**DETECCIÓN DE ATAQUES DDOS EN REDES IOT UTILIZANDO MODELO DE DEEP LEARNING BASADO EN EL**

**PROTOCOLO DE RED NETFLOW**



Autor

**JOHAN ANDRÉS HERRERA REYES**

**DANIEL EDUARDO CABIATIVA FELACIO**

**UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS**

**Facultad Tecnológica**

**Ingeniería en Telecomunicaciones**

Bogotá D.C. Septiembre, 2023

**DETECCIÓN DE ATAQUES DDOS EN REDES IOT UTILIZANDO MODELO DE DEEP LEARNING BASADO EN EL**

**PROTOCOLO DE RED NETFLOW**



Autor

**JOHAN ANDRÉS HERRERA REYES 20212373001**

**johaherrerar@udistrital.edu.co**

**DANIEL EDUARDO CABIATIVA FELACIO 20202373014**

**decabiativaf@udistrital.edu.co**

Modalidad

**Monografía**

**Presentado para optar al título de: Ingeniero en Telecomunicaciones**

**Director:**

**Ing. Henry Alberto Hernández Martínez, MSc.**

**UNIVERSIDAD DISTRITAL FRANCISCO JOSÉ DE CALDAS**

**Facultad Tecnológica**

**Ingeniería en Telecomunicaciones**

Bogotá D.C. Septiembre, 2023

Dedicatoria

*A Dios, por iluminar nuestro camino y darnos fortaleza. A nuestros padres y compañeros, por su inquebrantable apoyo y enseñanzas. Y a Henry Alberto Hernández, nuestro director de proyecto cuya dirección meticulosa y experta fue esencial para la culminación de este proyecto. A todos, nuestra profunda gratitud.*

Índice

[Dedicatoria 3](#_Toc154416359)

[Índice 4](#_Toc154416360)

[Índice de Figuras 7](#_Toc154416361)

[Índice de Tablas 8](#_Toc154416362)

[Índice de Anexos 9](#_Toc154416363)

[Glosario 10](#_Toc154416364)

[Lista de Abreviaturas y Siglas 11](#_Toc154416365)

[Resumen 12](#_Toc154416366)

[1. Introducción 1](#_Toc154416367)

[2. Planteamiento del problema 2](#_Toc154416368)

[3. Justificación 3](#_Toc154416369)

[4. Objetivos 4](#_Toc154416370)

[4.1. Objetivo General 4](#_Toc154416371)

[4.2. Objetivos específicos 4](#_Toc154416372)

[5. Marco de referencia 5](#_Toc154416373)

[5.1. Antecedentes 5](#_Toc154416374)

[Detección de amenazas en Redes IoT empleando modelo híbrido de Machine Learning y Deep Learning [3]: 5](#_Toc154416375)

[Diseño De Un Modelo De Detección De Intrusos En Entornos Iot Usando Inteligencia Artificial [10]. 5](#_Toc154416376)

[Modelo De Detección De Ataques DDoS (Distributed Denial Of Services), Con Base En El Clasificador Decision Tree [11]. 6](#_Toc154416377)

[Las amenazas de seguridad a las que se enfrenta IoT y las soluciones en desarrollo [12]. 6](#_Toc154416378)

[Mecanismos de seguridad en el internet de las cosas [13]. 6](#_Toc154416379)

[Un método para la identificación y prevención temprana de incidentes de ciberseguridad en dispositivos del Internet de las Cosas [14]. 7](#_Toc154416380)

[Defensa contra intrusos en redes de dispositivos IoT usando técnicas de Blockchain y Machine Learning [15]. 7](#_Toc154416381)

[NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems [5]. 7](#_Toc154416382)

[Towards a Standard Feature Set for Network Intrusion Detection System Datasets [9]. 8](#_Toc154416383)

[Medición de desempeño de sistema de monitoreo de redes con protocolo Netflow en modelo Big Data [16] 8](#_Toc154416384)

[Evaluación de modelos de machine learning para sistemas de detección de intrusos en Redes IoT [17] 8](#_Toc154416385)

[Implementación de análisis de tráfico y de flujos de red con tecnologías netflow y sflow en equipos de red de la UNAM utilizando software libre [18] 9](#_Toc154416386)

[Sistema preventivo contra ataques de Denegación de servicio web utilizando Deep Learning [19] 9](#_Toc154416387)

[The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 dataset [20] 9](#_Toc154416388)

[Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review [6] 9](#_Toc154416389)

[5.2. Marco teórico 10](#_Toc154416390)

[Internet de las cosas (IOT) 10](#_Toc154416391)

[Ataques de denegación de servicio DOS Y DDOS 10](#_Toc154416392)

[Machine learning y Deep learning 12](#_Toc154416393)

[Modelos de Deep Learning 14](#_Toc154416394)

[Algoritmos no supervisados en Deep Learning: 15](#_Toc154416395)

[Sistemas de Detección de intrusos con Machine learning y Deep learning 15](#_Toc154416396)

[Datasets (Conjunto de datos) 16](#_Toc154416397)

[Métricas para evaluar capacidad 16](#_Toc154416398)

[Análisis de tráfico 16](#_Toc154416399)

[Cisco Net Flow 17](#_Toc154416400)

[Herramientas de para machine learning y Deep learning 17](#_Toc154416401)

[6. Marco Legal 18](#_Toc154416402)

[7. Metodología 19](#_Toc154416403)

[Herramientas Y Plataformas De Deep Learning Apropiadas Para El Desarrollo 20](#_Toc154416404)

[Tensor Flow 20](#_Toc154416405)

[Keras 20](#_Toc154416406)

[Scikit-learn 21](#_Toc154416407)

[7.1. Preprocesamiento y acondicionamiento del conjunto de datos para determinar las características de un ataque DDOS en una red IOT utilizando el conjunto de datos NF-UQ-NIDS. 21](#_Toc154416408)

[Análisis Exploratorio De Los Datos 22](#_Toc154416409)

[Conjunto de datos: NF-UQ-NIDS-v2 22](#_Toc154416410)

[Carga y Reducción Dataset 23](#_Toc154416411)

[Desbalance en el conjunto de datos 23](#_Toc154416412)

[Limpieza de datos 25](#_Toc154416413)

[Reducción del conjunto de Datos 27](#_Toc154416414)

[Transformación y codificación 28](#_Toc154416415)

[Selección De Características 29](#_Toc154416416)

[Grupos De Características 30](#_Toc154416417)

[Inspección Visual 32](#_Toc154416418)

[Correlación De Pearson 33](#_Toc154416419)

[Información Mutua 36](#_Toc154416420)

[Eliminación De Características Recursiva (RFE) 37](#_Toc154416421)

[PCA 39](#_Toc154416422)

[Normalización 43](#_Toc154416423)

[División del conjunto de datos 44](#_Toc154416424)

[7.2. Implementación De Un Algoritmo De Aprendizaje Supervisado Basado En Una Red Neuronal Para Modelar El Conjunto De Datos. 45](#_Toc154416425)

[Arquitecturas y capas 45](#_Toc154416426)

[Hiperparámetros 46](#_Toc154416427)

[Métricas 49](#_Toc154416428)

[7.3. Análisis Estadístico De Los Resultados Obtenidos Utilizando El Modelo Y Comparación De Su Desempeño Frente A Otra Técnica Reportada En El Estado Del Arte. 50](#_Toc154416429)

[8. Resultados 50](#_Toc154416430)

[9. Conclusiones y Recomendaciones 50](#_Toc154416431)

[10. Referencias 51](#_Toc154416432)

[11. Anexos 52](#_Toc154416433)

Índice de Figuras

[Figura 1. Distribución Ataques conjunto datos 22](#_Toc154076652)

[Figura 2. Carga del 20% del Dataset. 24](#_Toc154076653)

[Figura 3. Total, duplicados hallados en el conjunto de datos. 25](#_Toc154076654)

[Figura 4. Eliminación de duplicados. 25](#_Toc154076655)

[Figura 5. Eliminación Valores Faltantes. 26](#_Toc154076656)

[Figura 6. Conteo valores nulos. 26](#_Toc154076657)

[Figura 8. Distribución de registros en el nuevo conjunto de datos y generación de nuevo archivo CSV. 27](#_Toc154076658)

[Figura 7.Tipos de características conjunto de datos. 27](#_Toc154076659)

[Figura 8. Resumen estadístico 46 columnas del conjunto de datos. 29](#_Toc154076660)

[Figura 9. Coeficiente de correlación características con la variable objetivo. 34](#_Toc154076661)

[Figura 10. Funciones auxiliares Análisis de componentes Principales. 41](#_Toc154076662)

[Figura 11. Cargas de los componentes Principales. 41](#_Toc154076663)

[Figura 12. Varianza explicada y Acumulada de cada componente principal. 42](#_Toc154076664)

Índice de Tablas

[Tabla 1. Comparación entre Machine Learning vs Deep Learning 12](#_Toc154076690)

[Tabla 2. Métodos ML y DL en ciberseguridad. 13](#_Toc154076691)

[Tabla 3. Ataques del conjunto de datos NF\_UQ-NIDS 22](#_Toc154076692)

[Tabla 4. Descripción de las características conjunto de datos. 31](#_Toc154076693)

[Tabla 5. Características Seleccionadas con Correlación de Pearson. 35](#_Toc154076694)

[Tabla 6. Características Seleccionadas con Información Mutua. 36](#_Toc154076695)

[Tabla 7. Características Seleccionadas mediante RFE. 38](#_Toc154076696)

[Tabla 8. Arquitecturas Implementadas para comparativa. 45](#_Toc154076697)

Índice de Anexos

Glosario

|  |  |
| --- | --- |
| **Concepto** | **Explicación breve y detallada del concepto** |
| DDoS (Distributed Denial of Service) | Es un ataque informático en el cual se utiliza una red de computadoras comprometidas para inundar un sistema con tráfico malicioso, lo que ocasiona indisponibilidad en el servicio. |
| IoT (Internet of things) | Se refiere a la interconexión de dispositivos físicos por medio de internet, lo cual les permite obtener y compartir datos del mundo real. |
| Netflow | Protocolo de red desarrollado por Cisco que se utiliza para recopilar información sobre el tráfico de red y facilitar la monitorización y el análisis del mismo |
| Deep learning | Es una rama del aprendizaje automático (machine learning) que involucra redes neuronales profundas para resolver tareas complejas de procesamiento de datos. |
| Modelo de detección | Es un conjunto de algoritmos y técnicas utilizadas para identificar patrones de comportamiento malicioso o anormal en una red. |
| Tensor | Estructura de datos multidimensional que se utiliza en el aprendizaje automático profundo para representar información como imágenes. |
| Aprendizaje profundo | Rama de la inteligencia artificial que utiliza redes neuronales profundas para aprender y realizar cierto tipo de tareas a partir de datos, pudiendo así simular el procesamiento de un humano. |
|  |  |

Lista de Abreviaturas y Siglas

|  |  |
| --- | --- |
| **Sigla/Abreviatura** | **Significado** |
| IoT | Internet of Things |
| DoS | Denial of Service |
| DDoS | Distributed Denial of Service |
| DL | Deep learning |
| Netflow | Netflow network protocol |
| IP | Internet Protocol |
| ML | Machine Learning |
| TCP | Transmission Control Protocol |
| UDP | User Datagram Protocol |
| AI | Artificial Intelligence |
| IDS | Intrusion Detection System |
| IPS | Intrusion Prevention System |
| NIDS | Network Intrusion Detection System |
| NN | Neural Networks |
| P2P | Peer to Peer |
| PCA | Principal Component Analysis |
| EDA | Exploratory Data Analysis |
| DNN | Deep Neural Network |
| CNN | Convolutional neural network |
| LSTM | Long short-term memory |
| AUC | Area Under the ROC curve |

Resumen

Los ataques cibernéticos han experimentado un aumento alarmante en la región de América Latina y el Caribe sufriendo 137 mil millones intentos de ciberataques entre enero del 2022 y junio del mismo año, lo que representa un incremento del 50% en comparación con el mismo período del año anterior, que registró 91 mil millones de intentos [1]. Además de este aumento en la frecuencia, la sofisticación de estos ataques continúa en aumento, ya que los ciberatacantes permanecen a la vanguardia y desarrollan técnicas cada vez más elaboradas, incluyendo el uso de inteligencia artificial [2]. Esta creciente complejidad plantea la posibilidad de que el volumen y la sofisticación de los ciberataques superen la capacidad de detección humana en un futuro cercano.

En este contexto, la inteligencia artificial (IA) emerge como un multiplicador de fuerza esencial. No solo permite a los equipos de seguridad responder de manera más rápida que los atacantes cibernéticos, sino que también brinda la capacidad de anticipar sus movimientos. En respuesta a este desafío, este proyecto propone diseñar e implementar un modelo de Deep Learning para la detección y prevención de ataques DDoS en redes IoT. Este enfoque se basa en el análisis de conjuntos de datos estandarizados que capturan características cruciales mediante el protocolo Netflow.

Es importante destacar que la estandarización de características a partir de cuatro conjuntos de datos de referencia en el campo de la ciberseguridad de IoT facilita la medición del rendimiento del modelo entrenado en un conjunto de datos específico y su aplicabilidad en entornos reales [3]. Este enfoque contribuirá a fortalecer la seguridad en las redes IoT, en un momento en que la ciberseguridad se ha convertido en una prioridad crítica.

*Palabras clave: Ciberataque, Aprendizaje profundo, Ciberseguridad, Redes Neuronales, Estandarización, Internet de las Cosas.*

1. Introducción

La digitalización fomenta en la vida cotidiana y de las industrias un aumento exponencial en el tráfico de redes. El informe de Deloitte 2022 destaca este fenómeno, revelando que después de la pandemia, el espacio digital está experimentando un crecimiento sin precedentes. Este aumento ha dado lugar a un panorama donde una variedad de servicios, desde transacciones hasta soluciones en la nube para infraestructuras comerciales y el almacenamiento de información crítica, exigen un nivel eficiente y robusto de seguridad [2]. Además, el despliegue de las redes 5G ha contribuido al incremento de dispositivos conectados a internet en constante expansión, lo que conocemos como el Internet de las cosas (IoT). Según las proyecciones de Statista, se espera que para 2030 haya más de 29.000 millones de dispositivos IoT en todo el mundo, en comparación con los 19.000 millones proyectados para 2025 [3].

Este aumento en la interconexión digital ha generado la necesidad de garantizar la seguridad de las redes, ya que los ciberatacantes continúan evolucionando y desarrollando métodos cada vez más sofisticados para vulnerar sistemas. Entre estas amenazas, los ataques de Denegación de Servicio Distribuido (DDoS) se han convertido en uno de los más comunes y desafiantes de detectar. Su variabilidad y las características cambiantes del tráfico lo convierten en un adversario formidable [4].

Esta creciente amenaza es abordada desde el punto de vista del aprendizaje automático (machine learning), y más específicamente en el Deep Learning (DL). Esta técnica ha demostrado ser una herramienta eficaz y precisa en la detección y protección de redes contra ciberataques [5]. Sin embargo, para que estos modelos de DL sean eficientes y versátiles, es esencial estandarizar la captura de características en los conjuntos de datos de entrenamiento, lo que permitirá evaluarlos en diferentes contextos y facilitará su implementación en diversos escenarios.

En este contexto, el presente trabajo se propone diseñar, implementar y evaluar un modelo de Deep Learning que detecta y previene ataques DDoS en redes IoT. Este modelo utilizará un conjunto de datos basado en el protocolo de red Netflow, que registra flujos de red para detectar comportamientos anómalos y actividades maliciosas, como accesos no autorizados, intrusiones y otras amenazas a la seguridad de la información.

1. Planteamiento del problema

Las redes IoT están experimentando un crecimiento exponencial, impulsado por la proliferación de dispositivos de bajo costo que se conectan a Internet, como cámaras, grabadoras de video y electrodomésticos. Este aumento en la escala y complejidad de las redes IoT ha llevado a un incremento significativo en los ataques cibernéticos, cuya magnitud a menudo supera la capacidad de detección humana [1]. Los dispositivos IoT, en muchos casos, carecen de medidas de seguridad adecuadas, presentando una variedad de vulnerabilidades, como contraseñas débiles, falta de actualizaciones, servicios de red no seguros y componentes obsoletos, lo que da lugar a amenazas como la suplantación de identidad y la manipulación de datos [6].

Entre las tácticas de ataque más utilizadas se encuentran las botnets, que facilitan los ataques distribuidos de denegación de servicio (DDoS) para sobrecargar el tráfico de acceso y tomar el control de los dispositivos [4]. Esta problemática se agrava debido a la creciente cantidad de dispositivos IoT conectados a la red sin suficiente seguridad [7]. Los ataques DDoS se han mantenido en el centro de atención, siendo uno de los principales tipos de ciberataques entre 2013 y 2020, según una revisión de seguridad en IoT realizada en 2021 [6]. Además, son conocidos los ataques a servicios y empresas de renombre, como Amazon Web Services, GitHub, CNN y muchas otras [8].

El Deep Learning, ha demostrado ser eficaz para la detección y protección de estas redes contra ciberataques, su implementación en entornos reales presenta desafíos significativos [5]. La IA depende en gran medida de la disponibilidad de datos de red, que a menudo son difíciles de obtener debido a preocupaciones de seguridad y privacidad. Por lo tanto, se recurre al desarrollo de bancos de pruebas para la captura de datos que conformen los datasets para el entrenamiento de modelos de IA. Sin embargo, la falta de estandarización en la captura de características en estos datasets da como resultado modelos diseñados para entornos de prueba específicos, lo que limita su aplicabilidad en diferentes situaciones. Además, las técnicas de extracción, recopilación y almacenamiento de datos utilizadas en los datasets existentes suelen ser complejas y poco adecuadas para redes reales con alto tráfico. Se requiere, por tanto, una metodología más sencilla y eficaz para capturar características de tráfico de red y aplicar el aprendizaje profundo de manera efectiva [9].

