redundantes o irrelevantes: Se identifican y eliminan aquellas características que no aportan información valiosa o que están altamente correlacionadas con otras, optimizando así el conjunto de datos.

Fase tres: transformación de datos

En esta etapa, se lleva a cabo un proceso de validación cruzada con el objetivo de determinar el número óptimo de características (k) para el modelo. Para ello, se emplea un clasificador de regresión logística, el cual permite evaluar el rendimiento del modelo con distintos valores de k y seleccionar el más adecuado. Tras este análisis, se estima que el valor óptimo de k es 65, obteniendo una puntuación de rendimiento de 0.9960. Posteriormente, se identifican las características más relevantes mediante un criterio de puntuación

estadística, utilizando en este caso la prueba ANOVA F-statistic, que permite seleccionar aquellas variables que contribuyen de manera significativa al problema de clasificación.

Además, se aplica el análisis de componentes principales (PCA), una técnica estadística versátil que reduce una matriz de datos organizada por casos y variables a sus componentes principales, también denominados elementos esenciales[23]. Esta técnica destaca por su eficacia para simplificar la complejidad de conjuntos de datos con múltiples variables, ya que identifica las direcciones principales de variabilidad y facilita una comprensión más clara de la estructura subyacente de los datos[24]. Como resultado de este proceso, se reduce la dimensionalidad del conjunto de datos a 25 componentes principales, logrando una varianza acumulada explicada del 95%, como se evidencia en la Fig. 1.

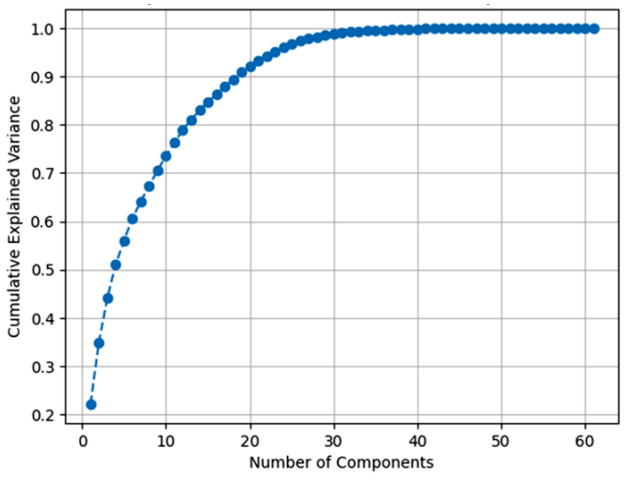

Fig. 1. Varianza acumulada explicada vs. Numero de componentes.

Nota: en la gráfica se aprecia el comportamiento de la varianza acumulada explicada cuando comienza a estabilizarse. Este punto se conoce como el «codo» en la curva, donde agregar más componentes no aporta una mejora significativa en la varianza explicada. Elaboración propia.

III. Resultados

Fase cuatro: minería de datos

A partir de la matriz de correlación del conjunto de datos CIC-DDoS2019, se pueden extraer importantes conclusiones sobre cómo se relacionan las diferentes variables entre sí y con la ocurrencia de ataques de denegación de servicio (DDoS). Esta técnica es una herramienta esencial para identificar lineales entre las variables y, en este caso, permite detectar patrones que pueden ser muy relevantes para la detección y clasificación de eventos anómalos o maliciosos[25], la cual se presenta en la Fig. 2.

Muchas variables que miden el tráfico, como el número de paquetes enviados, la cantidad de bytes transmitidos y la duración del flujo, presentan correlaciones positivas elevadas. Esto se debe a que, en un ataque DDoS, es común que ocurran ráfagas de tráfico donde tanto el volumen de datos como la cantidad de paquetes aumentan simultáneamente. Además, algunas variables capturan matices distintos del comportamiento del tráfico, lo que puede generar correlaciones moderadas o incluso negativas entre flujos entrantes y salientes, dependiendo de la arquitectura de la red y la naturaleza del ataque. Estas relaciones inversas pueden ser clave para identificar patrones atípicos y distinguir entre tráfico legítimo y malicioso.

De tal manera que, la variable que indica la ocurrencia del ataque suele estar altamente correlacionada con métricas que reflejan el incremento del tráfico, como el número de paquetes y la cantidad de bytes transmitidos. Este comportamiento refuerza la idea de que los ataques DDoS generan cambios significativos en el tráfico de la red. Aprovechar estas correlaciones permite desarrollar modelos de detección basados en aprendizaje automático, los cuales pueden identificar anomalías y mejorar la capacidad de respuesta ante este tipo de amenazas.

Por otro lado, en esta etapa se lleva a cabo el proceso de experimentación, en el cual se emplean diversos algoritmos de aprendizaje automático para abordar el problema de clasificación binaria. A continuación, se describen los algoritmos utilizados y sus fundamentos teóricos:

- Random Forest: Este algoritmo, basado en técnicas de aprendizaje supervisado, construye múltiples árboles de decisión durante el entrenamiento[26]. Los resultados obtenidos de cada árbol se integran para construir un modelo único, más robusto y preciso que cualquiera de los árboles individuales[27].