1. Justificación

La seguridad en las redes IoT es un tema crítico que requiere una solución efectiva y escalable. A medida que la cantidad de dispositivos IoT aumenta, también lo hacen las vulnerabilidades y los ataques cibernéticos, lo que puede tener consecuencias graves en términos de privacidad, seguridad y economía.

La detección y prevención de ataques DDoS es particularmente difícil debido a la gran cantidad de dispositivos IoT que se conectan a la red con poca o ninguna seguridad. La aplicación de técnicas de Inteligencia Artificial para crear IDS (Sistemas de Detección de Intrusiones) ha demostrado ser una solución prometedora, pero aún existen desafíos importantes que deben superarse para implementarla de manera efectiva en entornos reales. Por lo tanto, investigar formas de mejorar la precisión y escalabilidad de los IDS basados en IA para redes IoT puede tener un impacto significativo en la seguridad y confiabilidad de estas redes, y en última instancia, en la protección de la privacidad y seguridad de los usuarios [6].

En la realización de este proyecto se aplicarán los conocimientos de redes, específicamente en cuanto a redes de paquetes e ingeniería de tráfico. Se utilizarán las herramientas de desarrollo de software y programación adquiridas, entre otras, utilizando el lenguaje de programación Python, y se aplicarán las habilidades para el modelamiento matemático adquiridas durante la carrera, además de los conocimientos sobre seguridad informática en cuanto a tipos de ataques. Con esto se busca contribuir a la investigación de la universidad en el campo de la Inteligencia Artificial aplicada a redes de datos.

1. Objetivos

## Objetivo General

Proponer un algoritmo de detección de ataques DDoS basado en Deep Learning utilizando

redes neuronales para redes IoT a partir de conjuntos de datos NF-UQ-NIDS basado en el

protocolo de red Netflow.

## Objetivos específicos

* Realizar un preprocesamiento y acondicionamiento de la información contenida en el conjunto de datos para determinar las características de un ataque DDoS en una red IOT utilizando el dataset NF-UQ-NIDS.
* Implementar un algoritmo de aprendizaje supervisado basado en una red neuronal para modelar el conjunto de datos.
* Realizar un análisis estadístico de los resultados obtenidos utilizando el modelo y comparar su desempeño frente a otra técnica reportada en el estado del arte.

# Marco de referencia

El campo de la seguridad de la información en el contexto de las redes del Internet de las Cosas (IoT) está experimentando una rápida evolución, impulsada por la creciente sofisticación y frecuencia de los ataques cibernéticos, como los de denegación de servicio distribuido (DDoS). La literatura relacionada proporciona un conjunto diverso de enfoques y metodologías que pueden enriquecer y contextualizar este estudio.

## Antecedentes

Los siguientes trabajos aquí mencionados han explorado modelos híbridos de Machine Learning y Deep Learning, destacando la importancia de la selección de algoritmos basada en métricas de rendimiento específicas, como la precisión y la sensibilidad, para una clasificación efectiva de ataques DDoS. Otros han enfocado sus esfuerzos en el diseño de modelos de detección de intrusiones utilizando herramientas de código abierto y datos de tráfico real. Además, se han evaluado métodos de clasificación supervisada, como los árboles de decisión, en la detección de patrones anormales en la red, utilizando conjuntos de datos actualizados que reflejan el tráfico de un usuario común, lo que puede ofrecer aportes relevantes para el presente trabajo.

Por otro lado, la estandarización de conjuntos de características para sistemas de detección de intrusiones basados en NetFlow podría proporcionar una base comparativa más sólida para la evaluación de modelos de aprendizaje automático. Este trabajo puede beneficiarse de los hallazgos de estudios que han implementado tecnologías NetFlow para el análisis de tráfico, así como de aquellos que han abordado la complejidad de los conjuntos de datos y la evaluación de sistemas de detección de anomalías. Por tanto, este estudio se posiciona en un esfuerzo continuo y colaborativo para combatir las amenazas de seguridad en IoT, proponiendo un enfoque novedoso que aprovecha las fortalezas de los modelos de Deep Learning en conjunto con la riqueza de datos proporcionada por NetFlow, buscando no solo expandir el conocimiento existente sino también ofrecer soluciones prácticas y aplicables en entornos reales.

#### Detección de amenazas en Redes IoT empleando modelo híbrido de Machine Learning y Deep Learning [3]:

Se realizó la construcción del modelo híbrido es decir utilizando algoritmos de Machine Learning y Deep Learning que posean métricas de desempeño favorables para la clasificación correcta de los ataques DoS, evaluando una serie de algoritmos con base en las métricas de Exactitud, Precisión y Sensibilidad se eligieron los algoritmos Random Forest y la Red Neuronal Convolucional para crear el sistema de detección de amenazas en redes IoT.

#### Diseño De Un Modelo De Detección De Intrusos En Entornos Iot Usando Inteligencia Artificial [10].

En este trabajo se planteó el diseño de un modelo capaz de detectar intrusiones en entornos de red IoT mediante el uso de aprendizaje automático o aprendizaje profundo y lenguaje de programación de fuente abierta como Python o matlab con el fin de contribuir a la solución de actuales problemas de seguridad en las redes emergentes IoT.

En este caso se diseñó un Sistemas De Detección De Intrusiones Basados En Red (Nids) estos sistemas Monitorean el tráfico en la red para detectar intrusiones y pueden trabajar de forma secuencial o paralela detectando datos anómalos, inapropiados u otros que puedan considerarse no autorizados y dañinos en una red. Para el diseño de tal sistema se utilizó un conjunto de datos del tráfico real de una cámara de seguridad auténticamente infectada por Mirai y BASHLITE el cual se compone de 11 subconjuntos de datos y se realizó el análisis de estos datos mediante libreria de python 3.7 y probando varios algoritmos de machine learning: Logistic Regression,K-Nearest-Neighbor. Random Forest, Support Vector Machine, Gradient Boosting Classifiers.

#### Modelo De Detección De Ataques DDoS (Distributed Denial Of Services), Con Base En El Clasificador Decision Tree [11].

En este proyecto se desarrolló un modelo efectivo de detección de intrusos mediante la técnica de Machine Learning de clasificación supervisada Decisión Tree (Árbol de decisiones), comparándola a su vez con otras técnicas de aprendizaje supervisado (Random Forest, Logistic Regression, K Nearest Neighbor Algorithm, Support Vector Machines SVM) con el fin de identificar patrones anormales en las conexiones a una red, tomando como referencia un dataset de datos llamado CIC-IDS-2017, que fue elegido por tener un tráfico de red completo y actualizado acorde a la tendencia de navegación de un usuario común, usando protocolos como HTTP, HTTPS, FTP, SSH y email. En este trabajo se centraron principalmente en ataques de tipo DDoS (ataque distribuido de negación de servicio).

#### Las amenazas de seguridad a las que se enfrenta IoT y las soluciones en desarrollo [12].

En este este artículo se realiza una descripción de los principales problemas de seguridad que afronta IoT, detallando cómo afectan estos problemas al usuario final también algunas de las soluciones que están siendo utilizadas para solventar estos problemas de seguridad, entre otras soluciones basadas en Blockchain, Fog computing, Machine learning, y edge computing.

#### Mecanismos de seguridad en el internet de las cosas [13].

En este artículo se realizó un análisis de algunos métodos de seguridad que se emplean en el internet de las cosas, teniendo en cuenta sus características, funcionamiento, beneficios y esquemas de vanguardia, así como las buenas prácticas de seguridad, generando estrategias de control y protección en tecnologías de hardware y software. Se presentan características y funcionalidades de los mecanismos de seguridad de IoT sirviendo para analizar e identificar los problemas de seguridad en los dispositivos conectados a internet.

#### Un método para la identificación y prevención temprana de incidentes de ciberseguridad en dispositivos del Internet de las Cosas [14].

En este trabajo se propuso un método basado en la unión de redes neuronales convolucionales y redes neuronales recurrentes para la detección y clasificación de ataques

de denegación de servicios en entornos IoT. Para este método se transforman los flujos de datos a un formato de imágenes de tres canales. Para realizar este trabajo el autor exploró y evaluar distintos conjuntos de datos con datos reales o simulados de un ataque de denegación de servicios en una red IoT con el fin de consolidar el más óptimo para implementar el modelo de aprendizaje profundo, posteriormente Implementó los modelos de red neuronal seleccionadas durante el al conjunto de datos e hizo una revisión de la literatura para encontrar trabajos donde se hayan abordado problemas similares que den solución a la detección de ataques DoS en IoT y hayan utilizado el conjunto Bot IoT, para realizar la comparación y validación de los resultados generados por el modelo entrenado

#### Defensa contra intrusos en redes de dispositivos IoT usando técnicas de Blockchain y Machine Learning [15].

Esta tesis describe cómo las redes de IoT se han integrado en la infraestructura industrial y cómo esto puede presentar riesgos de seguridad para las compañías. Para contrarrestar este riesgo, se utilizaron algoritmos de blockchain y técnicas de machine learning para identificar y bloquear amenazas. El proyecto propone integrar estas soluciones para crear un mecanismo de protección integral para las redes de dispositivos IoT que permita identificar amenazas y activar mecanismos seguros de transferencia de información. El objetivo final es el afinamiento de un algoritmo que permita implementar la solución propuesta, que se adecue a las capacidades computacionales de IoT industrial. La solución propuesta logra detectar y contener intrusos en una red de IoT y supera en algunos casos los mecanismos de detección tradicionales. El algoritmo es escalable y permite la interconectividad de grandes cantidades de nodos.

#### NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems [5].

Este documento aborda la dificultad de comparar los sistemas de detección de intrusos en la red (NIDS) basados en el aprendizaje automático (ML) en diferentes conjuntos de datos, lo que genera una brecha entre la investigación académica y las implementaciones prácticas. Para cerrar esta brecha, el documento proporciona cinco conjuntos de datos NIDS con un conjunto de características comunes basado en NetFlow, un formato ampliamente implementado en redes de producción. Estos conjuntos de datos están etiquetados para experimentos de clasificación de ataques y tráfico binario y multiclase y están disponibles para la comunidad de investigación. El documento presenta una evaluación de un clasificador de conjunto Extra Trees en estos conjuntos de datos como un caso de uso y un escenario de aplicación. NetFlow tiene los beneficios de que tiene una gran relevancia práctica, su amplia implementación y que permite un fácil escalamiento.

#### Towards a Standard Feature Set for Network Intrusion Detection System Datasets [9].

Este documento analiza la importancia de los sistemas de detección de intrusos en la red (NIDS) para proteger las redes informáticas contra los ataques cibernéticos, con esfuerzos recientes centrados en el desarrollo de NIDS basados en aprendizaje automático (ML). Sin embargo, la falta de un conjunto de características estándar en conjuntos de datos disponibles públicamente dificulta la comparación del rendimiento de diferentes clasificadores de tráfico basados en ML en diferentes conjuntos de datos, lo que limita la capacidad de evaluar la generalización de estos sistemas. Para abordar esta limitación, el documento propone y evalúa conjuntos de funciones NIDS estándar basados en el sistema y el protocolo de recopilación de metadatos de la red NetFlow. El conjunto de funciones de NIDS basado en NetFlow propuesto, junto con cuatro conjuntos de datos de referencia, se pone a disposición de la comunidad de investigación para permitir una evaluación más rigurosa y completa de los NIDS basados en ML. Los resultados muestran que el conjunto más grande con 43 funciones de NetFlow logra un rendimiento de clasificación consistentemente más alto en comparación con el conjunto de funciones original, que se adaptó a cada uno de los conjuntos de datos NIDS considerados.

#### Medición de desempeño de sistema de monitoreo de redes con protocolo Netflow en modelo Big Data [16]

Este documento analiza y evalúa la cadena de procesos que limita la capacidad en un canal de información utilizando el protocolo Netflow. Se identifican las aplicaciones de mayor consumo y se determina la causa de la saturación del canal. Se utiliza el modelo Ergódico, un modelo matemático que evalúa el comportamiento promedio, para identificar los procesos que causan la saturación del servicio. Se implementó la tecnología AVC "Application Visibility and Control" en los dispositivos muestreados y se diseñó un código en MATLAB para el proyecto. Con este software, se evaluó si el canal de comunicaciones es Ergódico o no. Se busca responder la pregunta ¿se puede optimizar el servicio?

#### Evaluación de modelos de machine learning para sistemas de detección de intrusos en Redes IoT [17]

Este trabajo aborda la importancia de la seguridad en las redes IoT y cómo los Sistemas de Detección de Intrusos son fundamentales para enfrentar los desafíos de seguridad. Se entrenan dos modelos de Machine Learning, Support Vector Machine y TabNet, para detectar ataques de tipo DoS, Backdoor y Reconnaissance. Los modelos son evaluados con la métrica de recall y se comparan entre sí. El modelo de mejor desempeño (TabNet) logra detectar cerca del 100% de los ataques, pero tiene un alto número de falsas alarmas (50%) y tiene dificultades para identificar el tipo exacto de ataque.

#### Implementación de análisis de tráfico y de flujos de red con tecnologías netflow y sflow en equipos de red de la UNAM utilizando software libre [18]

La propuesta busca implementar el análisis de tráfico de red en la UNAM utilizando tecnologías como NetFlow y sFlow. El objetivo es apoyar la resolución de incidentes y permitir el monitoreo en tiempo real y almacenamiento estadístico del tráfico en los enlaces de la UNAM. La solución deberá operar continuamente y notificar al Centro de Operación de RedUNAM mediante correo elec

Dataset CICDDoS2019. El sistema Dique tiene una interfaz gráfica que permite cambiar entre modo de detección y modo de prevención y muestra la información de los paquetes y su respectiva clasificación.

#### Sistema preventivo contra ataques de Denegación de servicio web utilizando Deep Learning [19]

El documento describe el proceso de investigación para crear un sistema preventivo contra ataques DDoS en servidores web y el diseño y construcción de un software que integra una interfaz gráfica de usuario con el modelo de clasificación. El sistema Dique clasifica los paquetes que ingresan a la red en dos tipos: Maligno y NoMaligno, utilizando un algoritmo de Deep Learning con la red neuronal artificial Deep Feed Forward y se entrenó con el Dataset CICDDoS2019. El sistema Dique tiene una interfaz gráfica que permite cambiar entre modo de detección y modo de prevención y muestra la información de los paquetes y su respectiva clasificación.

#### The evaluation of Network Anomaly Detection Systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 dataset [20]

Este documento analiza la importancia de los sistemas de detección de anomalías (ADS) en la detección de nuevos ataques en los sistemas de detección de intrusos en la red (NIDS). Sin embargo, la evaluación de los NIDS utilizando conjuntos de datos de referencia existentes no arroja resultados satisfactorios. Para abordar esto, se generó el conjunto de datos UNSW-NB15, que incluye estilos de ataque modernos de bajo impacto y nuevos patrones de tráfico normal. En este documento se muestra la complejidad del conjunto de datos UNSW-NB15 en tres aspectos: análisis estadístico, correlaciones entre características y evaluación utilizando cinco clasificadores existentes. Los hallazgos revelan que el conjunto de datos UNSW-NB15 es más complejo que el conjunto de datos KDD99, lo que lo convierte en un nuevo punto de referencia para evaluar los sistemas de detección de intrusiones en red (NIDS).

#### Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review [6]

Este articulo analiza los desafíos y riesgos asociados con la tecnología IoT, incluida la gestión de la seguridad, la heterogeneidad, la escalabilidad y la calidad del servicio. La falta de seguridad en los dispositivos IoT supone un riesgo importante y puede dar lugar a ataques de seguridad y pérdidas financieras. El texto también se centra en los ataques de denegación de servicio distribuido (DDoS) y su impacto en el mundo cibernético. Se analizan diferentes tipos de ataques DDoS, impactos y soluciones de mitigación, con una comparación de los modelos de detección y prevención de intrusiones para mitigar los ataques DDoS. El texto también cubre varias técnicas de aprendizaje automático y aprendizaje profundo para el preprocesamiento de datos y la detección de malware. Finalmente, el texto analiza los desafíos de investigación y las soluciones propuestas para el futuro de la seguridad de IoT.

## Marco teórico

En esta seccion, se exploraron los conceptos y teorías relacionadas con la detección de ataques DDoS utilizando algoritmos basados en Deep Learning y redes neuronales. Se examinará la importancia de los conjuntos de datos NF-UQ-NIDS, que son datasets estandarizados basados en el protocolo de red Netflow, y cómo pueden ser utilizados para entrenar y evaluar los modelos de detección.

#### Internet de las cosas (IOT)

Es una red de dispositivos interconectados que se comunican entre sí y con servidores en la nube. Esto ha sido posible gracias a microchips económicos y una infraestructura de comunicaciones de alta velocidad. Como resultado, ha habido un aumento en dispositivos conectados, desde cepillos de dientes hasta automóviles y maquinaria industrial, todos equipados con sensores para recopilar datos y tomar decisiones inteligentes.[x1]

IoT tuvo sus inicios en los años 90 cuando ingenieros comenzaron a incorporar sensores y unidades de procesamiento en objetos cotidianos. A pesar de que la evolución fue inicialmente lenta debido al tamaño de los microchips, la miniaturización y el desarrollo de chips de radiofrecuencia de baja potencia, como las etiquetas RFID, redujeron significativamente el tamaño y el costo de integrar capacidades computacionales en objetos pequeños.[x1]

Un ejemplo de esta tendencia es la adición de funciones de conectividad mediante asistentes de voz como Alexa a dispositivos con menos de 1MB de memoria RAM, como interruptores de luz. Todo esto ha dado lugar a una industria que llena los entornos, desde hogares hasta oficinas, con dispositivos de IoT que pueden enviar datos automáticamente a Internet. En conjunto, estos "dispositivos de cómputo discretos" y las tecnologías que los respaldan se conocen como Internet de las cosas.[x1]

#### Ataques de denegación de servicio DOS Y DDOS

Un ataque de Denegación de servicios DoS es un tipo de ataque que tiene como objetivo sobrecargar un servidor, una red o una página web para que los servicios que estos prestan sean inaccesibles para los usuarios reales o legítimos. Los ataques DoS se realizan sobrecargando el servicio o recurso que se quiere denegar mediante un gran volumen de tráfico o solicitudes de modo que las solicitudes de los usuarios reales no puedan ser procesadas. Estos Ataques se realizan para causar pérdidas económicas o de reputación para la organización afectada. [6] Los métodos más comunes son:

* **Ataque de inundación de tráfico:** el atacante envía una gran cantidad de tráfico falso al servicio o recurso objetivo con el fin de agotar sus recursos de red.
* **Ataque de amplificación:** el atacante envía solicitudes al servidor de un tercero que responderán con paquetes de datos mucho más grandes, lo que puede sobrecargar el servicio o recurso objetivo.
* **Ataque de reflexión:** el atacante falsifica la dirección IP de la solicitud para que parezca que proviene de un tercero. Luego, envía la solicitud a un servidor de ese tercero, que responde con una gran cantidad de datos al servicio o recurso objetivo, lo que puede sobrecargarlo.
* **Ataque de consumo de recursos:** el atacante aprovecha vulnerabilidades en el software o hardware del servicio o recurso objetivo para agotar sus recursos, como el uso excesivo de CPU o memoria.

Un ataque DDoS (Denegación de Servicio Distribuido) es una variante avanzada de un ataque DoS (Denegación de Servicio), en el que se utilizan múltiples dispositivos comprometidos en diferentes lugares para sobrecargar una red, sistema o servidor específico. El objetivo es hacer que el servicio sea inaccesible para usuarios legítimos. La diferencia clave entre DDoS y DoS radica en que en DDoS participan varios dispositivos controlados por una o varias personas, mientras que en DoS un solo dispositivo realiza el ataque.

Los atacantes suelen infectar numerosos dispositivos previamente con malware para crear una red de bots llamada botnet. Luego, utilizan esta botnet para enviar tráfico malicioso a la víctima, saturando su infraestructura de red y volviéndola inaccesible.

Los atacantes pueden aprovechar dispositivos IoT para llevar a cabo ataques DDoS masivos, ya que incluso unos pocos dispositivos comprometidos pueden generar suficiente tráfico para interrumpir un servicio en línea. Por lo tanto, es esencial asegurar y proteger los dispositivos IoT contra accesos no autorizados para prevenir su uso en ataques maliciosos.

La clasificación de los ataques DDoS se puede realizar según las siguientes categorías:

* **Basados en volumen:** se enfocan en inundar la red del objetivo con una gran cantidad de tráfico, lo que puede saturar la capacidad de la red y hacer que los servicios sean inaccesibles. Algunos ejemplos de ataques basados en volumen son los ataques de inundación UDP, SYN Flood, Ping Flood, entre otros.
* **Basados en la capa de aplicación:** buscan explotar vulnerabilidades en aplicaciones específicas, como servidores web, para agotar los recursos del servidor y hacer que sea inaccesible. Ejemplos de ataques de capa de aplicación incluyen el ataque HTTP Flood, el ataque Slowloris, entre otros.
* **Basados en amplificación:** estos ataques aprovechan la capacidad de algunos servicios para amplificar el tráfico de los atacantes. Por ejemplo, un atacante puede enviar una solicitud falsa a un servidor DNS que hace que el servidor responda con una gran cantidad de datos a la víctima. Los ataques de amplificación incluyen el ataque DNS Amplification, NTP Amplification, entre otros.
* **Basados en vulnerabilidades:** aprovechan las vulnerabilidades específicas en dispositivos conectados a Internet de las cosas (IoT) para crear botnets y lanzar ataques DDoS. Los atacantes pueden aprovechar contraseñas predeterminadas débiles o vulnerabilidades conocidas en dispositivos IoT para tomar el control de ellos y utilizarlos para lanzar ataques DDoS.
* **Ataques combinados:** Los ataques DDoS también pueden combinar varias técnicas para hacer que el ataque sea más difícil de mitigar. Por ejemplo, un atacante puede lanzar un ataque de inundación de la capa de red al mismo tiempo que un ataque de amplificación de DNS para maximizar el impacto del ataque.

#### Machine learning y Deep learning

El aprendizaje automático, o Machine Learning, es un subcampo de la inteligencia artificial que se enfoca en desarrollar algoritmos que permiten a las máquinas aprender a partir de datos, sin necesidad de programarlas explícitamente. Se emplea para llevar a cabo diversas tareas, como clasificación, regresión, agrupación, detección de anomalías, reconocimiento de patrones y toma de decisiones.

El aprendizaje profundo, o Deep Learning, es una subdisciplina del aprendizaje automático que se especializa en aprender y extraer características de datos complejos mediante el uso de redes neuronales artificiales profundas. Estas redes constan de múltiples capas y se aplican en la detección de patrones, el reconocimiento de imágenes y voz, el procesamiento del lenguaje natural, entre otros. Se entrenan con grandes conjuntos de datos y han tenido un impacto significativo en la resolución de problemas de inteligencia artificial, impulsando avances notables en áreas como la visión artificial, la robótica y la conducción autónoma.

|  |  |  |
| --- | --- | --- |
|  | Machine Learning (ML) | Deep Learning (DL) |
| Estructura de datos | Requiere datos estructurados y definidos con características definidas. | Puede manejar datos estructurados y no estructurados. |
| Algoritmos | Utiliza algoritmos supervisados, no supervisados y de refuerzo. | Utilizan redes neuronales profundas, tales como las redes neuronales convolucionales (CNN) y también las redes neuronales recurrentes (RNN). |
| Representación de características | Requiere que las características se extraigan manualmente. | Puede extraer automáticamente las características de los datos. |
| Capacidad de generalización | Menos propenso a la sobreajuste y puede generalizar bien para datos nuevos. | Más propenso al sobreajuste, pero puede generalizar bien para datos nuevos si se entrena adecuadamente. |
| Escalabilidad | Limitado en la escalabilidad, requiere mucho tiempo para entrenar grandes conjuntos de datos. | Es altamente escalable y puede entrenar grandes conjuntos de datos en paralelo utilizando GPUs. |

Tabla 1. Comparación entre Machine Learning vs Deep Learning

Apruzze y Colajanni[21] propusieron una taxonomía original dirigida específicamente a los operadores de seguridad. Esta clasificación distingue el aprendizaje profundo (DL) moderno de los algoritmos tradicionales de aprendizaje automático, llamado aprendizaje superficial (SL). Mientras que SL requiere que los expertos en el dominio identifiquen las características relevantes de los datos antes de ejecutar el algoritmo, DL permite que la selección de características sea autónoma mediante el aprendizaje de la representación.

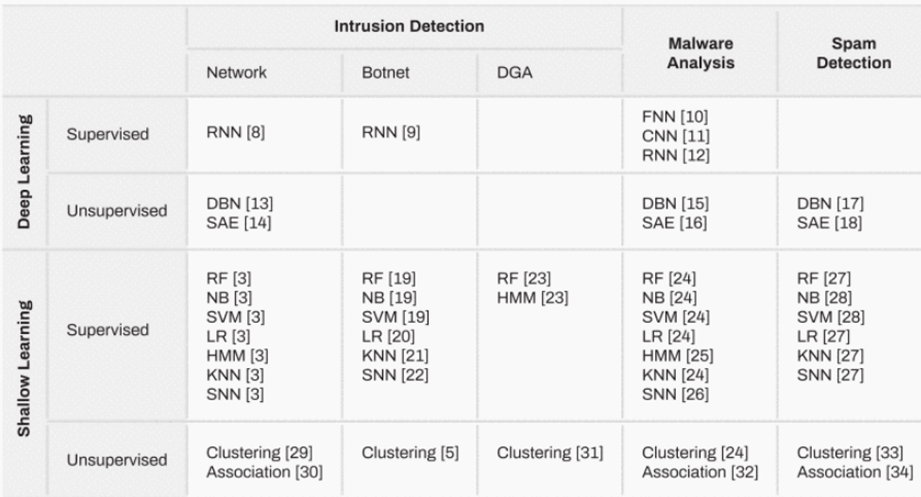


Tabla 2. Métodos ML y DL en ciberseguridad.

#### Modelos de Deep Learning

Los algoritmos supervisados en Deep Learning son una rama de la inteligencia artificial que se enfoca en el aprendizaje automático. Los algoritmos supervisados son aquellos que se entrenan con un conjunto de datos etiquetados y luego se utilizan para predecir la etiqueta de nuevos datos.

* **Las Redes Neuronales Convolucionales, o CNN en inglés**, son un tipo de red neuronal profunda ampliamente empleado en la visión por computadora, especialmente en la clasificación de imágenes. Su concepto principal consiste en procesar la imagen de entrada mediante capas de convolución y agrupación para extraer sus características, y posteriormente utilizar estas características en capas completamente conectadas para llevar a cabo la clasificación.
* **Las redes neuronales recurrentes (RNN)** son un tipo de red neuronal que se utiliza principalmente para el procesamiento de lenguaje natural. Estas redes son capaces de procesar secuencias de datos y recordar información anterior para predecir la siguiente palabra o frase.
* **Las redes neuronales profundas** son un tipo de red neuronal que se utiliza para procesar grandes conjuntos de datos. Estas redes tienen muchas capas ocultas y son capaces de detectar patrones complejos en los datos.
* **Las redes neuronales de alimentación** hacia adelante son un tipo de red neuronal que se utiliza principalmente para clasificación y regresión. Estas redes tienen una estructura simple y están compuestas por una capa de entrada, una o varias capas ocultas y una capa de salida.

#### Algoritmos no supervisados en Deep Learning:

* **Autoencoders:** son redes neuronales que aprenden a comprimir datos en una representación de menor dimensión, llamada "codificación", y luego reconstruyen los datos de la codificación. Se utilizan en aplicaciones como la reducción de la dimensionalidad, la eliminación de ruido y la generación de nuevas muestras.
* **Redes neuronales generativas (GAN):** son redes que generan muestras de datos sintéticos similares a los datos de entrenamiento. Una red GAN consta de dos partes: un generador que produce datos sintéticos y un discriminador que evalúa la autenticidad de los datos. Se utilizan en aplicaciones como la generación de imágenes y la síntesis de voz.
* **Redes neuronales convolucionales 3D (3DCNN):** son extensiones de las redes neuronales convolucionales (CNN) que se utilizan para datos tridimensionales, como videos y volúmenes de datos médicos. Utilizan filtros 3D para extraer características espaciales y temporales.
* **Redes neuronales de flujo:** son redes que modelan la distribución de probabilidad de los datos de entrada mediante el uso de transformaciones de flujo inverso. Utilizan la regla de cambio de variables para calcular la densidad de probabilidad del espacio de entrada y se utilizan en aplicaciones como la generación de imágenes y el modelado de la densidad de probabilidad de los datos.

#### Sistemas de Detección de intrusos con Machine learning y Deep learning

Los sistemas de prevención de ataques DDoS se dividen en dos categorías: sistemas de detección de intrusiones (IDS) y sistemas de prevención de intrusiones (IPS). Los IDS emiten alertas en caso de intrusión sin tomar medidas, mientras que los IPS toman medidas de penalización.

Los IDS son herramientas de seguridad utilizadas después de los firewalls y se dividen en IDS basados en host y en red. Los basados en host monitorean un host específico, mientras que los basados en red supervisan todo el tráfico de red, detectan y responden a ataques. Los IDS también se dividen en basados en firmas (comparan con una base de datos de firmas conocidas) y en anomalías (detectan comportamientos anómalos en relación con el comportamiento normal del sistema).

Los IDS tienen ventajas como la detección temprana y la recopilación de información detallada, pero presentan debilidades como la fragmentación de paquetes y la dificultad para distinguir entre comunicaciones normales y ataques. Un IDS consta de tres componentes principales: recopilación de datos, selección/conversión de características y un motor de decisiones que determina si se ajusta a la definición de una intrusión, pudiendo ser basado en firmas o anomalías. Este motor es crucial para la eficiencia del sistema de IDS. [22]

#### Datasets (Conjunto de datos)

En el campo de la ciberseguridad, se han identificado 15 conjuntos de datos (datasets) para entrenar sistemas de detección de intrusos, según una encuesta de 2019 [17]. Estos datasets presentan diversas características seleccionadas por los autores en función de sus conocimientos y experiencias, pero análisis posteriores han revelado que muchas de estas características carecen de relevancia y su diversidad dificulta la generalización del rendimiento de los modelos en un dataset específico [11]. Además, estos datasets se generan a partir de entornos virtuales, lo que crea una brecha entre los modelos desarrollados con ellos y los sistemas IDS utilizados en entornos del mundo real.

#### Métricas para evaluar capacidad

La Matriz de Confusión (CM) es una herramienta comúnmente utilizada para evaluar la corrección de un modelo de clasificación. Si bien no es una medida de rendimiento en sí misma, la mayoría de las otras métricas se derivan de sus parámetros. Para reducir los errores, la CM proporciona dos estrategias: disminuir los Falsos Negativos o los Falsos Positivos. Qué estrategia utilizar depende del contexto; por ejemplo, en la clasificación de spam de correo electrónico, se deben minimizar los Falsos Positivos, mientras que en la clasificación de pacientes con cáncer se deben minimizar los Falsos Negativos.[6]

Hay varias métricas de rendimiento derivadas de la CM, que incluyen la Exactitud, la Precisión, la Sensibilidad/Recall, el Puntaje F-1, la Especificidad y la Curva AUC-ROC.

* **La exactitud** mide el número de predicciones correctas sobre el total de predicciones, pero es más adecuada para conjuntos de datos balanceados.
* **La precisión** determina la precisión de un modelo en encontrar el número de positivos reales de los positivos totales predichos, y es útil cuando los Falsos Positivos son costosos, como en la detección de spam de correo electrónico.
* **El recall** mide la precisión de un modelo en encontrar el número de positivos de los positivos reales totales y es útil cuando los Falsos Negativos son costosos, como en la detección de fraude.
* **El Puntaje F-1** se calcula como la Media Armónica de la Precisión y el Recall, otorgando igual importancia a ambas métricas, y es útil para evaluar el rendimiento de un conjunto de datos desequilibrado.
* **La especificidad** es el opuesto del Recall y mide la tasa de Falsos Positivos.

La Curva AUC-ROC es una medida de la estabilidad entre la Precisión y el Recall al variar los umbrales y se representa por el Área Bajo la Curva (AUC), siendo una mayor AUC indicativa de un mejor modelo de clasificación.

Otras métricas, como la métrica KAPPA y el Error Cuadrático Medio (RMSE), también pueden utilizarse según los requisitos específicos.

#### Análisis de tráfico

El monitoreo y análisis de redes involucra la captura de todo el tráfico de red o un resumen en forma de flujos. Para prevenir ataques DDoS con dispositivos IoT, es esencial analizar patrones de tráfico y mitigar el tráfico malicioso que podría sobrecargar la red. Estos ataques suelen originarse en dispositivos IoT infectados. Las técnicas incluyen la inspección de paquetes y la detección de patrones inusuales, además del uso de soluciones como el filtrado de paquetes para bloquear el tráfico malicioso antes de llegar a la red de destino.

#### Cisco Net Flow

NetFlow de Cisco es un protocolo desarrollado por Cisco Systems para recopilar información sobre el tráfico de red, como el origen, el destino, el tipo de protocolo y el tiempo. Este protocolo ayuda en la gestión y solución de problemas de redes al recopilar estos datos a través de routers y switches, enviándolos a un colector de NetFlow para su análisis.[16]

El NetFlow se usa para diversas finalidades, como detectar problemas de red, monitorear la actividad, planificar la capacidad e identificar tráfico malicioso. También sirve para analizar el comportamiento del usuario y mejorar el rendimiento de la red según las necesidades de la organización.

Los sistemas de detección de intrusos en red (NIDS) basados en aprendizaje automático (ML) son prometedores en la ciberseguridad, pero enfrentan desafíos en la implementación práctica debido a la falta de conjuntos de datos comunes. La idea de crear conjuntos de datos NIDS basados en NetFlow con características relevantes es una solución potencial. Utilizar NetFlow como formato común ofrece ventajas prácticas y de escalabilidad, mejorando la seguridad cibernética en redes del mundo real. [5]

#### Herramientas de para machine learning y Deep learning

* **Jupyter Notebook:** Una aplicación web de código abierto que permite crear documentos interactivos con código, ecuaciones, visualizaciones y texto narrativo.
* **Scikit-Learn:** Una biblioteca en Python para algoritmos de clasificación, regresión y clustering, aunque no está diseñada para aprendizaje profundo.
* **TensorFlow:** Ampliamente utilizado para aprendizaje profundo en Python, ofrece una amplia gama de herramientas y es altamente personalizable.
* **Keras:** Interfaz de aprendizaje profundo en Python compatible con TensorFlow, que simplifica la construcción de redes neuronales.

# Marco Legal

NTC 27001:2006c es una norma técnica colombiana que establece los requisitos para un sistema de gestión de seguridad de la información.

Ley 1273 de 5 de enero de 2009 es una ley colombiana que modifica el Código Penal y crea un nuevo bien jurídico tutelado - denominado “de la protección de la información y de los datos” - y se preservan integralmente los sistemas que utilicen las tecnologías de la información y las comunicaciones.

CONPES 3701 de 2011 es un documento que establece los lineamientos de política para ciberseguridad y ciberdefensa en Colombia.

Resolución 500 del 2021 es una resolución del Ministerio de Tecnologías de la Información y las Comunicaciones (MinTIC) que establece los requisitos técnicos para la implementación del Sistema Nacional de Ciberseguridad.

# Metodología

Presentación del cómo se llevó a cabo el proyecto. Se puede presentar por actividades o fases del proyecto de acuerdo con el proyecto de grado aprobado. Se debe incluir un análisis del porqué de la selección de los métodos, técnicas, estándares y normas empleadas.