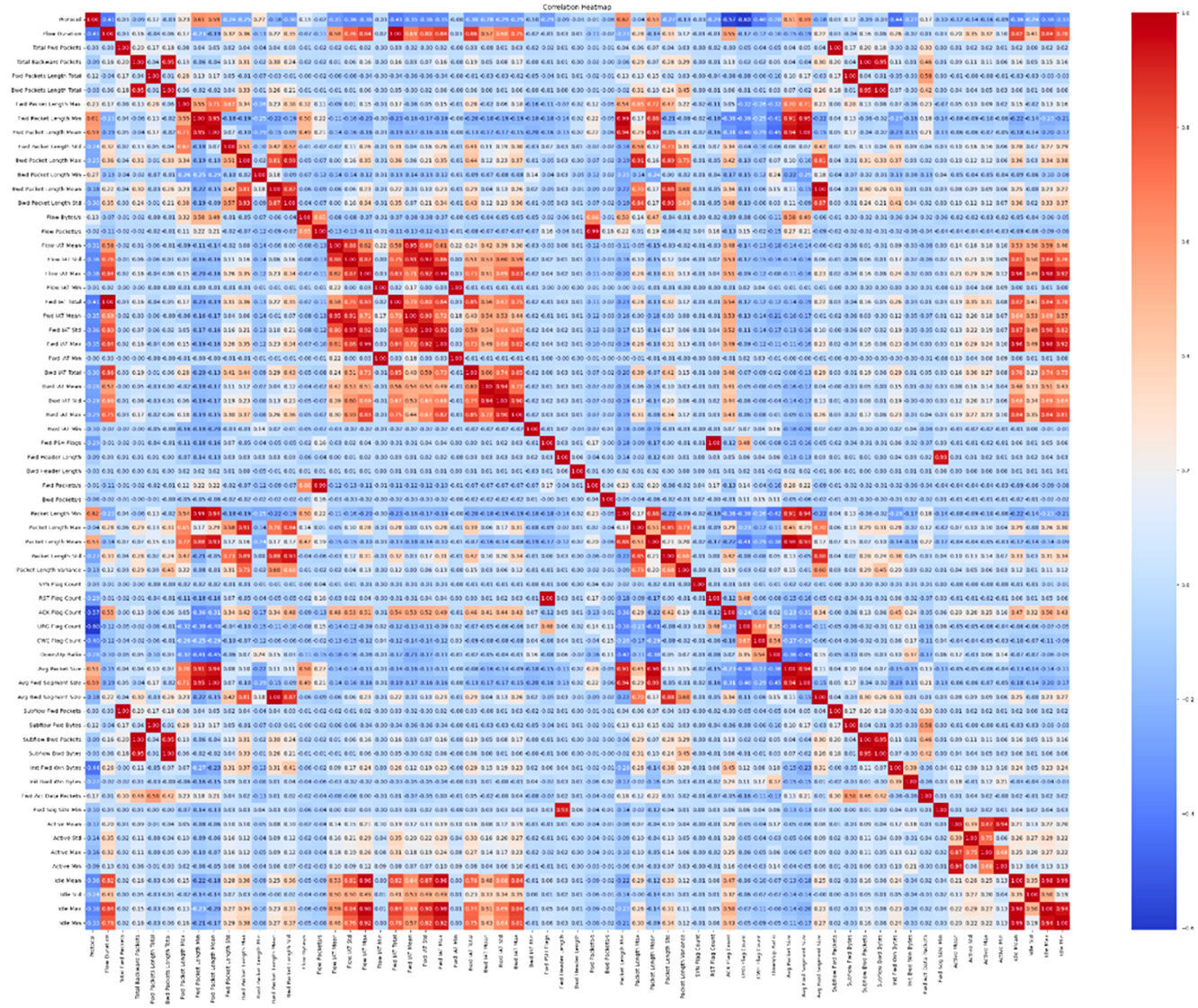

Fig. 2. Matriz de correlación de las variables.

Nota: una matriz de evaluación es una herramienta que muestra la relación lineal entre variables en un conjunto de datos. Cada valor representa el coeficiente de valoración, que varía entre -1 y 1. Elaboración propia.

- XGBoost: Es un método de ensamblado secuencial que agrega árboles de decisión de manera iterativa. Cada nuevo árbol aprende de los errores cometidos por los anteriores, corrigiéndolos progresivamente hasta que no es posible reducir más el error[28].

- Regresión Logística: Este algoritmo se enfoca en estimar la relación entre variables de entrada y una variable de salida binaria o continua. Su objetivo es modelar la probabilidad de que una observación pertenezca a una clase específica[29].

- K-Nearest Neighbors (KNN): Basado en la similitud entre instancias, este algoritmo clasifica los datos según la mayoría de las clases de los k vecinos más cercanos. Es una generalización de las reglas del vecino más cercano y ofrece un rendimiento competitivo en ciertos contextos[30].

- Árboles de Decisión: Este método utiliza pruebas estadísticas para realizar predicciones basadas en observaciones y construcciones lógicas. El árbol toma decisiones óptimas desde un enfoque probabilístico, dividien-

do los datos en función de las características más informativas[31].

- Naive Bayes: Este clasificador se basa en el teorema de Bayes, que describe la relación entre probabilidades condicionales. En el contexto de clasificación, se busca estimar la probabilidad de que una observación pertenezca a una clase dada sus características observadas[32].

- Gradient Boosting: Es una técnica que combina múltiples modelos débiles (generalmente árboles de decisión) para construir un modelo fuerte. Esto se logra mediante la optimización secuencial del error residual, mejorando iterativamente el rendimiento del modelo[33]. Tabla I.

Asimismo, en esta fase del proceso de experimentación, se emplean diversas redes neuronales artificiales (RNA), cuyo objetivo principal es simular o emular el proceso de aprendizaje del cerebro humano[34]. En estas redes, el aprendizaje se lleva a cabo mediante la ponderación de conexiones entre neuronas. Cada conexión tiene un peso asociado, el cual se ajusta durante el entrenamiento para optimizar el rendimiento del modelo. Estos pesos se suman y, en función de un umbral predefinido, generan la activación de un nodo, también conocido como unidad lógica de umbral[35]. Entre las redes se encuentran:

El Perceptrón Multicapa (MLP) se caracteriza por organizar sus neuronas en capas de diferentes niveles. Estas capas incluyen:

Tabla I. Características generales de los algoritmos

Algoritmo	Hiperparámetros por Defecto	Características
Random Forest	• n_estimators=100. • criterion='gini' . • max_depth=None. • min_samples_split=2.	Basado en árboles de decisión, robusto a overfitting, maneja bien datos no lineales.
XGBoost	• n_estimators=100 • max_depth=3 • learning_rate=0.1 • objective='binary:logistic'.	Algoritmo de boosting, eficiente y preciso, maneja datos desbalanceados.
Logistic Regression	• penalty='l2' • C=1.0 • solver='lbfgs' • max_iter=100.	Modelo lineal, simple y rápido, ideal para problemas de clasificación binaria.
K-Nearest Neighbors	• n_neighbors=5 • weights='uniform' • algorithm='auto' • leaf_size=30.	Basado en instancias, simple pero puede ser lento con grandes datasets.
Decision Tree	• criterion='gini' • splitter='best' • max_depth=None • min_samples_split=2	Modelo simple, propenso a overfitting si no se regulariza.
Naive Bayes	• priors=None • var_smoothing=1e-9	Basado en probabilidades, rápido y eficiente para datasets grandes.
Gradient Boosting	• n_estimators=100 • learning_rate=0.1 • max_depth=3 • loss='deviance'.	Algoritmo de boosting, similar a XGBoost pero con implementación diferente.

Nota: elaboración propia.

- Capa de entrada: Recibe los datos iniciales.
- Capas ocultas: Realizan transformaciones no lineales sobre los datos de entrada.
- Capa de salida: Genera la predicción o clasificación final.

Cada capa está compuesta por un conjunto de neuronas que procesan la información de manera secuencial, permitiendo que el modelo aprenda representaciones jerárquicas de los datos[36].

Las Redes Neuronales Convolucionales (CNN) constituyen un tipo de red neuronal artificial que integra capas convolucionales, capas de agrupación (pooling) y capas completamente conectadas para el procesamiento eficiente de datos estructurados, especialmente en tareas de visión por computadora. Estas capas trabajan de manera conjunta para transformar datos sin procesar en representaciones visualmente significativas[37]. Las capas convolucionales y de agrupación analizan la entrada para extraer características relevantes, generando mapas de características. La capa completamente conectada recibe estas características y las clasifica, permitiendo la identificación de patrones complejos en los datos[38].