En este capítulo se presenta el enfoque y las técnicas adoptadas para abordar cada uno de los objetivos para la detección de ataques DDoS en redes IoT. El enfoque de este proyecto es proponer un algoritmo basado en el aprendizaje profundo y específicamente, en redes neuronales, para afrontar este problema. Este algoritmo se construirá sobre el conjunto de datos NF-UQ-NIDS, que se basa en el protocolo de red Netflow.

El objetivo principal se abordará comenzando por un preprocesamiento y acondicionamiento del conjunto de datos, con el fin de destacar y comprender las características intrínsecas de un ataque DDoS en el contexto de una red IoT. Posteriormente, nos adentraremos en el diseño e implementación de un algoritmo supervisado basado en una arquitectura de red neuronal para modelar y clasificar estos datos. Finalmente, para validar y evaluar la eficacia de nuestro modelo propuesto, llevaremos a cabo un análisis estadístico de los resultados, y contrastaremos su desempeño con otras técnicas presentes en el estado del arte. A través de estos pasos meticulosos, aspiramos a ofrecer una solución robusta y precisa para la detección de ataques DDoS en entornos IoT.

Se presenta una serie de tareas detalladas y exhaustivas para cumplir con los objetivos planteados:

* Fase 1.

Realizar un preprocesamiento y acondicionamiento de la información del conjunto de datos para determinar las características de un ataque DDoS en una red IoT.

* Carga y exploración inicial del dataset NF-UQ-NIDS.
* Identificación y tratamiento de valores faltantes o atípicos.
* Selección de características relevantes y eliminación de aquellas no esenciales.
* Normalización y escalado de las características numéricas.
* Codificación de variables categóricas si es necesario.
* División del conjunto de datos en conjuntos de entrenamiento, prueba y validacion.
* Fase 2:

Implementar un algoritmo de aprendizaje supervisado basado en una red neuronal para modelar el conjunto de datos.

* Diseño de la arquitectura de la red neuronal (número de capas, nodos, funciones de activación, etc.).
* Definición de hiperparámetros y criterios de parada.
* Entrenamiento del modelo usando el conjunto de datos de entrenamiento.
* Validación y ajuste del modelo utilizando un subconjunto de validación.
* Pruebas de detección de ataques DDoS con el conjunto de datos de prueba.
* Evaluación del rendimiento del modelo (precisión, recall, F1-score, etc.).
* Fase 3:

Realizar un análisis estadístico de los resultados obtenidos y comparar su desempeño frente a otra técnica reportada en el estado del arte.

* Revisión de técnicas reportadas en la literatura relacionadas con la detección de ataques DDoS en IoT.
* Aplicación de métricas de rendimiento sobre el modelo propuesto.
* Comparación cuantitativa y cualitativa de los resultados obtenidos con los reportados en el estado del arte.
* Documentación de los hallazgos y conclusiones obtenidas.
* Propuestas de mejoras y trabajos futuros basados en los resultados del análisis.

Con estas tareas, se busca llevar a cabo de manera organizada y metódica el desarrollo y análisis de un algoritmo de detección de ataques DDoS en redes IoT, desde la adquisición y preprocesamiento de datos hasta la evaluación y comparación con técnicas existentes.

## Herramientas Y Plataformas De Deep Learning Apropiadas Para El Desarrollo

#### Tensor Flow

TensorFlow, una biblioteca de código abierto concebida por Google Brain, se destaca por su habilidad en la construcción y entrenamiento de modelos de aprendizaje profundo, además de otras técnicas de aprendizaje automático. Una de sus principales fortalezas radica en ofrecer una plataforma de cálculo numérico eficaz basada en gráficos de flujo de datos. Además, su escalabilidad es notable, permitiendo entrenar modelos no solo en CPU, sino también en GPU y en clusters que engloban múltiples GPUs. A pesar de que su nivel de abstracción pueda ser más complejo en comparación con otras herramientas, esta característica concede al usuario una amplia flexibilidad, esencial para diseñar modelos a medida.

#### Keras

Keras actúa como una interfaz de alto nivel destinada específicamente para redes neuronales, y opera encima de TensorFlow. Esta herramienta se diseñó pensando en simplificar y acelerar la experimentación. Ofrece bloques de construcción ya predefinidos, lo que optimiza notablemente el proceso de diseño de modelos. Cabe destacar que, desde la versión 2.0 de TensorFlow, Keras se ha integrado directamente en éste, permitiendo así que los usuarios aprovechen la intuitiva interfaz de Keras a la par que se benefician del poder y capacidad de TensorFlow en segundo plano. Por todo ello, la elección de estas herramientas se alinea con el objetivo de garantizar una implementación robusta, eficiente y flexible para el proyecto.

#### Scikit-learn

En la fase inicial de nuestro proyecto, se a decidido emplear Scikit-learn, una biblioteca consagrada en el mundo del aprendizaje automático. A pesar de que está diseñada principalmente para el aprendizaje automático clásico y no para el aprendizaje profundo, Scikit-learn se erige como una herramienta altamente valiosa en las etapas preliminares de cualquier proyecto relacionado con la ciencia de datos. Esto se debe a que Scikit-learn ofrece herramientas simples pero potentes para el análisis predictivo de datos. Su diseño intuitivo permite un rápido despliegue y experimentación, lo que es crucial durante las etapas iniciales de exploración y preprocesamiento de datos. Esta biblioteca proporciona una gama extensa de algoritmos de clasificación, regresión, clustering y reducción de dimensionalidad. Tal variedad nos permitirá tener una visión completa y un entendimiento profundo de nuestros datos antes de avanzar a etapas posteriores.

Dado que no está diseñada para modelar redes neuronales profundas ni para trabajar en GPU, Scikit-learn se presenta como la herramienta ideal para la fase inicial donde el enfoque está en comprender, limpiar y preparar los datos, más que en modelar soluciones de aprendizaje profundo.

En resumen, al optar por Scikit-learn para las primeras etapas del proyecto, buscamos garantizar una exploración y un preprocesamiento de datos eficientes y robustos, creando así una base sólida para las fases subsecuentes de modelado y análisis más profundos.

## Preprocesamiento y acondicionamiento del conjunto de datos para determinar las características de un ataque DDOS en una red IOT utilizando el conjunto de datos NF-UQ-NIDS.

* Carga y exploración inicial del “Dataset” NF-UQ-NIDS.
* Identificación y tratamiento de valores faltantes o atípicos.
* Selección de características relevantes y eliminación de aquellas no esenciales.
* Normalización y escalado de las características numéricas.
* Codificación de variables categóricas si es necesario.
* División del conjunto de datos en conjuntos de entrenamiento, prueba y validación.

A continuación, se realizarán una serie de tareas de preprocesamiento antes de comenzar a realizar el entrenamiento del modelo. Primero se reducirá el conjunto de datos con el método adecuado para que nuestro modelo de aprendizaje automático no discrimine clases minoritarias y para adecuarse a la capacidad computacional. Después se investigará la integridad de los datos, buscando valores nulos y evaluando su impacto en la utilidad del conjunto. Se buscarán valores atípicos y se eliminaran o conservaran según sea el caso, considerando su impacto y si representan errores de carga o datos reales. Se identificará si los datos son discretos, continuos o categóricos para realizar las transformaciones necesarias. Se explorarán las características en busca de las que representen mayor importancia utilizando distintos métodos de selección, se mirara su distribución específica y analizara su correlación. La codificación de la variable objetivo es necesaria debido a que estamos abordando un problema de aprendizaje profundo supervisado. Finalmente se hará una División del conjunto de datos.

Este plan de análisis exploratorio y preprocesamiento nos permitirá comprender completamente nuestro conjunto de datos y tomar decisiones informadas en la próxima etapa del proyecto.

### Análisis Exploratorio De Los Datos

El análisis exploratorio de datos es un paso fundamental en cualquier proyecto de ciencia de datos y puede proporcionar una visión integral de la calidad y la estructura de tu conjunto de datos.

Comenzamos evaluando las características de nuestro conjunto de datos incluyendo la cantidad de registros disponibles y determinaremos si es adecuada para nuestros objetivos. Si los registros son insuficientes, consideraremos la capacidad de recursos (CPU y RAM) necesaria para el procesamiento.

#### Conjunto de datos: NF-UQ-NIDS-v2

En la página web de nuestro conjunto de datos encontramos la siguiente descripción del conjunto de datos:

Un conjunto de datos **NF-UQ-NIDS-v2** fusiona los siguientes conjuntos de datos: **NF-UNSW-NB15-v2, NF-ToN-IoT-v2, NF-BoT-IoT-v2, NF-CSE-CIC-IDS2018-v2.** El conjunto de datos recientemente publicado representa los beneficios de los conjuntos de características de conjuntos de datos compartidos, donde es posible fusionar múltiples conjuntos de datos más pequeños. Esto eventualmente conducirá a un conjunto de datos NIDS más grande y universal que contenga flujos de múltiples configuraciones de red y diferentes configuraciones de ataque.

Incluye una función de etiqueta adicional, que identifica el conjunto de datos original de cada flujo. Esto se puede utilizar para comparar los mismos escenarios de ataque realizados en dos o más redes de prueba diferentes. Las categorías de ataque se han modificado para combinar todas las categorías principales. Los ataques denominados ataques DoS-Hulk, ataques DoS-SlowHTTPTest, ataques DoS-GoldenEye y ataques DoS-Slowloris han cambiado de nombre a la categoría principal DoS.

Los ataques denominados ataque DDoS-LOIC-UDP, ataque DDoS-HOIC y ataques DDoS-LOIC-HTTP han cambiado de nombre a DDoS. Los ataques denominados FTP-BruteForce, SSH-Bruteforce, Brute Force -Web y Brute Force -XSS se han combinado como una categoría de fuerza bruta. Finalmente, los ataques de inyección SQL se han incluido en la categoría de ataques de inyección. El conjunto de datos NF-UQ-NIDS tiene un total de 75.987.976 registros, de los cuales 25.165.295 (33,12%) son flujos benignos y 50.822.681 (66,88%) son ataques. La siguiente tabla enumera la distribución de las categorías de ataque finales.

Tabla 3. Ataques del conjunto de datos NF\_UQ-NIDS

#### Carga y Reducción Dataset

En la descripción del conjunto de datos sé observa que hay una cantidad muy grande de datos que no es posible procesar a nivel computacional, además de que hay una variedad de datos que no son relevantes para el desarrollo del modelo, el cual solo va a estar enfocado en detección de ataques DDoS. La clase mayoritaria es de registros solamente para ataques DDoS y una cantidad similar de registros de flujos benignos. El modelo se va enfocar en detectar ataques DDoS por lo que debemos saber si se va a descartar o no los otros tipos de ataques en el conjunto de datos final con el cual se entrenará el modelo de aprendizaje profundo.

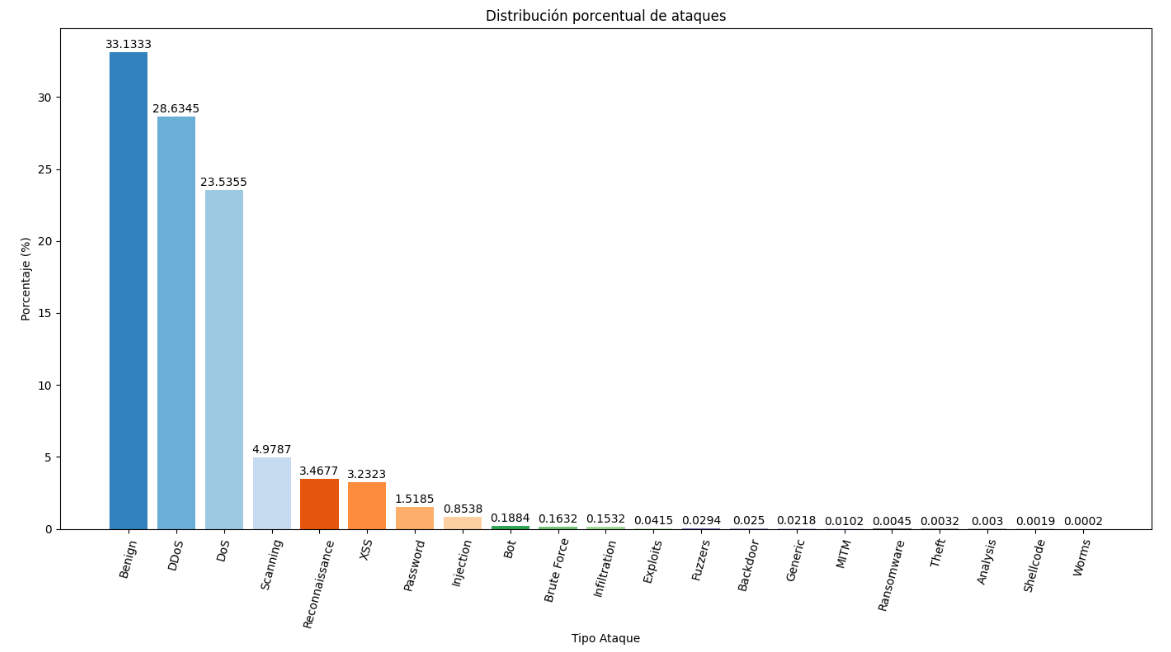


Figura 1. Distribución Ataques conjunto datos

#### Desbalance en el conjunto de datos

El Desbalance entre las clases de un conjunto de datos representa un problema que debe ser abordado desde las primeras etapas del preprocesamiento. El desbalance entre clase genera una serie de inconvenientes en las etapas de entrenamiento y evaluación del modelo como dificultad para la generalización ya que un modelo entrenado con un conjunto de datos desequilibrado no será bueno para generalizar con datos que no vistos. El desbalance de los datos puede llegar a producir una evaluación inadecuada mostrando métricas engañosas de precisión que no tienen en cuenta las clases minoritarias.

La elección de una estrategia para abordar el problema del desbalance se hizo teniendo en cuenta las características del conjunto de datos. La composición del conjunto de datos es mayoritaria para tres tipos de registros: trazas benignas, ataques DDoS y ataques DoS. Estos tres tipos de clases representan el 85,26% de todos los registros, el otro 14,74% son otros 18 tipos de ataques. Es claro que tenemos una desproporción muy alta entre los ataques DDoS y otros tipos de ataques. Sin embargo, debido a que el modelo va estar enfocado a este tipo de ataques se puede descartar todos los otros tipos de ataques y enfocarse exclusivamente en los ataques DDoS. Se deben analizar algunas ventajas y desventajas de esta elección.

*Las ventajas de descartar otros tipos de ataques:*

* **Enfoque Específico**: Al centrarse en un tipo específico de ataque, como los ataques DDoS, se puede desarrollar un modelo altamente especializado y ajustado a ese tipo de amenaza.
* **Mejora de la Eficiencia**: Si se trabajó en un entorno donde los ataques DDoS son la principal preocupación y se desea una solución de detección altamente eficiente.
* **Riesgo de Ruido Reducido**: Al eliminar registros de otros tipos de ataques, se reduce el ruido en los datos. Los modelos de aprendizaje automático pueden confundirse si tienen que distinguir entre múltiples clases de ataques, especialmente si algunas son raras o tienen patrones similares.
* **Menor Complejidad**: Al eliminar registros de otros tipos de ataques, puedes simplificar el problema y reducir la complejidad del modelo

*Las desventajas de descartar otros tipos de ataques:*

* **Limitación en la Detección**: Centrarse exclusivamente en un tipo de ataque puede llevar a un modelo que es ciego a otras amenazas. Si bien puede ser muy eficiente en la detección de DDoS, podría ser ineficaz en la detección de otros tipos de ataques.
* **Pérdida de Información**: Descartar datos de otros tipos de ataques significa perder información valiosa que podría ser relevante en un contexto más amplio de seguridad cibernética.
* **No se refleja en la Realidad**: En la realidad, las redes y sistemas enfrentan una variedad de amenazas. Descartar registros de otros tipos de ataques puede no reflejar con precisión el panorama de seguridad real.

Teniendo en cuenta los objetivos, las limitaciones del proyecto y lo mencionado anteriormente se creó un nuevo conjunto de datos ataques balanceado solo con tres clases de registros: Benignos, DDoS y DoS descartando todos los otros tipos de ataques. La carga del conjunto de datos no es posible debido a que sobrepasa la capacidad de la RAM y aparece un mensaje de error por esto realizamos una primera reducción cargando solo el 20% del Dataset o la quinta parte de este.

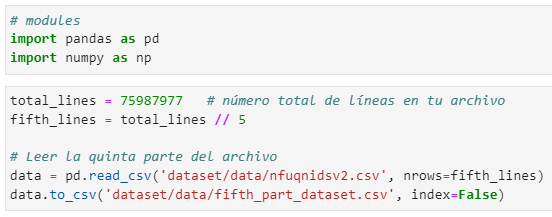




Figura 2. Carga del 20% del Dataset.

#### Limpieza de datos

Un conjunto de datos limpios es un conjunto de información que ha sido procesado, organizado y depurado de manera que esté libre de errores, inconsistencias y duplicados. Estos datos están en un formato que es fácil de entender y utilizar para el análisis.

La limpieza de datos es un paso crítico en el proceso de gestión de datos y análisis de datos, y es necesario por varias razones: Los datos limpios son precisos y confiables. Esto significa que reflejan con exactitud la realidad que representan, lo que permite tomar decisiones informadas basadas en ellos. La limpieza de datos garantiza que los datos sigan un formato y una estructura coherentes. Esto facilita su comparación y análisis, ya que los datos son uniformes y no varían en su presentación. La limpieza de datos elimina errores como valores atípicos, valores faltantes o información incorrecta. Esto ayuda a evitar que se tomen decisiones incorrectas debido a datos corruptos o inexactos. Los conjuntos de datos limpios se procesan de manera más eficiente, lo que acelera el análisis y reduce el tiempo necesario para extraer información valiosa. Los datos limpios también son más fáciles de representar visualmente en gráficos y tablas, lo que facilita la comunicación de resultados y tendencias a diferentes audiencias. La calidad de los datos ayuda a tomar decisiones más acertadas y fundamentadas.

La gestión adecuada de valores faltantes y registros duplicados es esencial para mantener la integridad y la precisión de cualquier análisis de datos. Además, la presencia de datos duplicados puede inflar artificialmente la importancia de ciertas observaciones, distorsionando estadísticas descriptivas y el desempeño de modelos predictivos. Eliminar o tratar estos valores optimiza el almacenamiento y el rendimiento computacional, y también garantiza que los insights extraídos del conjunto de datos sean representativos y fiables.

Una búsqueda con la función duplicates() de pandas arroja 4790 duplicados. Debido a que son irrelevantes para el desarrollo del modelo puesto que tienen los mismos valores en todas las columnas son eliminados.

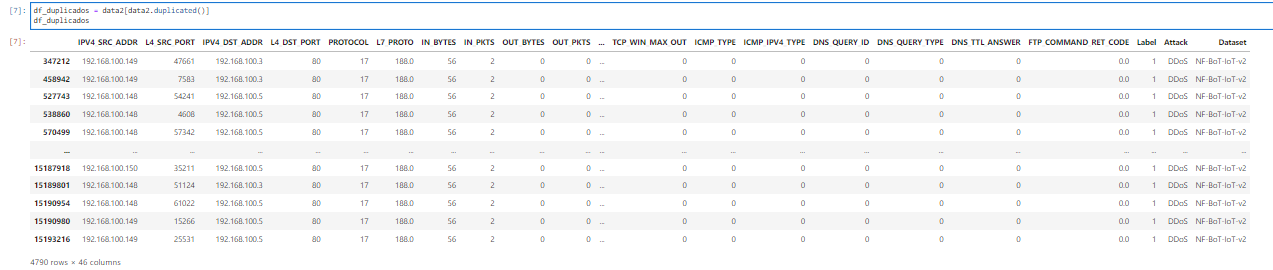


Figura 3. Total, duplicados hallados en el conjunto de datos.

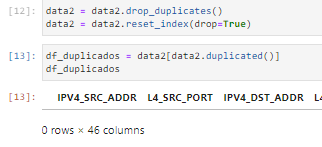


Figura 4. Eliminación de duplicados.

Los valores faltantes pueden introducir sesgos y alterar los resultados, llevando a conclusiones erróneas y a decisiones mal informadas. Debido a esto debemos tratar los datos faltantes o nulos.

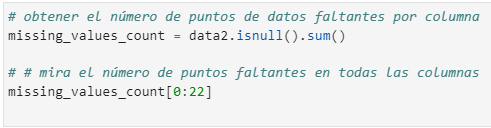


Figura 5. Eliminación Valores Faltantes.

|  |  |
| --- | --- |
|  |  |

Figura 6. Conteo valores nulos.

#### Reducción del conjunto de Datos

A partir del primer muestreo y ya habiendo hecho una limpieza y eliminación de valores nulos se crea un conjunto de datos solo con tres clases. Esta reducción del tamaño del conjunto de datos es necesaria debido a limitaciones de capacidad de RAM al momento de entrenamiento, lo que se puede dificultar con conjuntos de datos grandes. Además, la reducción del tamaño del conjunto de datos puede acelerar significativamente el tiempo de entrenamiento y evaluación del modelo, simplificando el análisis y aumentado la eficiencia del modelo. Sin embargo, es crucial garantizar que la reducción del tamaño no comprometa la representatividad del conjunto de datos y que se conserve la calidad de los datos relevantes para los objetivos del proyecto.

Para la reducción de conjunto de datos se realizó un muestreo tomando solo tres millones de datos del Dataset anterior de quince millones asegurando que todas las clases seleccionadas estén representadas proporcionalmente en la muestra. En etapas posteriores se pueden aprovechar los datos descartados para realizar validación cruzada con el fin de evaluar el rendimiento del modelo entrenado.

Para iniciar una exploración más detallada utilizando Python y comenzar a realizar pruebas se creó un nuevo conjunto de datos con las tres clases seleccionadas tomando un millón de registros por cada clase.

|  |  |
| --- | --- |
|  |  |

Figura 8. Distribución de registros en el nuevo conjunto de datos y generación de nuevo archivo CSV.

#### Transformación y codificación

En la sección anterior se hicieron dos reducciones del conjunto de datos y una limpieza de valores duplicados y nulos. Ahora con un conjunto de datos balanceado se realizarán algunas otras tareas de limpieza, transformación y codificación de las columnas o las características del conjunto con el fin de pasar a la selección de características. Para realizar limpieza se deben inspeccionar los datos para lo que se utilizan funciones de las bibliotecas pandas. Se carga el archivo csv creado para realizar una exploración de las características.

|  |  |
| --- | --- |
|  |  |

Figura 7.Tipos de características conjunto de datos.

En el conjunto de datos se encuentran distintos tipos de datos, existen enteros, flotantes y objetos se deben tratar estas columnas para que se puedan hacer los procesos posteriores de normalización y entrenamiento. En el aprendizaje automático, los datos pueden ser de dos tipos principales: numéricos y categóricos. Los datos categóricos no se pueden utilizar directamente en la mayoría de los modelos de aprendizaje automático, ya que la mayoría de los algoritmos requieren valores numéricos para realizar cálculos matemáticos. Antes de alimentar datos categóricos a un modelo de aprendizaje automático, es necesario realizar un proceso llamado "preprocesamiento" para convertir los datos categóricos en una forma que el modelo pueda entender.

El preprocesamiento de los datos categóricos se aborda de varias maneras: Eliminación, codificación y transformación. Teniendo en cuenta esto se realizaron las siguientes tareas durante el proceso de limpieza de los datos:

* Tratamiento de columnas tipo float y object mediante transformación de las características tipo float64 a valores int32 e Int64.
* Codificación Ordinal de Columnas Attack y Dataset
* Eliminación de columnas IPV4\_SRC\_ADDR, IPV4\_DST\_ADDR puesto que este es un Dataset compuesto de varios otros. Los datos de los conjuntos de datos originales fueron creados mediante bancos de prueba y no son representativos en entornos reales.
* L7\_PROTO: transformación mediante redondeo al entero menor más cercano
* Conversión de tipo a las columnas SRC\_TO\_DST\_SECOND\_BYTES, DST\_TO\_SRC\_SECOND\_BYTES, FTP\_COMMAND\_RET\_CODE a Int64

### Selección De Características

La selección de características es el proceso de elegir características de un conjunto de datos con el objetivo de que estos se ajusten de manera más adecuada al problema en cuestión. Se realiza la selección de características con el fin de mejorar el rendimiento predictivo de un modelo al reducir la influencia del ruido originado por características que no son necesarias para la predicción. Esto, a su vez, contribuye a disminuir las demandas computacionales y a mejorar la interpretabilidad y comprensión de los resultados obtenidos.

La selección de características se realiza con una variedad de técnicas entre otras: correlación de Pearson, RFE, etc. Estas técnicas nos permiten medir las asociaciones entre una característica y el objetivo con el fin de elegir un conjunto más pequeño de las características más útiles para desarrollar inicialmente. Se utilizaron algunas de estas para realizar una comparativa en el rendimiento del modelo.

**Correlación de Pearson:** La correlación de Pearson es una medida estadística que evalúa la relación lineal entre dos variables continuas. Se utiliza para cuantificar cuán fuerte y en qué dirección se relacionan dos variables. El coeficiente de correlación de Pearson, denotado como "r", varía en el rango de -1 a 1.

**Información Mutua:** Es una métrica univariable que se utiliza en una variedad de aplicaciones, como selección de características, análisis de datos, clasificación y clustering. Tiene la ventaja de poder detectar cualquier tipo de relación, mientras que la correlación sólo detecta relaciones lineales. La información mutua (IM) entre dos cantidades es una medida de hasta qué punto el conocimiento de una cantidad reduce la incertidumbre sobre la otra. La incertidumbre se mide utilizando una cantidad de la teoría de la información conocida como "entropía".

**Eliminación de Características Recursiva (RFE):** es una técnica de selección de características en aprendizaje automático que comienza entrenando un modelo con todas las características, calcula la importancia de cada una y elimina la menos importante en cada iteración. Este proceso se repite hasta alcanzar un criterio de parada, lo que resulta en un conjunto final de características óptimas que permiten un modelo más eficiente y preciso.

**PCA:** El análisis de componentes principales es una técnica de selección de características que reduce la dimensionalidad de un conjunto de datos al identificar y eliminar la información redundante o menos relevante. PCA transforma las características originales en un nuevo conjunto de características no correlacionadas, llamadas componentes principales, que capturan la mayor varianza en los datos. PCA solo funciona con funciones numéricas, como cantidades o recuentos.

#### Grupos De Características

El conjunto de datos cuenta con 46 características inicialmente por lo que debe reducirse su dimensionalidad y prepararlo para entrenar el modelo de detección de ataques DDoS, para esto se pueden utilizar varios métodos de selección de características que se denominan de filtro y envoltura con el fin de crear varios grupos de características y evaluar el rendimiento del modelo con cada uno de estos. Se realizo una visualización de las columnas mediante el comando “describe”.

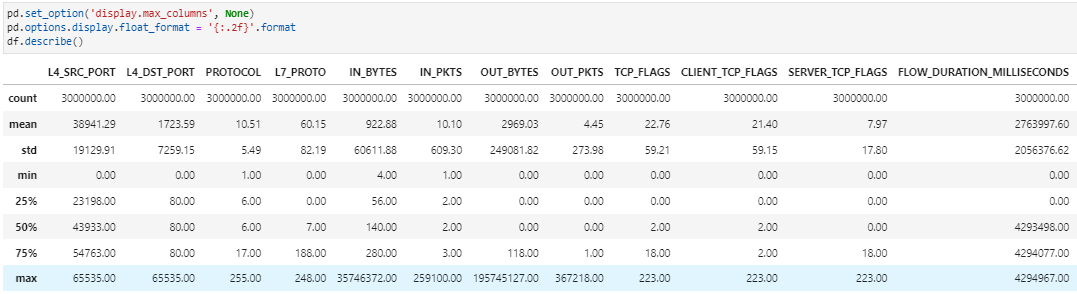


Figura 8. Resumen estadístico 46 columnas del conjunto de datos.

A continuación, realizamos una descripción breve de cada una de las columnas o características del conjunto de datos.

|  |  |
| --- | --- |
| CARACTERÍSTICA | DESCRIPCIÓN |
| L4\_SRC\_PORT | Fuente Ipv4 Puerto Número |
| L4\_DST\_PORT | Destino Ipv4 Puerto Número |
| PROTOCOL | Protocolo IP Byte Identificador |
| L7\_PROTO | Protocolo De Capa 7 (Numérico) |
| IN\_BYTES | Número De Bytes Entrante |
| OUT\_BYTES | Número De Bytes Salida |
| IN\_PKTS | Número De Paquetes Entrante |
| OUT\_PKTS | Número De Paquetes Salida |
| FLOW\_DURATION\_MILLISECONDS | Duración Del Flujo En Milisegundos |
| TCP\_FLAGS | Acumulativo De Todas Las Banderas TCP |
| CLIENT\_TCP\_FLAGS | Acumulativo De Todo Banderas TCP Del Cliente |
| SERVER\_TCP\_FLAGS | Acumulativo De Todas Las Banderas TCP Del Servidor |
| DURATION\_IN | Flujo De Cliente A Servidor Duración ( ms ) |
| DURATION\_OUT | Flujo De Servidor A Cliente Duración ( ms ) |
| MIN\_TTL | Flujo Mínimo de TTL |
| MAX\_TTL | Flujo Máximo de TTL |
| LONGEST\_FLOW\_PKT | Flujo más Largo Paquete (Bytes) |
| SHORTEST\_FLOW\_PKT | Flujo más Corto Paquete (Bytes) |
| MIN\_IP\_PKT\_LEN | Mínima longitud de paquete IP observado |
| MAX\_IP\_PKT\_LEN | Máxima longitud de Paquete IP observado |
| SRC\_TO\_DST\_SECOND\_BYTES | cantidad de bytes transmitidos desde la fuente (dst) (source-src) hacia el destino por segundo (seg) |
| DST\_TO\_SRC\_SECOND\_BYTES | cantidad de bytes transmitidos desde el destino (dst) hacia la fuente (source-src) por segundo (seg). |
| RETRANSMITTED\_IN\_BYTES | Número De Bytes De Flujo TCP Retransmitidos ( Src -> Dst ) |
| RETRANSMITTED\_IN\_PKTS | Número De Flujo TCP Retransmitido en Paquetes ( Src -> Dst ) |
| RETRANSMITTED\_OUT\_BYTES | Número De Bytes De Flujo TCP Retransmitidos ( Dst -> Src ) |
| RETRANSMITTED\_OUT\_PKTS | Número De Flujo TCP Retransmitido en Paquetes ( Dst -> Src ) |
| SRC\_TO\_DST\_AVG\_THROUGHPUT | Rendimiento promedio de la transferencia de datos en bits por segundo desde la fuente al destino. Promedio Thpt (Bps) |
| DST\_TO\_SRC\_AVG\_THROUGHPUT | Rendimiento promedio de la transferencia de datos en bits por segundo desde el destino hacia la fuente. Promedio Thpt (Bps) |
| NUM\_PKTS\_UP\_TO\_128\_BYTES | Paquetes IP de Tamaño <= 128 |
| NUM\_PKTS\_128\_TO\_256\_BYTES | Paquetes IP de Tamaño > 128 Y <= 256 |
| NUM\_PKTS\_256\_TO\_512\_BYTES | Paquetes IP de Tamaño > 256 Y <= 512 |
| NUM\_PKTS\_512\_TO\_1024\_BYTES | Paquetes IP de Tamaño > 512 Y <= 1024 |
| NUM\_PKTS\_1024\_TO\_1514\_BYTES | Paquetes IP de Tamaño >=1024 Y <= 1514 |
| TCP\_WIN\_MAX\_IN | Ventana TCP entrada Máxima ( Src -> Dst ) |
| TCP\_WIN\_MAX\_OUT | Ventana TCP entrada Máxima ( Dst -> Src ) |
| ICMP\_TYPE | Tipo ICMP \* 256 + Código ICMP |
| ICMP\_IPV4\_TYPE | Tipo ICMP |
| DNS\_QUERY\_ID | Consulta DNS ID De Transacción |
| DNS\_QUERY\_TYPE | Consulta DNS Tipo (Por Ejemplo, 1=A, 2= NS.. ) |
| DNS\_TTL\_ANSWER | TTL De El Primer Registro A ( Si Cualquier ) |
| FTP\_COMMAND\_RET\_CODE | código de retorno que un cliente FTP (File Transfer Protocol) recibe como respuesta después de enviar un comando al servidor FTP |

Tabla 4. Descripción de las características conjunto de datos.

#### Inspección Visual

La clasificación de los ataques DDoS es amplia, en la descripción del Conjunto de datos nos mencionan solo tres tipos de ataques que fueron todos clasificados bajo la categoría DDoS: DDoS attack-LOIC-UDP, DDoS attack-HOIC and DDoS attacks-LOIC-HTTP. En el estado del arte también se exploro la variedad de modalidades y características de este tipo de ataques los cuales puedes utilizar distintos protocolos: UDP, TCP, HTTP, MC-SQLR, LDAP, ICMP y otros. También aunque se considera que los ataques DDoS se caracterizan sobre todo por su volumen y velocidad (hay ataques DDoS Low-Rate que envían paquetes al objetivo ritmos muy bajos) [] .

La variedad de características, protocolos, tasas de envío y otras técnicas de los ataques DDoS es lo que los hace difíciles de detectar sin la ayuda de inteligencia artificial. Sin embargo, se puede analizar entre todas las características de conjunto de datos cuales son imprescindibles para la detección de estos ataques.

**L4\_SRC\_PORT** (Puerto de Origen de la Capa 4): Permite identificar patrones inusuales en los puertos de origen, lo que puede ser un indicio de actividad maliciosa.

**L4\_DST\_PORT** (Puerto de Destino de la Capa 4): Al igual que el puerto de origen, el puerto de destino puede revelar patrones atípicos asociados con ataques DDoS.

**PROTOCOL (Protocolo IP):** Diferentes tipos de ataques DDoS pueden usar diferentes protocolos (como TCP, UDP, ICMP). Es esencial para identificar la naturaleza del ataque.

**L7\_PROTO (Protocolo de Capa 7):**  Los ataques DDoS pueden apuntar específicamente a aplicaciones o servicios que operan en la capa 7 (la capa de aplicación), y conocer el protocolo de esta capa puede ayudar a identificar patrones específicos.

**FLOW\_DURATION\_MILLISECONDS (Duración del Flujo en Milisegundos):** Los ataques DDoS a menudo implican flujos de larga duración o inusualmente cortos, lo que hace que esta característica sea crucial para la detección.

**IN\_BYTES (Bytes Entrantes):** Un volumen anormalmente alto de bytes entrantes puede ser un indicativo de un ataque de inundación.

**OUT\_BYTES (Bytes Salientes):** Similar a los bytes entrantes, un aumento en los bytes salientes puede ser un signo de actividad de ataque.

**TCP\_FLAGS (Banderas TCP):** Las banderas TCP inusuales o la falta de ellas pueden indicar un intento de explotar vulnerabilidades del protocolo TCP.

**RETRANSMITTED\_IN\_BYTES (Bytes Retransmitidos Entrantes):** Las retransmisiones elevadas pueden ser un signo de congestión de red causada por un ataque DDoS.

**MAX\_IP\_PKT\_LEN (Longitud Máxima del Paquete IP):** Los ataques DDoS pueden manipular la longitud de los paquetes para generar tráfico anómalo.

**NUM\_PKTS\_UP\_TO\_128\_BYTES (Número de Paquetes hasta 128 Bytes):** Un aumento en el número de paquetes pequeños puede ser un indicativo de un ataque DDoS, como un ataque de inundación SYN.