Las Redes Neuronales Recurrentes (RNN) son redes dinámicas que incorporan realimentación (feedback) en su arquitectura. Esto significa que las salidas de un paso temporal se utilizan como entradas para el siguiente, permitiendo que la red capture dependencias temporales en secuencias de datos[39]. Esta característica las hace ideales para aplicaciones como la predicción de series temporales y la simulación de sistemas no lineales, ya que pueden procesar secuencias de longitud variable y mantener un «estado» interno que refleja información pasada[40].

Las Unidades de Corrientes Cerradas (GRU) son una variante de las RNN que utilizan un mecanismo de compuertas para gestionar el flujo de información entre las células de la red. Este mecanismo permite capturar dependencias a largo plazo en conjuntos de datos secuenciales sin descartar información relevante del pasado. Las GRU son más eficientes computacionalmente que otras arquitecturas recurrentes, como las LSTM (Long Short-Term Memory), y se utilizan ampliamente en tareas que requieren el procesamiento de secuencias, como el modelado de lenguaje y la predicción temporal[41].

Las Redes Neuronales Profundas (DNN) son modelos de aprendizaje automático compuestos por múltiples capas de neuronas artificiales. Cada capa transforma los datos de entrada mediante funciones de activación no lineales, permitiendo que la red aprenda representaciones jerárquicas y complejas de los datos[42]. Estas redes son especialmente efectivas en tareas que involucran datos de alta dimensionalidad, como el reconocimiento de imágenes, el procesamiento de lenguaje natural y la predicción basada en datos estructurados[43]. Tabla II.

Fase cinco: evaluación de modelos

Para el entrenamiento de los modelos, se utilizó la plataforma Google Colab junto con el lenguaje de programación Python, el cual cuenta con diversas librerías especializadas en análisis de datos, visualización y aprendizaje automático, tales como Pandas, NumPy, Matplotlib, Seaborn, Keras y TensorFlow, entre otras. El objetivo de los modelos es la identificación binaria de ataques de denegación de servicio distribuido (DDoS). Para ello, el conjunto de datos se divide en un 70 % para entrenamiento y un 30 % para pruebas. Posteriormente, el desempeño de los modelos se evalúa mediante métricas generales, obteniéndose los siguientes resultados. Tabla III.

El análisis del desempeño de los algoritmos de aprendizaje automático en la detección de ataques DDoS revela que los modelos basados en árboles de decisión, como *Random Forest* y *XGBoost*, obtienen los mejores resultados en términos de precisión y eficiencia. *Random Forest* alcanza un *accuracy* (0.9993) y un *F1-score* (0.9993) con un tiempo de ejecución moderado de 659.77s, lo que las convierte en una opción sólida y eficiente para la detección de ataques. Por otro lado, *XGBoost* muestra un desempeño casi idéntico con un *F1-score* (0.9992), pero con una ejecución significativamente más rápida (11.47s), lo que lo convierte en la opción más eficiente en términos de velocidad sin sacrificar precisión.

El rendimiento de *K-Nearest Neighbors (KNN)* y *Decision Tree* también es alto, con *F1-scores* de 0.9989

Tabla II. Parámetros generales de las redes neuronales

Parámetros	MLP	CNN	RNN	GRU	DNN
Arquitectura	Dense (128, 64, 32)	Conv1D (64) + MaxPooling1D + Dense (128)	LSTM (128, 64) + Dense (64)	GRU (128, 64) + Dense (64)	Dense (256,
Función de Activación	ReLU	ReLU	ReLU	ReLU	ReLU
Capa de salida	Sigmoid	Sigmoid	Sigmoid	Sigmoid	Sigmoid
Dropout	0.3, 0.2, 0.1	0.3	0.3	0.3	0.4, 0.3, 0.2
Batch Normalization	No	No	No	No	Si
Optimizador	Adam	Adam	Adam	Adam	Adam
Tasa de Aprendizaje	0.001	0.001	0.001	0.001	0.001
Función de Pérdida	Binary Crossentropy	Binary Crossentropy	Binary Crossentropy	Binary Crossentropy	Binary Crossentropy
Regularización	L2 (0.001)	No	No	No	L2 (0.001)
Épocas	100	100	100	100	100
Batch Size	64	64	64	64	64

Nota: elaboración propia.

Tabla III. Métricas de evaluación de los modelos

Algoritmo	Accuracy	Precision	Recall	F1 Score	Tiempo de ejecución
Random Forest	0.9993	0.9989	0.9997	0.9993	659.77s
XGBoot	0.9992	0.9990	0.9995	0.9992	11.47s
Logistic Regression	0.9949	0.9943	0.9955	0.9949	5.95s
K-Nearest Neighbors	0.9989	0.9987	0.9992	0.9989	500.69s
Decision Tree	0.9987	0.9986	0.9988	0.9987	99.59s
Naive Bayes	0.6629	0.8437	0.4012	0.5438	0.83s
Gradient Boosting	0.9968	0.9950	0.9986	0.9968	1126.48s
Redes Neuronales					
Algoritmo	**Accuracy**	**Precision**	**Recall**	**F1 Score**	**Tiempo de ejecución**
MLP	0.9988	0.9977	0.9999	0.9988	2356.78s
CNN	0.9987	0.9976	0.9998	0.9987	7198.51s
RNN	0.9990	0.9983	0.9997	0.9990	8400.25s
GRU	0.9992	0.9986	0.9998	0.9992	7300.25s
DNN	0.9986	0.9972	0.9999	0.9986	6700.56s

Nota: elaboración propia.

y 0.9987, respectivamente, aunque KNN tiene un tiempo de ejecución elevado (500.69s), lo que puede ser una limitante en aplicaciones en tiempo real. En cuanto a *Naive Bayes*, su bajo *accuracy* (0.6629) y *F1-score* (0.5438) indican que no es adecuado para este problema, ya que su simplicidad no logra capturar correctamente las relaciones en los datos. *Gradient Boosting* obtiene un buen desempeño (*F1-score* de 0.9968), pero su tiempo de ejecución (1126.48s) es significativamente mayor que el de *XGBoost*, lo que lo hace menos competitivo.

Por otro lado, el análisis del desempeño de las redes neuronales en la detección de ataques DDoS muestra que todas las arquitecturas presentan una alta precisión y capacidad de detección, con valores de *accuracy* superiores al 99.86% y un *F1-score* cercano a 1.0. Entre ellas, las Unidades de Corrientes Cerradas (GRU) obtuvo los mejores resultados en términos de *accuracy* (0.9992), *precision* (0.9986) y *F1-score* (0.9992), lo que sugiere un excelente equilibrio entre la capacidad de predicción y la reducción de falsos positivos y negativos.

La Red Neuronal Recurrente (RNN) también mostró un desempeño destacado con un *F1-score* de 0.9990, aunque ligeramente inferior al de GRU. Sin embargo, el tiempo de ejecución de ambas redes (8400.25s para RNN y 7300.25s para GRU) es significativamente mayor en comparación con el Perceptrón Multicapa (MLP), que logró resultados competitivos con un *accuracy* (0.9988) en un tiempo mucho menor (2356.78s), lo que lo hace una opción más eficiente en términos computacionales.

En términos de eficiencia computacional, las redes convolucionales (CNN) y las redes profundas (DNN) también presentan tiempos de ejecución elevados, con 7198.51s y 6700.56s respectivamente, sin mejorar significativamente el rendimiento sobre MLP o GRU. Si bien CNN y DNN tienen una alta capacidad de generalización en tareas de clasificación complejas, su costo computacional puede ser un factor limitante en entornos donde la latencia y el procesamiento en tiempo real son críticos. En este contexto, MLP emerge como la mejor opción en términos de rapidez y efectividad, mientras que GRU se posiciona como la mejor alternativa si se prioriza la precisión sobre el tiempo de ejecución. Fig. 3.

Comparando los modelos tradicionales de aprendizaje automático con las redes neuronales en la detección de ataques DDoS, se observa que los algoritmos como XGBoost y Random Forest ofrecen un rendimiento similar o incluso superior

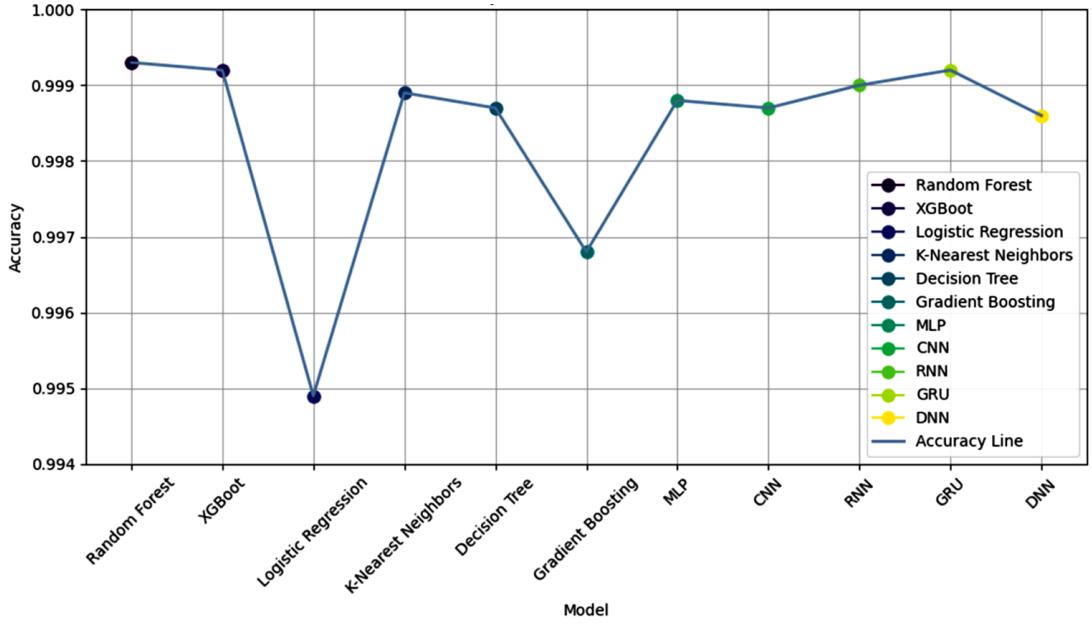

Fig. 3. Comparación de la exactitud de los diferentes modelos

Nota: Elaboración propia.

en términos de precisión (F1-score > 0.999) con tiempos de ejecución significativamente menores. Mientras que redes como GRU y RNN logran una alta precisión, su tiempo de ejecución supera las 7000s, lo que los hace menos viables para implementaciones en tiempo real. En contraste, XGBoost alcanza un F1-score (0.9992) en solo 11.47s, lo que lo convierte en la opción más eficiente sin sacrificar exactitud. Fig. 4.

Cada matriz representa el desempeño del modelo en función de cuatro métricas clave: verdaderos positivos (VP), verdaderos negativos (VN), falsos positivos (FP) y falsos negativos (FN). Es así como, los algoritmos Random Forest, XGBoost y Gradient Boosting se destacan por su equilibrio entre verdaderos positivos (VP) y verdaderos negativos (VN), reflejando un alto desempeño en la clasificación correcta de ambas clases. Sin embargo, modelos como Naive Bayes presentan una mayor cantidad de errores, particularmente falsos positivos (FP), lo que sugiere que podría no ser el más adecuado para el problema analizado. La regresión logística, por otro lado, muestra una ligera tendencia a clasificar incorrectamente instancias negativas como positivas, lo que puede influir negativamente en su precisión global. Fig. 5.

En general, todas las arquitecturas de las redes neuronales muestran un alto número de VP y VN, lo que indica un buen desempeño en la clasificación correcta de las instancias. Sin embargo, la diferencia radica en los valores de FP y FN, que varían ligeramente entre las arquitecturas. Por ejemplo, el modelo DNN presenta una menor cantidad de errores en comparación con otros modelos, lo que sugiere una mayor precisión y sensibilidad. Por otro lado, los modelos MLP y RNN presentan valores un poco más elevados en los FP, lo que podría afectar su capacidad de generalización.

La interpretación de estas matrices sugiere que las arquitecturas GRU y DNN sobresalen en términos de balance entre precisión y sensibilidad,