Estas características ofrecen una buena combinación de información sobre el tráfico de red, los patrones de los paquetes, y las anomalías en la duración y el tamaño de los flujos lo cual es fundamental para que el modelo entrenado pueda ser eficiente en la Detección de ataques DDoS. Por otra parte, se pueden tener en cuenta otras características: MIN\_TTL y MAX\_TTL, si los paquetes toman rutas inusuales, DNS\_QUERY\_TYPE y DNS\_TTL\_ANSWER útil si el ataque se dirige a infraestructura DNS, TCP\_WIN\_MAX\_IN cuando los ataques aprovechan vulnerabilidades en control de congestión TCP.

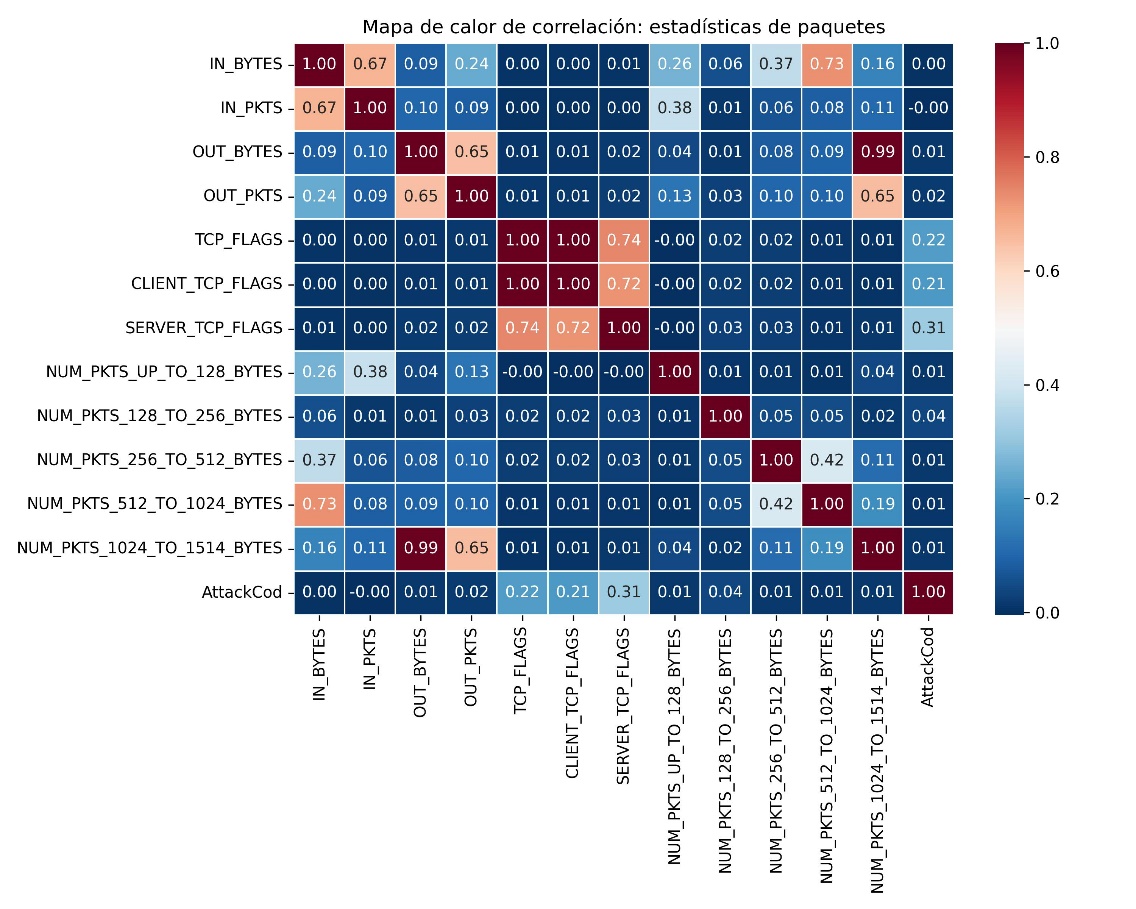
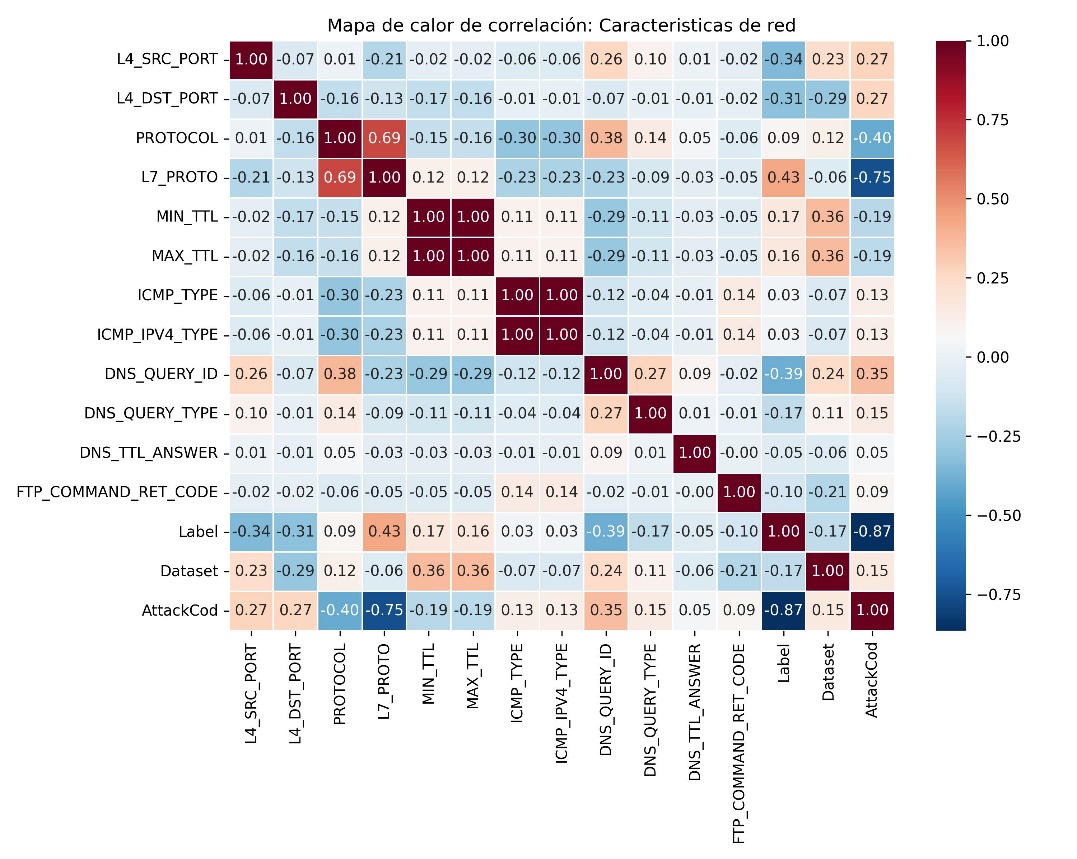
El conocimiento de dominio y la experiencia de campo es una de las formas de elegir un grupo de características relevantes para entrenar un modelo de inteligencia artificial sin embargo existen técnicas ya probadas basadas en métodos estadísticos. Se utilizarán cuatro de estas técnicas para seleccionar cuatro grupos diferentes.

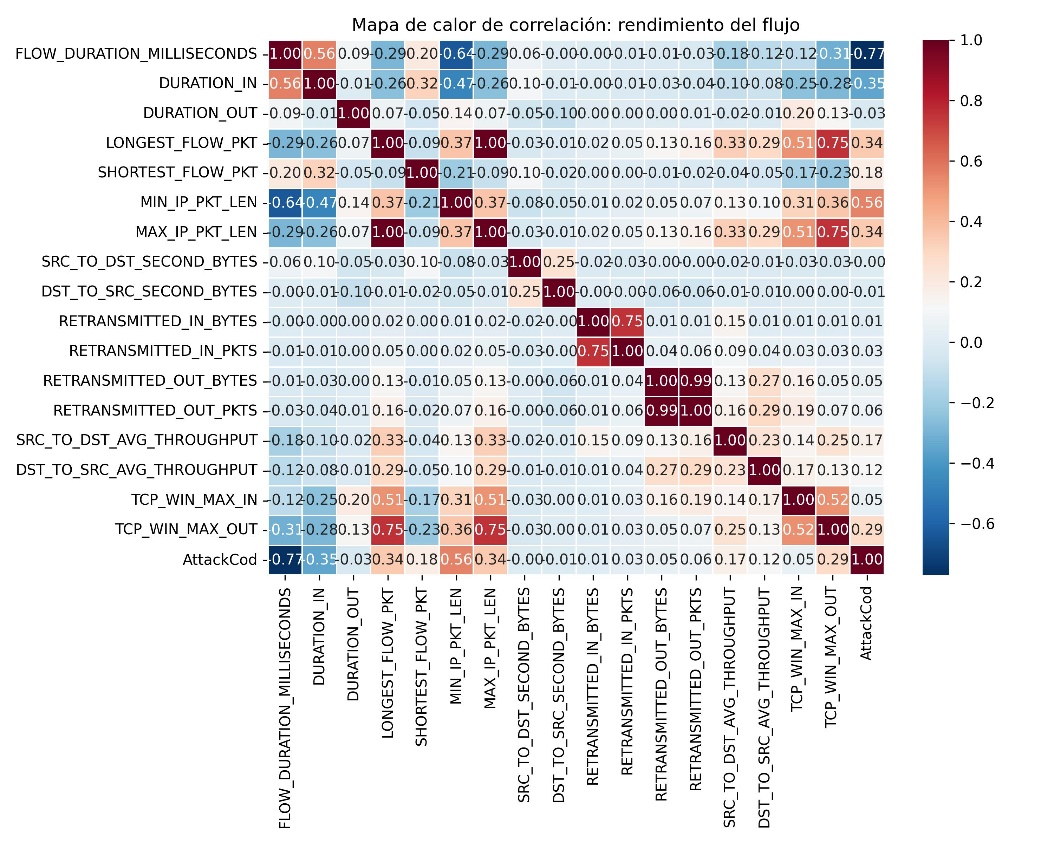
#### Correlación De Pearson

Un gráfico de correlación, típicamente visualizado como un mapa de calor, puede ser una herramienta útil al decidir qué características (features) incluir en un modelo de deep learning o cualquier otro modelo de machine learning. Una correlación cercana a 1 o -1 sugiere una fuerte relación lineal entre dos características. Una correlación cercana a 0 indica una débil relación lineal. La fórmula para realizar este cálculo es la siguiente:

Donde r es el coeficiente de correlación ‘x’ será una variable, ‘y’ la variable objetivo.

Si dos características están altamente correlacionadas (es decir, cerca de 1 o -1), es posible que contengan información redundante. En tales casos, puede considerarse eliminar una de las dos características para reducir la dimensionalidad y evitar la multicolinealidad, que puede hacer que el modelo sea inestable o más difícil de interpretar. Cada característica se correlaciona con la variable objetivo (la que se desea predecir). Las características que tienen una correlación muy baja con la variable objetivo podrían no ser muy útiles para la predicción y podrían ser candidatas a eliminarse. Sin embargo, se debe tener en cuenta que la correlación lineal no captura relaciones no lineales.





Al imprimir la tabla de correlaciones con la variable objetivo AttackCod podemos observar las características con mayor correlación positiva y negativa.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Figura 9. Coeficiente de correlación características con la variable objetivo.

Los resultados anteriores permiten seleccionar las características que tienen la mayor correlación con la variable objetivo. Se seleccionaron las correlaciones positivas y negativas mayores a un valor absoluto de 0.1

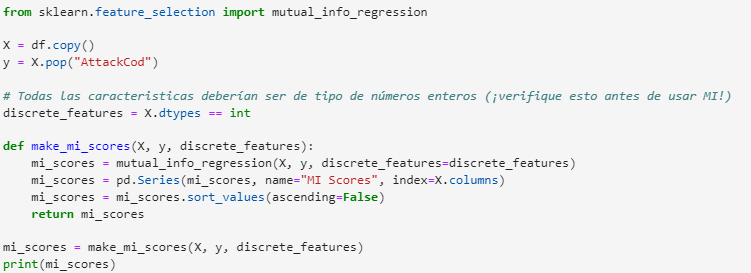
|  |  |  |  |
| --- | --- | --- | --- |
| *CARACTERISTICA* | *CORRELACION* | *CARACTERISTICA* | *CORRELACION* |
| MIN\_IP\_PKT\_LEN | 0,556295 | **SRC\_TO\_DST\_AVG\_THROUGHPUT** | 0,168938 |
| DNS\_QUERY\_ID | 0,352794 | **DNS\_QUERY\_TYPE** | 0,145678 |
| MAX\_IP\_PKT\_LEN | 0,344973 | **ICMP\_TYPE** | 0,126962 |
| LONGEST\_FLOW\_PKT | 0,344973 | **ICMP\_IPV4\_TYPE** | 0,126961 |
| SERVER\_TCP\_FLAGS | 0,314716 | **DST\_TO\_SRC\_AVG\_THROUGHPUT** | 0,120763 |
| TCP\_WIN\_MAX\_OUT | 0,285248 | **MAX\_TTL** | -0,187913 |
| L4\_SRC\_PORT | 0,272314 | **MIN\_TTL** | -0,189937 |
| L4\_DST\_PORT | 0,270411 | **DURATION\_IN** | -0,354707 |
| TCP\_FLAGS | 0,218952 | **PROTOCOL** | -0,400641 |
| CLIENT\_TCP\_FLAGS | 0,207578 | **L7\_PROTO** | -0,754175 |
| SHORTEST\_FLOW\_PKT | 0,181678 | **FLOW\_DURATION\_MILLISECONDS** | -0,770636 |

Tabla 5. Características Seleccionadas con Correlación de Pearson.

#### Información Mutua

El cálculo de la información mutua se realiza para todas las características respecto a la variable objetivo que en este caso son los ataques. La información mutua es una cantidad o métrica de la teoría de la información que tiene ciertas ventajas respecto a la correlación debido a que permite encontrar relaciones lineales y no lineales entre dos variables. Para la implementación de este algoritmo se utilizó el método ‘mutual\_info\_regression’ de ‘Scikit-learn’ el cual permite calcular la información mutua para variables discretas y continuas en nuestro caso todas las características fueron transformadas a entero por lo se utiliza solo para este caso.

El cálculo de la información mutua se realiza creando una función, esta función tiene como parámetros las variables que fueron asignadas A ‘X’ y la variable objetivo ‘y’.



|  |  |
| --- | --- |
|  |  |

En los resultados de Información Mutua (MI Scores), podemos observar que características como MAX\_IP\_PKT\_LEN, LONGEST\_FLOW\_PKT y SRC\_TO\_DST\_SECOND\_BYTES tienen los valores más altos de información mutua. Esto sugiere que estas variables tienen una mayor relación con la variable objetivo y, por tanto, podrían ser más relevantes para el modelo. La alta puntuación indica que los cambios en estas características tienen una gran influencia en la capacidad del modelo para predecir correctamente el resultado deseado. Por otro lado, algunas características como FTP\_COMMAND\_RET\_CODE y DURATION\_OUT tienen valores más bajos de información mutua, lo que indica que proporcionan menos información predictiva sobre la variable objetivo.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *CARACTERISTICA* | *MI* | *CARACTERISTICA* | *MI* | |
| LONGEST\_FLOW\_PKT | 1,02232 | **TCP\_FLAGS** | 0,50802 |
| MAX\_IP\_PKT\_LEN | 1,02138 | **L4\_DST\_PORT** | 0,49914 |
| SRC\_TO\_DST\_SECOND\_BYTES | 1,00747 | **CLIENT\_TCP\_FLAGS** | 0,49877 |
| IN\_BYTES | 0,97925 | **DST\_TO\_SRC\_SECOND\_BYTES** | 0,44583 |
| SRC\_TO\_DST\_AVG\_THROUGHPUT | 0,92029 | **OUT\_BYTES** | 0,44148 |
| SHORTEST\_FLOW\_PKT | 0,82119 | **DST\_TO\_SRC\_AVG\_THROUGHPUT** | 0,41220 |
| L7\_PROTO | 0,78841 | **NUM\_PKTS\_128\_TO\_256\_BYTES** | 0,39616 |
| TCP\_WIN\_MAX\_IN | 0,74769 | **IN\_PKTS** | 0,38376 |
| NUM\_PKTS\_UP\_TO\_128\_BYTES | 0,61783 | **OUT\_PKTS** | 0,29501 |
| FLOW\_DURATION\_MILLISECONDS | 0,61081 | **MIN\_IP\_PKT\_LEN** | 0,26746 |
| DURATION\_IN | 0,60465 | **PROTOCOL** | 0,25933 |
| MIN\_TTL | 0,51701 | **TCP\_WIN\_MAX\_OUT** | 0,20855 |

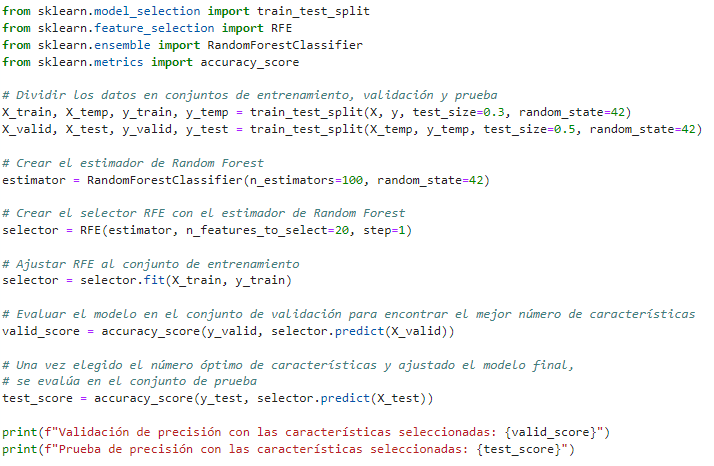
Tabla 6. Características Seleccionadas con Información Mutua.