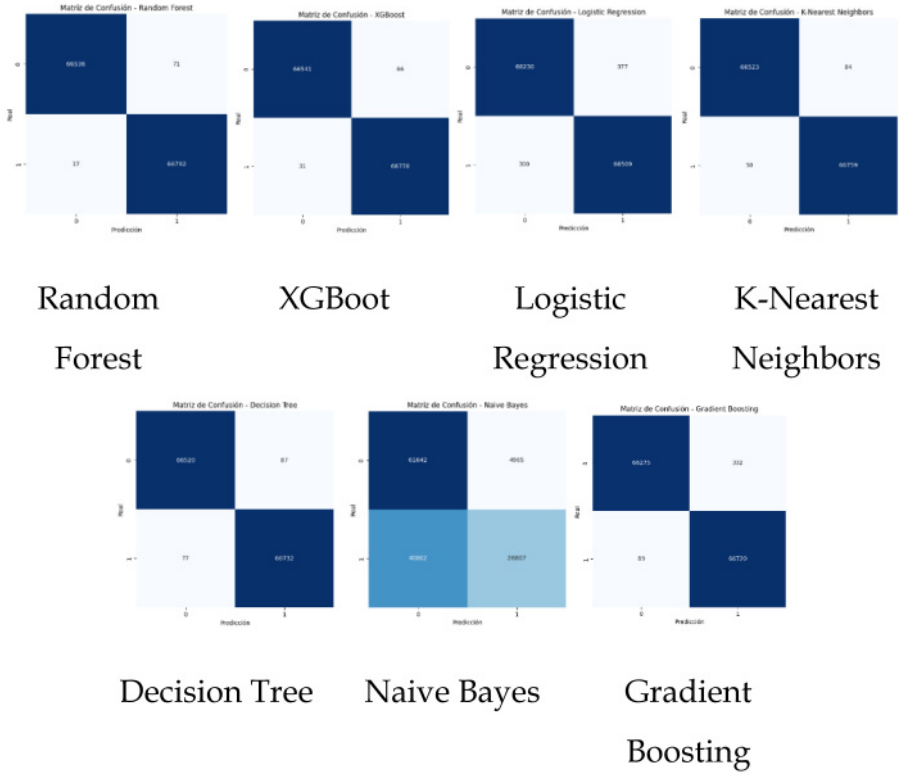

Fig. 4. Análisis de la aplicación de la matriz de confusión a los algoritmos

Nota: las matrices de confusión representan las predicciones acertadas en la diagonal principal, donde se encuentran los verdaderos positivos y negativos, mientras que los valores fuera de esta diagonal corresponden a los errores, es decir, los falsos positivos y negativos[44]. Elaboración propia.

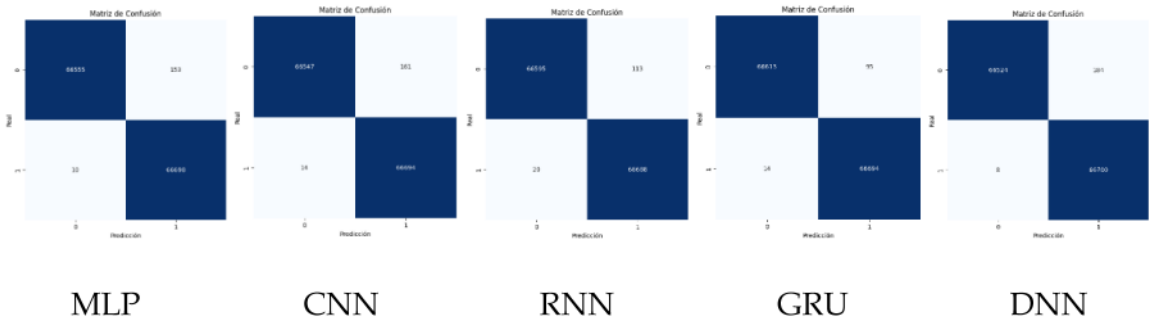

Fig. 5. Análisis de la aplicación de la matriz de confusión a las redes neuronales

Nota: Elaboración propia.

ya que minimizan errores y mantienen un alto rendimiento en la detección de clases. Esto podría deberse a su capacidad de capturar relaciones temporales o características complejas en los datos, lo que es especialmente importante en problemas de secuencias o datos con alta dimensionalidad.

IV. Discusiones

Al analizar el desempeño de los algoritmos tradicionales y redes neuronales en la detección de ataques de denegación de servicio distribuido (DDoS) empleando la base de datos CIC-DDoS-2019, se presenta que modelos como *XGBoost* es la mejor alternativa en términos de balance entre precisión y velocidad, mientras que *Random Forest* es una opción sólida si se prioriza la estabilidad del modelo sobre la rapidez. Estos hallazgos se relacionan a los encontrados en[45], donde menciona que *XGBoost* presenta una precisión alta (99.11%) en comparación a otros modelos de clasificación.

En cuanto al modelo *Random Forest* presenta una exactitud (0.9993) y un *F1-score* (0.9993) con un tiempo de ejecución moderado de 659.77s, lo que lo convierte en una opción robusta para la detección de ataques. Resultados acordes a los estudios de[46],[47],[48] donde muestra la comparativa este modelo destacando con valoración (99%) en la identificación de ataques en la clasificación binaria. Al igual,[49] indica que los algoritmos supervisados como *Decision Tree* y *Random Forest* tienen mayor desempeño (0.9999) en razón a algoritmos no supervisados y semi supervisados.

Al comparar las redes neuronales los resultados muestran como el modelo como Gated Recurrent Units (GRU) obtuvo los mejores resultados en términos de precisión (0.9992). La Red Neuronal Recurrente (RNN) también mostró un desempeño destacado con (0.9990), aunque ligeramente inferior al de GRU, pero esta arquitectura mostro un alto gasto computacional. Situación encontrada en el estudio de[50], donde la RNN es la que obtiene el mejor desempeño en la clasificación binaria (99.99%), aunque entre todas las redes analizadas en la que mayor tiempo de ejecución.

El modelo de Redes Neuronales Profundas (DNN) presenta una menor tasa de error en comparación con otros enfoques, lo que sugiere una mayor precisión y sensibilidad en la detección de anomalías. Este hallazgo coincide con lo reportado en[51], donde se destaca que las soluciones basadas en DNN superan a otros modelos en términos de precisión al analizar el tráfico de la red. Gracias a su capacidad para aprovechar arquitecturas avanzadas y algoritmos de aprendizaje profundo, las redes pueden identificar de manera eficiente patrones complejos y relaciones ocultas en los datos.

Al emplearse una rede neuronal convolucional esta muestra un buen desempeño (0.9987) en la tarea de clasificación. Sin embargo, con alta carga computacional[52], que podría ser una desventaja en aplicación en tiempo real. Estadísticas que se alinean con los hallazgos de[38], los cuales experimentan con varias redes neuronales convolucionales de este tipo y concluyen que tiene la capacidad de clasificación alta (99.7) en rangos de tiempo de ejecución entre 320 a 600 segundos. Pero la matriz de confusión presenta una mayor cantidad de errores, particularmente falsos positivos (FN).

Se emplea el análisis de componentes principales (PCA), una técnica estadística versátil que reduce una matriz de datos organizada por casos y variables a sus componentes principales. Se reduce la dimensionalidad del conjunto de datos a 25 componentes principales, logrando una varianza acumulada explicada del 95%. Cuando se emplean este tipo de técnicas se mejora el desempeño de los algoritmos como en el caso de[53], donde el modelo de red neuronal artificial (MLP) alcanza un desempeño (99.40%) en la clasificación de ataques.

V. Conclusiones

El estudio demuestra que la integración de algoritmos de aprendizaje automático y arquitecturas de redes neuronales artificiales (RNA) ofrece un enfoque robusto y eficaz para la detección de ataques de denegación de servicio distribuido (DDoS). A partir del análisis de la matriz de correlación del conjunto de datos CIC-DDoS2019, se identificaron relaciones significativas entre variables como el número de paquetes, bytes transmitidos y duración del flujo, las cuales presentan correlaciones positivas elevadas durante los ataques, confirmando su utilidad como indicadores clave.

En la experimentación con algoritmos clásicos de aprendizaje automático, Random Forest y XGBoost destacaron como los modelos más eficientes, alcanzando valores de accuracy y F1-score superiores al 99.9%. XGBoost, en particular, demostró un equilibrio óptimo entre precisión (F1-score: 0.9992) y velocidad (11.47s), posicionándose como la alternativa más viable para implementaciones en tiempo real. Por otro lado, Naive Bayes mostró un desempeño insuficiente (F1-score: 0.5438), subrayando la necesidad de modelos capaces de capturar relaciones no lineales en los datos.

En el ámbito de las redes neuronales, las Unidades de Corrientes Cerradas (GRU) obtuvieron el mejor rendimiento (accuracy: 0.9992; F1-score: 0.9992), gracias a su capacidad para procesar dependencias temporales y reducir falsos positivos. No obstante, su elevado tiempo de ejecución (7300.25s) contrasta con la eficiencia del Perceptrón Multicapa (MLP) (accuracy: 0.9988 en 2356.78s), que emerge como una solución balanceada para escenarios con restricciones computacionales. Las arquitecturas convolucionales (CNN) y profundas (DNN), aunque precisas, no superaron significativamente a modelos más simples, lo que sugiere que su complejidad no justifica su uso en este contexto específico.

La comparación entre métodos tradicionales y redes neuronales artificiales revela que XGBoost y Random Forest igualan o superan el rendimiento de las redes neuronales en precisión, pero con tiempos de entrenamiento hasta 650 veces menores. Esto resalta la ventaja de los algoritmos basados en árboles para aplicaciones que priorizan la latencia, como sistemas de detección en línea. Sin embargo, las redes neuronales particularmente GRU y RNN, muestran potencial en entornos donde la captura de patrones temporales o secuenciales es crítica.

Las matrices de confusión confirman que los modelos basados en árboles, como Random Forest y XGBoost, minimizan significativamente los errores de clasificación, tanto falsos positivos (FP) como falsos negativos (FN). Por otro lado, arquitecturas más complejas, como las Redes Neuronales Profundas (DNN) y las Unidades de Corrientes Cerradas (GRU), logran un equilibrio notable entre sensibilidad y especificidad, lo que las hace especialmente adecuadas para capturar patrones complejos en los datos. Sin embargo, la elección del modelo óptimo dependerá del balance entre la necesidad de una alta precisión y la disponibilidad de recursos computacionales, especialmente en escenarios que requieren detección en tiempo real de ataques DDoS.

Fuentes de financiamiento

Este estudio no recibió financiamiento de organismos públicos, entidades comerciales ni organizaciones sin fines de lucro.

Conflictos de interés

Los autores declaran que no existen conflictos de interés, ya sean financieros, profesionales o personales, que pudieran influir de manera indebida en los resultados obtenidos o en su interpretación.

Contribuciones de los autores

Autor 1: extraer, transformar, cargar (ETL), desarrollo de código, manejo del software, análi-

sis de datos, investigación, metodología, recursos, escritura, preparación del borrador original.

Autor 2: análisis formal, investigación, diseño metodológico, gestión del proyecto, asignación de recursos, validación, supervisión, redacción, revisión y edición.

Referencias

[1] W. Alhalabi, A. Gaurav, V. Arya, I. F. Zamzami, y R. A. Aboalela. «Machine Learning-Based Distributed Denial of Services (DDoS) Attack Detection in Intelligent Information Systems», *Httpsservicesigi-Glob.*, vol. 19, n.º 1, pp. 1-17, ene. 1d. C., doi: DOI: 10.4018/IJSWIS.327280.

[2] O. R. Sanchez, M. Repetto, A. Carrega, R. Bolla, y J. F. Pajo. «Feature Selection Evaluation towards a Lightweight Deep Learning DDoS Detector», en *ICC 2021-IEEE International Conference on Communications*, jun. 2021, pp. 1-6. doi: 10.1109/ICC42927.2021.9500458.

[3] M. Mittal, K. Kumar, y S. Behal. «Deep learning approaches for detecting DDoS attacks: a systematic review», *Soft Comput.*, vol. 27, n.º 18, pp. 13039-13075, sep. 2023, doi: 10.1007/s00500-021-06608-1.

[4] J. T. Mejía, M. I. Gonzales, A. del R. Fernández, y N. M. Crespo. «Seguridad contra ataques DDoS en los entornos SDN con Inteligencia Artificial», *Mag. Las Cienc. Rev. Investig. E Innov.*, vol. 7, n.º 3, pp. 105-127, jul. 2022, doi: 10.33262/rmc.v7i3.2844.

[5] A. A. Bahashwan, M. Anbar, S. Manickam, T. A. Al-Amiedy, M. A. Aladaileh, y I. H. Hasbullah. «A Systematic Literature Review on Machine Learning and Deep Learning Approaches for Detecting DDoS Attacks in Software-Defined Networking», *Sensors*, vol. 23, n.º 9, Art. n.º 9, ene. 2023, doi: 10.3390/s23094441.

[6] M. Idhammad, K. Afdel, y M. Belouch. «Semi-supervised machine learning approach for DDoS detection», *Appl. Intell.*, vol. 48, n.º 10, pp. 3193-3208, oct. 2018, doi: 10.1007/s10489-018-1141-2

[7] M. A. Al-Shareeda, S. Manickam, y M. A. Saare. «DDoS attacks detection using machine learning and deep learning techniques: analysis and comparison», *Bull. Electr. Eng. Inform.*, vol. 12, n.º 2, Art. n.º 2, abr. 2023, doi: 10.11591/eei.v12i2.4466.

[8] P. Kumar, C. Kushwaha, D. Sethi, D. Ghosh, P. Gupta, y A. Vidyarthi. «Investigating the performance of multivariate LSTM models to predict the occurrence of Distributed Denial of Service (DDoS) attack.», *PLoS ONE*, vol. 20, n.º 1, pp. 1-17, 20250117, doi: 10.1371/journal.pone.0313930.

[9] S. Aktar y A. Yasin Nur. «Towards DDoS attack detection using deep learning approach», *Comput. Secur.*, vol. 129, n.º 20, pp. 10-25, jun. 2023, doi: 10.1016/j.cose.2023.103251.

[10] S. S. Priya, M. Sivaram, D. Yuvaraj, y A. Jayanthiladevi. «Machine Learning based DDOS Detection», en *2020 International Conference on Emerging Smart Computing and Informatics (ESCI)*, mar. 2020, pp. 234-237. doi: 10.1109/ESCI48226.2020.9167642.

[11] R. Amrish, K. Bavapriyan, V. Gopinaath, A. Jawahar, y C. V. Kumar. «DDoS Detection using Machine Learning Techniques», *J. IoT Soc. Mob. Anal. Cloud*, vol. 4, n.º 1, pp. 24-32, may 2022.