#### Eliminación De Características Recursiva (RFE)

El algoritmo Recursive Feature Elimination (RFE) se utilizó junto con un clasificador de bosque aleatorio (Random Forest Classifier) para mejorar un modelo predictivo de clasificación.El uso de RFE puede capturar la importancia de las características en un problema de clasificación complejo y RFE puede iterativamente optimizar el conjunto de características para mejorar la precisión del modelo. El proceso de eliminación de características recursiva tiene una serie de pasos:

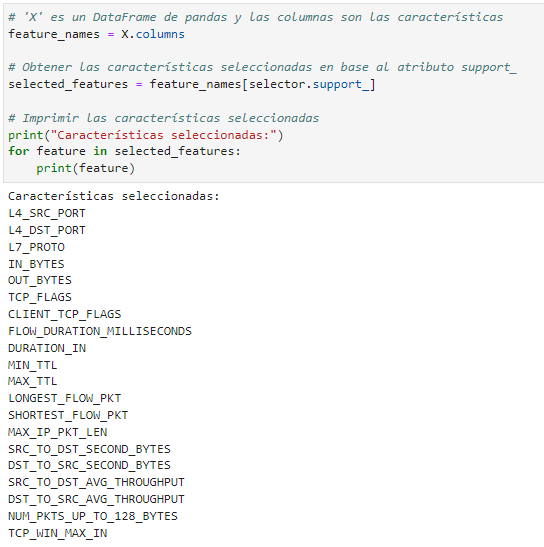
1. Entrenar Modelo Base: se entrena un modelo con todas las características del conjunto de datos. El tipo de modelo utilizado depende del problema y puede ser cualquier algoritmo de aprendizaje automático.
2. Ranking de Características: Después de entrenar el modelo base, se obtiene una puntuación o ranking de importancia para cada característica.
3. Eliminación de Características: Se elimina la característica menos importante según el ranking obtenido en el paso anterior.
4. Entrenamiento Iterativo: El modelo se vuelve a entrenar con el conjunto de datos reducido, que ahora contiene una característica menos. Este proceso se repite de manera iterativa, eliminando una característica en cada iteración, hasta que se alcanza un criterio de parada predefinido.
5. Evaluación del Rendimiento: Se evalúa el rendimiento del modelo en un conjunto de validación o prueba en cada iteración. Esto permite determinar si la eliminación de características ha mejorado o degradado el rendimiento del modelo.
6. Selección de Características Óptimas: Finalmente, se selecciona el conjunto óptimo de características que proporciona el mejor rendimiento en función del criterio de evaluación establecido.

Para la implementación de este algoritmo primero se realiza una división de los datos en conjuntos de entrenamiento, validación y prueba con un tamaño de prueba del 30% para el conjunto de entrenamiento y validación y luego divide el restante 70% por la mitad para la validación y prueba. Ajusta RFE al conjunto de entrenamiento para encontrar el mejor conjunto de características. Evalúa el modelo en el conjunto de validación para encontrar el mejor número de características utilizando la precisión como métrica. Evalúa el modelo final en el conjunto de prueba para obtener la precisión de las predicciones con las características seleccionadas.



Se inicializa un clasificador de bosque aleatorio con 100 árboles y una semilla aleatoria para la reproducibilidad (random\_state=42). Configura RFE con el clasificador de bosque aleatorio como el estimador base y especifica que se desean seleccionar las 20 características más importantes. El parámetro step=1 indica que RFE eliminará una característica en cada iteración para determinar el conjunto de características más importantes. Ajusta RFE al conjunto de datos completo X y y para obtener el ranking de las características.

Imprime la precisión de validación y la precisión de prueba para ver cómo se desempeña el modelo con el número de características seleccionado por RFE. Finalmente se obtiene y muestra las características seleccionadas por RFE.



|  |  |
| --- | --- |
| *CARACTERISTICAS* |  |
| L4\_SRC\_PORT | **MAX\_TTL** |
| L4\_DST\_PORT | **LONGEST\_FLOW\_PKT** |
| L7\_PROTO | **SHORTEST\_FLOW\_PKT** |
| IN\_BYTES | **MAX\_IP\_PKT\_LEN** |
| OUT\_BYTES | **SRC\_TO\_DST\_SECOND\_BYTES** |
| TCP\_FLAGS | **DST\_TO\_SRC\_SECOND\_BYTES** |
| CLIENT\_TCP\_FLAGS | **SRC\_TO\_DST\_AVG\_THROUGHPUT** |
| FLOW\_DURATION\_MILLISECONDS | **DST\_TO\_SRC\_AVG\_THROUGHPUT** |
| DURATION\_IN | **NUM\_PKTS\_UP\_TO\_128\_BYTES** |
| MIN\_TTL | **TCP\_WIN\_MAX\_IN** |

Tabla 7. Características Seleccionadas mediante RFE.

#### PCA

El Análisis de Componentes Principales (PCA) es una técnica estadística utilizada en proyectos de aprendizaje profundo principalmente para la reducción de dimensionalidad. En el contexto del proyecto que se está realizando el conjunto de datos contiene un gran número de variables o características en total 41, muchas de las cuales pueden estar correlacionadas o no ser relevantes para el problema en cuestión. PCA ayuda a simplificar estos conjuntos de datos al transformar las variables originales en un nuevo conjunto de variables, llamadas componentes principales, que son ortogonales (no correlacionadas) entre sí. Esto se logra al encontrar las direcciones (componentes) que maximizan la varianza en los datos.

La cantidad de variables reducida no solo facilita la visualización y el análisis de los datos, sino que también puede mejorar el rendimiento del modelo de aprendizaje profundo al reducir el ruido y la complejidad, acelerar el entrenamiento y evitar el sobreajuste.

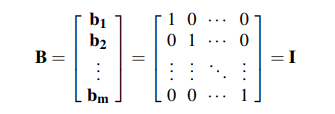
El PCA se puede realizar mediante dos métodos matriz de correlación y matriz de covarianzas. El método de la matriz de covarianza es aplicado a datos estandarizados. Los datos de X se estandarizan para que tengan una media (μ) de 0 y una desviación estándar (σ) de 1. Esto es importante para PCA, ya que está basado en la matriz de covarianza, que puede verse muy afectada por las escalas de las características.

Tratamos cada vez muestra (o ensayo experimental) como muestra individual en nuestro

conjunto de datos. En cada muestra de tiempo registramos un conjunto de datos que consta de múltiples mediciones (En este caso las llamamos características). En el conjunto de datos cada registro o traza de tráfico capturado puede expresarse como un vector de N columnas.

Cada muestra x es un vector de n dimensiones, es decir que traza será un vector en un espacio vectorial. En algebra lineal una base canónica es un conjunto de vectores que permite describir o representar otros vectores de manera sistemática y eficiente en términos de coordenadas. Por ejemplo, un vector cualquiera en R3 se puede representar de forma única mediante una combinación lineal de los vectores de la base canónica.

En algebra lineal esta base canónica puede ser expresada como una matriz de m x m donde cada fila será un vector ortonormal de m componentes. Cada registro o traza en un conjunto de datos puede ser expresando como una combinación lineal de bi.



Así que PCA es un método para buscar una base que sea una combinación lineal de la base original y nos permita reexpresar nuestro conjunto de datos.

La matriz de covarianza Cx es una herramienta que resume cómo cada par de variables en un conjunto de datos se relaciona entre sí. Cada elemento Cij de esta matriz indica la covarianza entre la variable i y la variable j. Si tenemos, por ejemplo, un conjunto de datos con tres mediciones (X, Y, Z), la matriz de covarianza Cx podría verse así:

El objetivo al hallar los Componentes principales es minimizar la redundancia medida por la magnitud de la covarianza y maximizar la medida de la varianza. Para esto se debe crear una matriz CY .

El algoritmo de PCA los que hace es: 1. Selecciona la dirección que maximiza la varianza en los datos en el espacio m-dimensional. 2. Encuentra la siguiente dirección que maximiza la varianza y es ortogonal a la anterior. 3. Repite el proceso hasta obtener el número deseado de componentes. El resultado será una base:

Esta base serán los componentes principales del conjunto de datos original.

Para realizar este procedimiento se utiliza la descomposición de eigenvectores.

El objetivo será encontrar alguna matriz ortonormal P en Y = PX tal que es una matriz diagonal. Las filas de P son los componentes principales de X.

|  |  |
| --- | --- |
|  |  |

La reducción de dimensionalidad del conjunto de datos mediante la técnica de PCA es diferente a las técnicas aplicadas anteriormente ya que es como crear características nuevas a partir de las originales y no solamente seleccionar entre las ya existentes según un criterio especifico. El análisis de componentes principales como se explico nos da como resultado una serie de combinaciones lineales de las características originales las cuales explican cada un un porcentaje de varianza de la variable objetivo. La biblioteca de scikit-learn cuenta con una función para el calculo de los componentes principales y además se define una función auxiliar con el fin de grafica la varianza explicada de cada uno de los componentes principales y la varianza acumulada.

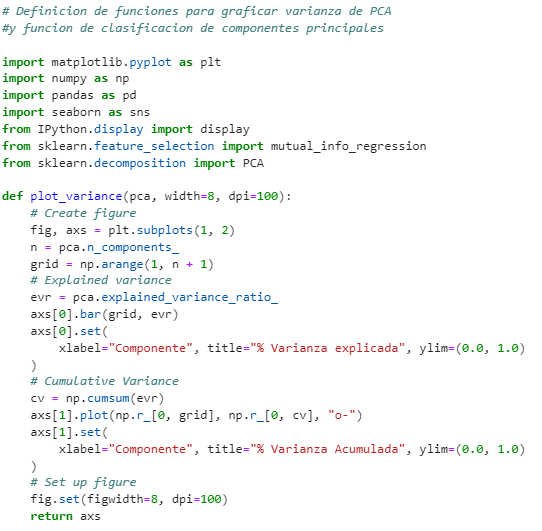


Figura 10. Funciones auxiliares Análisis de componentes Principales.

Se hace una selección de características y una estandarización de las características originales.

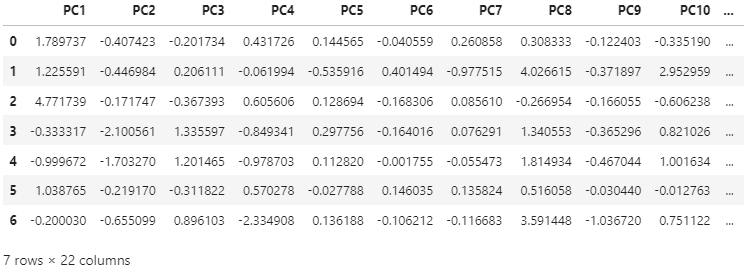


Figura 11. Cargas de los componentes Principales.

La figura representa las cargas de los componentes principales y cada fila representa una variable o característica original del conjunto de datos, y el número asociado es la carga de esa característica en PC1. Las cargas nos dicen cómo cada característica contribuye al componente principal.

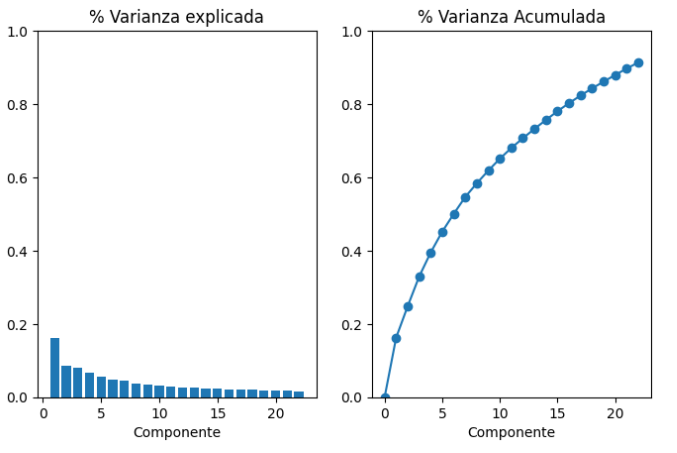


Figura 12. Varianza explicada y Acumulada de cada componente principal.

Finalmente, ya aplicadas las técnicas de selección de características se tienen cuatro grupos de características con los que se pueden entrenar el modelo. Sin embargo, también es necesaria la normalización de los primeros tres grupos y la división del conjunto de datos.

#### Normalización

La estandarización de características se realizó utilizando la función MinMaxScaler de la librería ScikitLearn. Usualmente, se escalan las características para que todas tengan el mismo rango, típicamente [0,1], para que ninguna característica domine sobre las otras simplemente debido a su escala. Este proceso puede ayudar a ciertos algoritmos (como las redes neuronales, SVM, K-means, etc.) a converger más rápidamente y/o a actuar de manera más eficiente. Cuando se trabaja con características que tienen escalas muy diferentes, algunos algoritmos pueden tener dificultades para converger. Esto significa que pueden requerir más iteraciones o tiempo para llegar a una solución óptima o pueden no converger en absoluto. La normalización también se realizó antes del análisis de componentes de componente principales PCA para reducir la dimensionalidad del Dataset.

La transformación de la función MinMaxScaler utiliza las siguientes ecuaciones:

La anterior es la fórmula de normalización ‘MinMax’ consiste en restar el valor mínimo de cada característica (columna) de cada observación y luego dividir por el rango de esa característica (la diferencia entre el valor máximo y mínimo). El resultado de esta operación es que cada característica se escala para que sus valores estén entre 0 y 1.

Después de Normalizar los datos se re-escalan los datos normalizados a un nuevo rango definido por los valores ‘min’ y ‘max’, que representan el rango deseado para los datos escalados.

#### División del conjunto de datos

La división de un conjunto de datos en entrenamiento, validación y prueba es una estrategia que busca abordar diferentes aspectos del proceso de modelado.

**Conjunto de Entrenamiento:** Este conjunto se utiliza para entrenar el modelo. Durante el entrenamiento, el modelo intenta aprender patrones o relaciones entre las características (variables independientes) y la variable objetivo (variable dependiente).

**Conjunto de Validación:** Este conjunto se utiliza una vez que el modelo ha sido entrenado, para evaluar su rendimiento, afinar y seleccionar modelos. Si es necesario, ajustar sus hiperparámetros. Usando el conjunto de validación, podemos evaluar cómo se desempeña el modelo en datos no vistos durante el entrenamiento.

**Conjunto de Prueba:** Este conjunto se utiliza para evaluar el rendimiento final del modelo. Después de seleccionar y afinar el modelo con los conjuntos de entrenamiento y validación, utilizamos el conjunto de prueba para obtener una estimación imparcial del rendimiento del modelo. Este rendimiento es una buena aproximación de cómo se comportaría el modelo en el "mundo real" con datos totalmente desconocidos.

Los Beneficios de esta división son evitar el Sobreajuste ya que, al tener un conjunto de validación separado, se puede evitar ajustar el modelo inconscientemente al conjunto de prueba. Sin un conjunto de validación, podríamos terminar con un modelo que se desempeña muy bien en el conjunto de prueba simplemente porque se ha ajustado demasiado a él. También para tener una estimación imparcial del rendimiento dado que nunca se usa en el proceso de entrenamiento o validación, ofrece una imagen clara de cómo se desempeñaría el modelo en situaciones reales con datos no vistos.

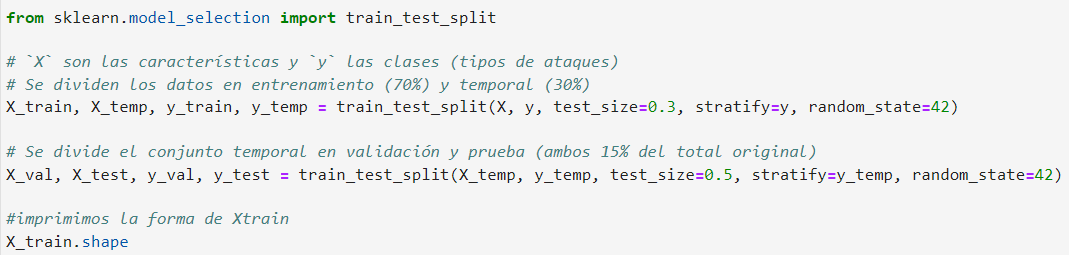


Figura 13. División de conjunto de datos. Entrenamiento 70%. Validación 15%. Prueba 15%

|  |  |  |
| --- | --- | --- |
|  |  |  |

Figura 14. Distribuciones Subconjuntos entrenamiento, validación y prueba.

## Implementación De Un Algoritmo De Aprendizaje Supervisado Basado En Una Red Neuronal Para Modelar El Conjunto De Datos.

Una taxonomía hecha del año 2023 sobre los métodos utilizados para la detección de ataque DDoS [] se describen algunas de las arquitecturas de redes neuronales que han tenido mejores resultados en la detección de este tipo de ataques. Según esta clasificación son las redes neuronales convoluciones CNN y las redes neuronales basadas en bloques de memoria a corto plazo LSTM las redes neuronales que han conseguido los mejores resultados en métricas de evaluación. Se realizo una comparativa entre tres tipos de redes neuronales: CNN, DNN, LSTM esto con el fin de elegir cuál de estas arquitecturas tiene mejores resultados para la detección de Ataques DDoS con el conjunto de datos disponible.

En la programación de redes neuronales existen diferentes herramientas para la construcción de las redes neuronales. La Implementación de la red neuronal se realizó mediante biblioteca de diferenciación automática Tensor Flow y la biblioteca de composición de capas para redes neuronales Keras. Mediante estas bibliotecas se puede definir la arquitectura de nuestra red neuronal como una serie de capas densas apiladas o completamente conectadas que serán añadidas al modelo junto con una función de perdida que evalúa el rendimiento y la función de optimización que ajusta los parámetros para minimizar la función de perdida utilizando el algoritmo de descenso del gradiente o alguna de sus variantes.

#### Arquitecturas y capas

Las arquitecturas implementadas tienen las siguientes capas:

**Conv1**: Esta capa aplica operaciones de convolución a los datos de entrada con el fin de filtrar y detectar características relevantes.

**Maxpooling1D**: Esta capa que reduce la dimensionalidad de los datos al seleccionar el valor máximo en ventanas de datos. Esto resalta las características más importantes y reduce el costo computacional.