[12] R. Santos, D. Souza, W. Santo, A. Ribeiro, y E. Moreno. «Machine learning algorithms to detect DDoS attacks in SDN», *Concurr. Comput. Pract. Exp.*, vol. 32, n.º 16, pp. 1-25, 2020, doi: 10.1002/cpe.5402.

[13] J. Pei, Y. Chen, y W. Ji. «A DDoS Attack Detection Method Based on Machine Learning», *J. Phys. Conf. Ser.*, vol. 1237, n.º 3, pp. 32-40, jun. 2019, doi: 10.1088/1742-6596/1237/3/032040.

[14] S. Pande, A. Khamparia, D. Gupta, y D. N. H. Thanh. «DDOS Detection Using Machine Learning Technique», en *Recent Studies on Computational Intelligence: Doctoral Symposium on Computational Intelligence (DoSCI 2020)*, A. Khanna, A. K. Singh, y A. Swaroop, Eds., Singapore: Springer, 2021, pp. 59-68. doi: 10.1007/978-981-15-8469-5_5.

[15] J. T. Mejía, M. I. Gonzales, A. del R. Torres, y N. M. Crespo. «Seguridad contra ataques DDoS en los entornos SDN con Inteligencia Artificial», *Mag. Las Cienc. Rev. Investig. E Innov.*, vol. 7, n.º 3, pp. 105-127, jul. 2022, doi: 10.33262/rmc.v7i3.2844.

[16] D. Acosta-Tejada, J. Sanchez-Galan, y N. Torres-Batista. «ABORDANDO EL DESEQUILIBRIO DE DATOS EN CLASIFICACIÓN DE ATAQUES DE DENEGACIÓN DE SERVICIO DISTRIBUIDO (DDOS)», *Congr. Nac. Cienc. Tecnol. – APANAC*, vol. 19, n.º 19, pp. 117-126, sep. 2023, doi: 10.33412/apanac.2023.3922.

[17] L. Joyanes, *Big Data: Análisis de grandes volúmenes de datos en organizaciones*. Alfaomega Grupo Editor, 2013.

[18] F. Medina-Quispe, W. Castillo-Rojas, y C. Meneses-Villegas. «Métricas para el apoyo de la exploración visual de componentes en modelos de minería de datos», *Ingeniare Rev. Chil. Ing.*, vol. 28, n.º 4, pp. 596-611, dic. 2020, doi: 10.4067/S0718-33052020000400596.

[19] J. Hernández, J. Ramírez, y C. Ferri, *Introducción a la Minería de Datos*. Pearson, 2005.

[20] M. Bitew, A. Genovese, D. Agostinello, y V. Piuri. «Robust DDoS attack detection with adaptive

transfer learning», *Comput. Secur.*, vol. 144, n.º 10, p. 103962, sep. 2024, doi: 10.1016/j.cose.2024.103962.

[21] C. Ramos, *Mineria de datos: modelos y algoritmos aplica los conocimientos al analisis predictivos*. Independiente, 2020.

[22] D. Rios y D. Gómez-Ullate, *Big data Conceptos, tecnologías y aplicaciones*. Catarata, 2019.

[23] A. G. Sánchez, P. Arguijo, J. A. Vázquez, y R. Á. Melendez. «Técnicas de selección de características y su aplicación en el análisis de polen a través de su textura», *Res. Comput. Sci.*, vol. 150, n.º 6, pp. 203-214, 2021, doi: https://rcs.cic.ipn.mx/2021_150_6/Tecnicas%20de%20seleccion%20de%20caracteristicas%20y%20su%20aplicacion%20en%20el%20analisis%20de%20polen.pdf.

[24] C. Aguirre y E. M. Poveda. «Implementación del Método de Análisis de Componentes Principales (PCA) para la reducción de la dimensionalidad en los datos inmobiliarios de la ciudad de Riobamba», *Dominio Las Cienc.*, vol. 10, n.º 3, Art. n.º 3, sep. 2024, doi: 10.23857/dc.v10i3.4022.

[25] J. Zheng, Q. Li, G. Gu, J. Cao, D. K. Y. Yau, y J. Wu. «Realtime DDoS Defense Using COTS SDN Switches via Adaptive Correlation Analysis», *IEEE Trans. Inf. Forensics Secur.*, vol. 13, n.º 7, pp. 1838-1853, jul. 2018, doi: 10.1109/TIFS.2018.2805600.

[26] L. I. Arango-Carvajal. «Predicción de la erosión del suelo mediante random forest: caso de estudio cuenca río grande, Antioquia», *Rev. Investig. Agrar. Ambient.*, vol. 15, n.º 1, Art. n.º 1, dic. 2023, doi: 10.22490/21456453.6755.

[27] V. Guzmán-Brand y L. Gélvez-García. «Identificación de patrones a través de algoritmos de machine learning en los casos registrados de intentos suicidas en una ciudad de Colombia», *Psicoespacios*, vol. 18, n.º 32, Art. n.º 32, may 2024, doi: 10.25057/21452776.1634.

[28] J. J. Espinosa-Zúñiga. «Aplicación de algoritmos Random Forest y XGBoost en una base de solicitudes de tarjetas de crédito», *Ing. Investig. Tecnol.*, vol. 21, n.º 3, pp. 1-20, sep. 2020, doi: 10.22201/fi.25940732e.2020.21.3.022.

[29] N. Correa y M. A. Leguizamón. «Regresión Logística Técnica de Machine Learning para predicciones académicas», *XIKUA Bol. Científico Esc. Super. Tlahuelilpan*, vol. 12, n.º 12, pp. 71-80, jul. 2024, doi: 10.29057/xikua.v12iEspecial.12746.

[30] D. Lévano-Rodriguez y F. Cerdán-León. «Discriminación de masas mamográficas mediante K-Nearest Neighbor y atributos BIRADS», *Rev. Científica Sist. E Informática*, vol. 2, n.º 1, pp. 1-20, 2022, doi: https://doi.org/10.51252/rcsi.v2i1.225.

[31] M. A. Díaz-Martínez, M. de los A. Ahumada-Cervantes, y J. P. Melo-Morín. «Arboles de Decisión como Metodología para Determinar el Rendimiento Académico en Educación Superior», *Rev. Lasallista Investig.*, vol. 18, n.º 2, pp. 94-104, dic. 2021, doi: 10.22507/rli.v18n2a8.

[32] J. I. Bagnato, *Aprende Machine Learning en Español Teoría + Práctica Python*. Lean Publishing, 2020.

[33] J. Pozuelo, J. Martínez, y P. Carmona. «Análisis de la utilidad del algoritmo Gradient Boosting Machine (GBM) en la predicción del fracaso empresarial», *Rev. Esp. Financ. Contab.*, vol. 47, n.º 4, pp. 507-532, 2018, doi: https://dialnet.unirioja.es/servlet/articulo?codigo=6721421.

[34] F. J. Valderrama-Purizaca, D. A. Chávez-Barturen, S. P. Muñoz-Pérez, V. Tuesta-Monteza, y H. I. Mejía-Cabrera. «Importancia de las redes neuronales artificiales en la ingeniería civil: Una revisión sistemática de la literatura», *Iteckne*, vol. 18, n.º 1, pp. 71-83, jun. 2021, doi: 10.15332/iteckne.v18i1.2542.

[35] W. A. Castañeda, B. R. Polo, y F. Vega. «Redes neuronales artificiales: una medición de aprendizajes de pronósticos como demanda potencial», *Univ. Cienc. Tecnol.*, vol. 27, n.º 118, pp. 51-60, mar. 2023, doi: 10.47460/uct.v27i118.686.

[36] J. Corona, H. Diez, y C. Morell. «Un estudio empírico del modelo de red neuronal MLP para problemas de predicción con salidas múltiples.», *Ser. Científica Univ. Las Cienc. Informáticas*, vol. 13, n.º 6, Art. n.º 6, may 2020.

[37] D. López-Betancur, R. Bosco-Durán, C. Guerrero-Méndez, R. Zambrano-Rodríguez, y T. Saucedo-Anaya. «Comparación de arquitecturas de redes neuronales convolucionales para el diagnóstico de COVID-19», *Comput. Sist.*, vol. 25, n.º 3, pp. 601-615, sep. 2021, doi: 10.13053/cys-25-3-3453.

[38] J. Shaikh, T. A. Syed, S. A. Shah, S. Jan, Q. Ul Ain, y P. K. Singh. «Advancing DDoS attack detection with hybrid deep learning: integrating convolutional neural networks, PCA, and vision transformers», *Int. J. Smart Sens. Intell. Syst.*, vol. 17, n.º 1, pp. 1-20, dic. 2024, doi: 10.2478/ijssis-2024-0040.

[39] R. M. Suárez-Castro y I. D. Vega. «Redes neuronales aplicadas al control estadístico de procesos con cartas de control EWMA», *Tecnura*, vol. 27, n.º 75, Art. n.º 75, ene. 2023, doi: 10.14483/22487638.18623.

[40] R. M. Suárez y I. D. Ladino. «Revista Tecnura», *Tecnura*, vol. 27, n.º 75, Art. n.º 75, ene. 2023, doi: 10.14483/22487638.18623.

[41] F. S. Bustamante, J. N. Fiallos, C. I. Quinatoa, y H. R. Reinoso. «Unidades recurrentes cerradas (GRU)

vs redes neuronales artificiales en la predicción de la generación eléctrica de la Central Hidroeléctrica Illuchi», *AlfaPublicaciones*, vol. 5, n.º 3, Art. n.º 3, ago. 2023, doi: 10.33262/ap.v5i3.395.

[42] J. L. Sarmiento-Ramos. «Aplicaciones de las redes neuronales y el deep learning a la ingeniería biomédica», *Rev. UIS Ing.*, vol. 19, n.º 4, Art. n.º 4, jul. 2020, doi: 10.18273/revuin.v19n4-2020001.

[43] M. J. Suarez, J. S. Gonzalez, y J. E. Espindola. «Deep Neural Network (DNN) Applied to the Analysis of Student Dropout in a Higher Education Institution», *Investig. E Innov. En Ing.*, vol. 10, n.º 1, Art. n.º 1, jun. 2022, doi: 10.17081/invinno.10.1.5607.

[44] M. Elsayed, N.-A. Le-Khac, S. Dev, y A. Jurcut. «DDoSNet: un modelo de aprendizaje profundo para detectar ataques a la red | Publicación de la conferencia IEEE | EEE Xplorar», *Simp. Int. IEEE 2020 Sobre Un Mundo Redes Inalámbricas Móviles Multimed. WoWMoM*, vol. 21, n.º 21, pp. 391-396, 2020, doi: https://doi.org/10.1109/WoWMoM49955.2020.00072.

[45] S. Farhat, M. Abdelkader, A. Meddeb-Makhlouf, y F. Zarai. «Evaluation of DoS/DDoS Attack Detection with ML Techniques on CIC-IDS2017 Dataset», presentado en 9th International Conference on Information Systems Security and Privacy, SciTePress, ene. 2025, pp. 287-295. doi: DOI: 10.5220/0011605700003405.

[46] J. Gamboa, J. Arroyave, y E. Unamuno. «Detección de Ataque de DDoS utilizando Machine Learning – algoritmo de Random Forest», *Ser. Científica Univ. Las Cienc. Informáticas*, vol. 15, n.º 3, Art. n.º 3, feb. 2022.

[47] A. Fathima, G. S. Devi, y M. Faizaanuddin. «Improving distributed denial of service attack detection using supervised machine learning», *Meas. Sens.*, vol. 30, n.º 30, p. 100911, dic. 2023, doi: 10.1016/j.measen.2023.100911.

[48] W. Alhalabi, A. Gaurav, V. Arya, I. F. Zamzami, y R. A. Aboalela. «Machine Learning-Based Distributed Denial of Services (DDoS) Attack Detection in Intelligent Information Systems», *Int. J. Semantic Web Inf. Syst. IJSWIS*, vol. 19, n.º 1, pp. 1-17, ene. 2023, doi: 10.4018/IJSWIS.327280.

[49] F. B. Saghezchi, G. Mantas, M. A. Violas, A. M. de Oliveira Duarte, y J. Rodriguez. «Machine Learning for DDoS Attack Detection in Industry 4.0 CPPSs», *Electronics*, vol. 11, n.º 4, Art. n.º 4, ene. 2022, doi: 10.3390/electronics11040602.

[50] M. Ramzan *et al.*. «Distributed Denial of Service Attack Detection in Network Traffic Using Deep Learning Algorithm», *Sensors*, vol. 23, n.º 20, Art. n.º 20, ene. 2023, doi: 10.3390/s23208642.

[51] A. I. Hassan, E. A. El Reheem, y S. K. Guirguis. «An entropy and machine learning based approach for DDoS attacks detection in software defined networks», *Sci. Rep.*, vol. 14, n.º 1, p. 18159, ago. 2024, doi: 10.1038/s41598-024-67984-w.

[52] T. Emad, Y.-W. Chong, y S. Manickam. «Comparison of ML/DL Approaches for Detecting DDoS Attacks in SDN», *Appl. Sci.*, vol. 13, n.º 5, Art. n.º 5, ene. 2023, doi: 10.3390/app13053033.

[53] A. Alabdulatif, N. N. Thilakarathne, y M. Aashiq. «Machine Learning Enabled Novel Real-Time IoT Targeted DoS/DDoS Cyber Attack Detection System», *Comput. Mater. Contin.*, vol. 80, n.º 3, pp. 3655-3683, 2024, doi: 10.32604/cmc.2024.054610.