**Flatten**: Esta capa convierte una salida tridimensional en un vector unidimensional. Esto se hace tomando todos los valores de la salida y colocándolos uno al lado del otro en un solo vector.

**Dense**: Estas son capas completamente conectadas que realiza operaciones lineales seguidas de una función de activación.

**Dropout**: Esta capa aleatoriamente desactiva un porcentaje de las unidades de neuronas durante el entrenamiento para prevenir el sobreajuste.

**LSTM:** Estas capas LSTM son capaces de recordar información de largo plazo y aprender dependencias a largo plazo en los datos. Fueron diseñadas para solucionar el problema de desvanecimiento y de explosión del gradiente en redes neuronales recurrentes.

**BatchNormalization:** Esa capa estandariza ajustando las medias y desviaciones estándar de las activaciones las activaciones de las capas anteriores acelerando el entrenamiento y a mejorando la generalización del modelo.

|  |  |  |
| --- | --- | --- |
|  |  |  |

Tabla 8. Arquitecturas Implementadas para comparativa.

#### Hiperparámetros

Los hiperparámetros son valores externos al conjunto de datos y que son definidos por el desarrollador de la red neuronal con el fin controlar el proceso de aprendizaje.

|  |  |  |
| --- | --- | --- |
| DNN | LSTM | CNN |
| Dropout\_rate: Tasa de abandono, porcentaje de neuronas que se apagan en una capa para evitar sobreajuste. | Dropout rate: Tasa de abandono, porcentaje de neuronas que se apagan en una capa para evitar sobreajuste. | Kernel\_size: Matriz bidimensional que se multiplica con los valores de entrada, para luego sumarse y obtener un valor de salida o feature map. |
| Neurons\_layer1: Número de neuronas en la primera capa densa. | Activation function: Permite a los modelos aprender patrones no lineales. | Activation function: Permite a los modelos aprender patrones no lineales. |
| Neurons\_layer2: Número de neuronas en la segunda capa densa. | Neuron\_Layer1\_LSTM: Numero de unidades de cálculo que realiza varias operaciones para procesar secuencias de datos y aprender representaciones temporales. | Dropout Rate: Tasa de abandono, apaga neuronas de capas densas para evitar sobreajuste. |
| Epochs: Número de veces que todo el conjunto de datos de entrenamiento se pasa a través de la red neuronal. | Momentum: permite guardar información de gradientes anteriores lo que permite una convergencia más rápida | Neuron\_Layer\_Dense: Numero de neuronas que realizan operaciones en los datos de entrada ajustando pesos y sesgos. |
| Batch\_size: número de ejemplos de entrenamiento que se utilizan en cada paso de actualización de los pesos durante una época. | Learning Rate: rapidez a la que se actualizan los parámetros o pesos y sesgos de la red | Pool size: tamaño de la ventana que se desliza sobre la entrada para realizar el muestreo máximo. |
| Learning Rate: rapidez a la que se actualizan los parámetros o pesos y sesgos de la red | Epochs: Número de veces que todo el conjunto de datos de entrenamiento se pasa a través de la red neuronal | Optimizer: Algoritmo que ajusta los pesos de las conexiones entre las neuronas con el objetivo de minimizar la función de pérdida o error |
| Activation Function: Permite a los modelos aprender patrones no lineales. | Batch\_size: número de ejemplos de entrenamiento que se utilizan en cada paso de actualización de los pesos durante una época. | Epochs: Número de veces que todo el conjunto de datos de entrenamiento se pasa a través de la red neuronal |
| Optimizer: Algoritmo que ajusta los pesos de las conexiones entre las neuronas con el objetivo de minimizar la función de pérdida o error | Neuron\_Layer\_Dense: Numero de neuronas que realizan operaciones en los datos de entrada ajustando pesos y sesgos. | Batch\_size: número de ejemplos de entrenamiento que se utilizan en cada paso de actualización de los pesos durante una época. |

Tabla 9.Hiperparametros de Arquitecturas Implementadas.

Las tres arquitecturas implementadas CNN, DNN, LSTM de prueba para la detección de Ataques DDoS, tienen una serie de funciones que sirven como esqueleto de cada uno de los modelos. Se describirá de forma general la estructura y las funciones que la componen:

* **Sequential():** Esta función permite crear un modelo secuencial donde las capas se agregan una tras otra en secuencia.
* **Layer():** Define las capas a travez de las cuales pasaran los datos con el fin de ajustar los pesos y sesgos del modelo para realizar las predicciones.
* **compile():** Configura la función de pérdida y el optimizador que se utilizarán durante el entrenamiento, junto con métricas adicionales para evaluar el rendimiento del modelo.
* **summary():** Proporciona una descripción resumida del modelo, incluyendo el número de parámetros y la forma de las capas. .
* **fit():** Entrena el modelo utilizando datos etiquetados, ajustando los pesos de las capas para minimizar la función de pérdida.
* **history:** Almacena el registro del proceso de entrenamiento, incluyendo la pérdida y las métricas en cada época, lo que permite evaluar el rendimiento del modelo a lo largo del tiempo.

En definitiva, estas funciones y modelos se utilizan para compilar, entrenar y evaluar modelos dependiendo de las características de los datos de entrada y los requisitos del problema.



Figura 15. Arquitectura DNN implementada.

El cambio en la especificación de input\_shape en la capa Conv1D es una adaptación necesaria debido a que las redes convolucionales están diseñadas para operar sobre datos secuenciales o temporales.. En el caso de Conv1D, se espera que los datos de entrada tengan la forma de (longitud\_de\_la\_secuencia, numero\_de\_canales). Aquí, X\_train.shape[1] representa la longitud de la secuencia o el número de pasos de tiempo, y el 1 indica que hay un solo canal de características.

En cuanto a las redes LSTM, también se utilizan para procesar secuencias de datos en pasos de tiempo, y el input\_shape debe coincidir con la estructura de estos datos. En input\_shape=(X\_train.shape[1], 1), X\_train.shape[1] es el número de pasos de tiempo en la secuencia y 1 es el número de características en cada paso de tiempo, lo que implica que se está trabajando con secuencias univariadas. El ‘return\_sequences=True’ es necesario cuando se tiene una arquitectura de red en la que hay múltiples capas LSTM apiladas.Las redes LSTM en Keras esperan datos de entrada con una forma específica, que es: (num\_samples, num\_timesteps, num\_features).

* **num\_samples:** El número total de muestras independientes en tu conjunto de datos, es decir, el número total de secuencias.
* **num\_timesteps**: La longitud de cada secuencia de tiempo. Esto es, cuántos pasos temporales se consideran para cada muestra.
* **num\_features:** El número de características observadas en cada paso temporal. En otras palabras, es la dimensión de los datos en cada paso temporal.



Figura 16. Arquitectura CNN implementada.

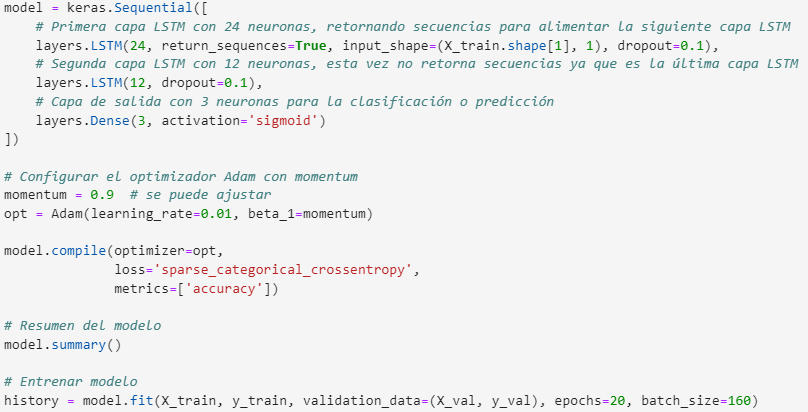


Figura 17. Arquitectura LSTM implementada.

#### Métricas

|  |  |  |  |
| --- | --- | --- | --- |
|  | Accuracy Entrenamiento | Accuracy  Validación | Accuracy  Prueba |
| CNN |  |  |  |
| DNN |  |  |  |
| LSTM |  |  |  |

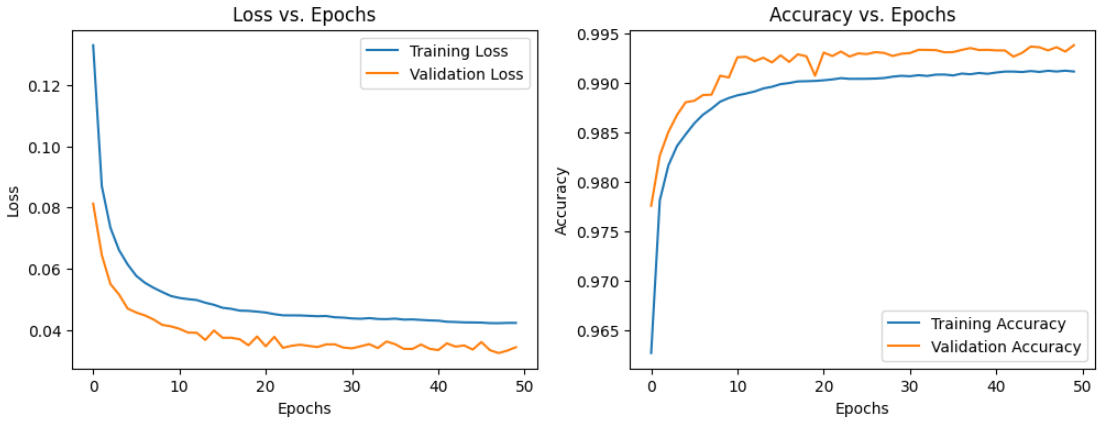


Figura 18. Curvas de aprendizaje optimización y rendimiento Arquitectura CNN.

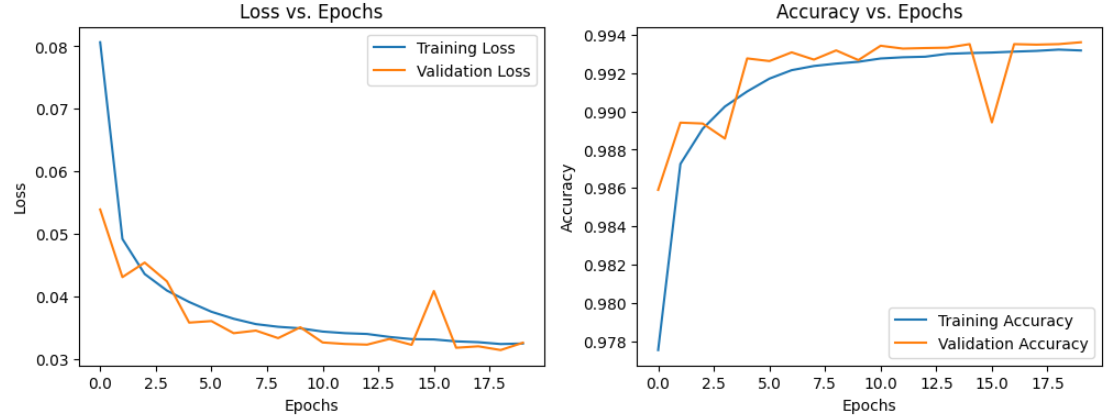


Figura 19. Curvas de aprendizaje optimización y rendimiento arquitectura DNN.

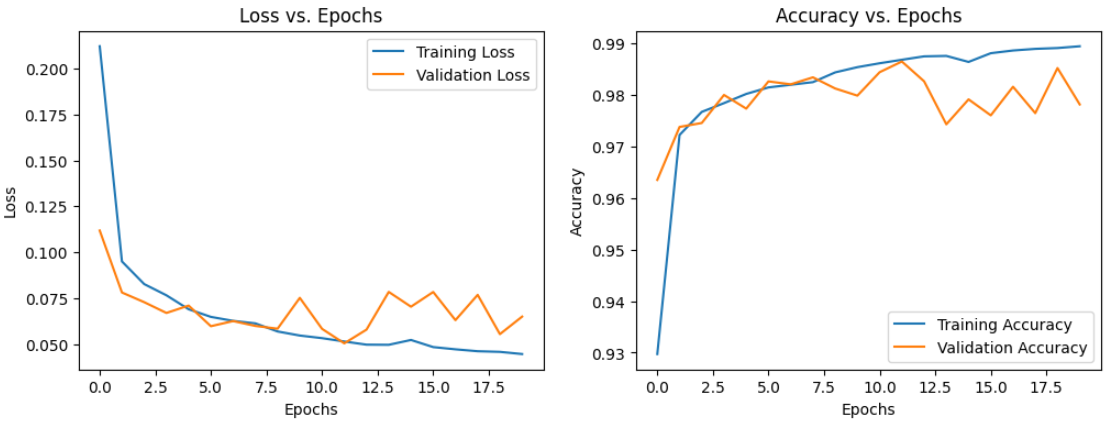


Figura 20. Curvas aprendizaje optimización y rendimiento arquitectura LSTM.

## Análisis Estadístico De Los Resultados Obtenidos Utilizando El Modelo Y Comparación De Su Desempeño Frente A Otra Técnica Reportada En El Estado Del Arte.

.

# Resultados

Se esperan 4 sub-secciones de resultados: Propuesta definitiva (Diseño definitivo) de solución al problema planteado; Restricciones de la Solución; validación de la solución (resultados de pruebas hechas a la solución propuesta para verificar el cumplimiento de los requerimientos); y Evaluación del impacto de la solución propuesta

Modelo final seleccionado

Análisis estadístico descriptivo del modelo

Restricciones y limitaciones del modelo propuesto

Comparación del desempeño del modelo con otro modelo desarrollado

Prueba uso del modelo

# Conclusiones y Recomendaciones

Presentación resumida de los aspectos encontrados en el proyecto que van más allá de los aspectos factuales y específicos del mismo. Esta sección se nutre de las implicaciones del proyecto en la práctica, de las lecciones metodológicas aprendidas en la ejecución del proyecto, de las limitaciones del proyecto, de los aspectos no resueltos del proyecto y nuevas preguntas creadas a partir del mismo, propuestas de nuevos proyectos a partir de los resultados actuales.

# Referencias

[x1] Amazon Web Services. (n. d.). ¿Qué es el Internet de las cosas (IoT)? Recuperado de https://aws.amazon.com/es/what-is/iot/

# Anexos

De ser necesario, se pueden preparar Anexos en orden alfabético (Anexo 1, Anexo 2, etc.). Pueden ser utilizados para presentar mayor detalle sobre algún aspecto del documento sin romper el hilo conductor del texto principal. Por ejemplo, se puede usar para presentar el formato de encuesta que se espera usar en el proyecto, o un protocolo de observación, etc.

**PAUTAS PARA LA REDACCIÓN DEL DOCUMENTO**

Se propone el seguimiento de las normas de la APA o IEEE. En general se espera que la ortografía sea impecable (no confiarse del corrector de MS Word). Las propuestas deben estar bien presentadas; se debe cuidar: la unidad del estilo del texto, títulos y sub-títulos a lo largo del documento.

Cada capítulo debe comenzar en una nueva página. El fin de una sección y el encabezado de la próxima no deben ser separados por espacios adicionales. Cuándo una sección comienza al final de la página, ésta debe ser trasladada a la página siguiente si el primer párrafo de la sección no alcanza a tener dos líneas de texto.

La extensión del documento no debe superar las 60 páginas sugerido (70 páginas máximo) sin incluir los preliminares (portada, contraportada, índices, dedicatoria, glosario, siglas); Listado de referencias bibliográficas ni Anexos

**Estilo y formato:**

***Redacción****:* La propuesta debe ser redactada en tercera persona.

***Texto y formato de los párrafos:***La letra del documento es Times New Roman tamaño 12 o equivalente, el interlineado es sencillo. Además, se deben parametrizar el párrafo con los siguientes valores:

* Alineación: Justificada.
* Espaciado: Anterior: 12 pto, Posterior: 12 pto, Interlineado: 1,0
* Viñetas y significado de variables y parámetros de fórmulas: Espaciado (Anterior: 6 pto, Posterior: 6 pto)

***Títulos:*** A continuación, una descripción de los tipos de títulos y sus características:

* Título 1 (Sección primaria o Capítulo): Times New Roman Negrita tamaño 14
* Titulo 2 (sección secundaria o Sub-capítulo): Times New Roman Negrita tamaño 13.
* Título 3 (sección terciaria): Times New Roman Negrita tamaño 12

***Número de Página***Todas las páginas deben estar numeradas excepto la portada. Las páginas previas al capítulo de introducción deben ser numeradas en números romanos (i, ii, iii, iv, v,…) en minúsculas, mientras que a partir de la introducción debe ser arábiga (1,2,3…).

***Notas al pie de página****.* Las notas al pie de página tienen por finalidad brindar una información extra o aclarar un concepto dentro del cuerpo principal. Son una herramienta útil para agregar contenido que para un lector lego puede ser de utilidad para interpretar o entender lo que se está presentando. Las notas al pie de página deberán ser referenciadas utilizando un superíndice y la nota al pie de página deberán tener un tamaño de letra entre 10 a 8. Utilizar la funcionalidad pie de página para esta parte del texto.

***Figuras y Tablas en el texto.***La resolución adecuada de fotos y figuras que permitan leer y entender el mensaje que se desea entregar. Toda tabla y figura debe tener un título. Además, toda tabla y figura debe ser anunciada en el texto del documento antes de aparecer en el mismo, no olvide referenciarla

*Ejemplo:*

Se evaluó la simulación de la marcha normal, al igual que las diferentes fases de un ciclo, trazando la flexión y extensión de la cadera contra la flexión y extensión de la rodilla. La cadera está en el eje *x* y la rodilla está en el eje *y*; La magnitud de los gráficos se indica en ángulos. El resultado del ciclograma mostrado en la Figura 6 concuerda con los resultados reportados en la literatura, ya que los ángulos de rodilla y cadera están dentro de los límites de los valores observados en estudios previos.

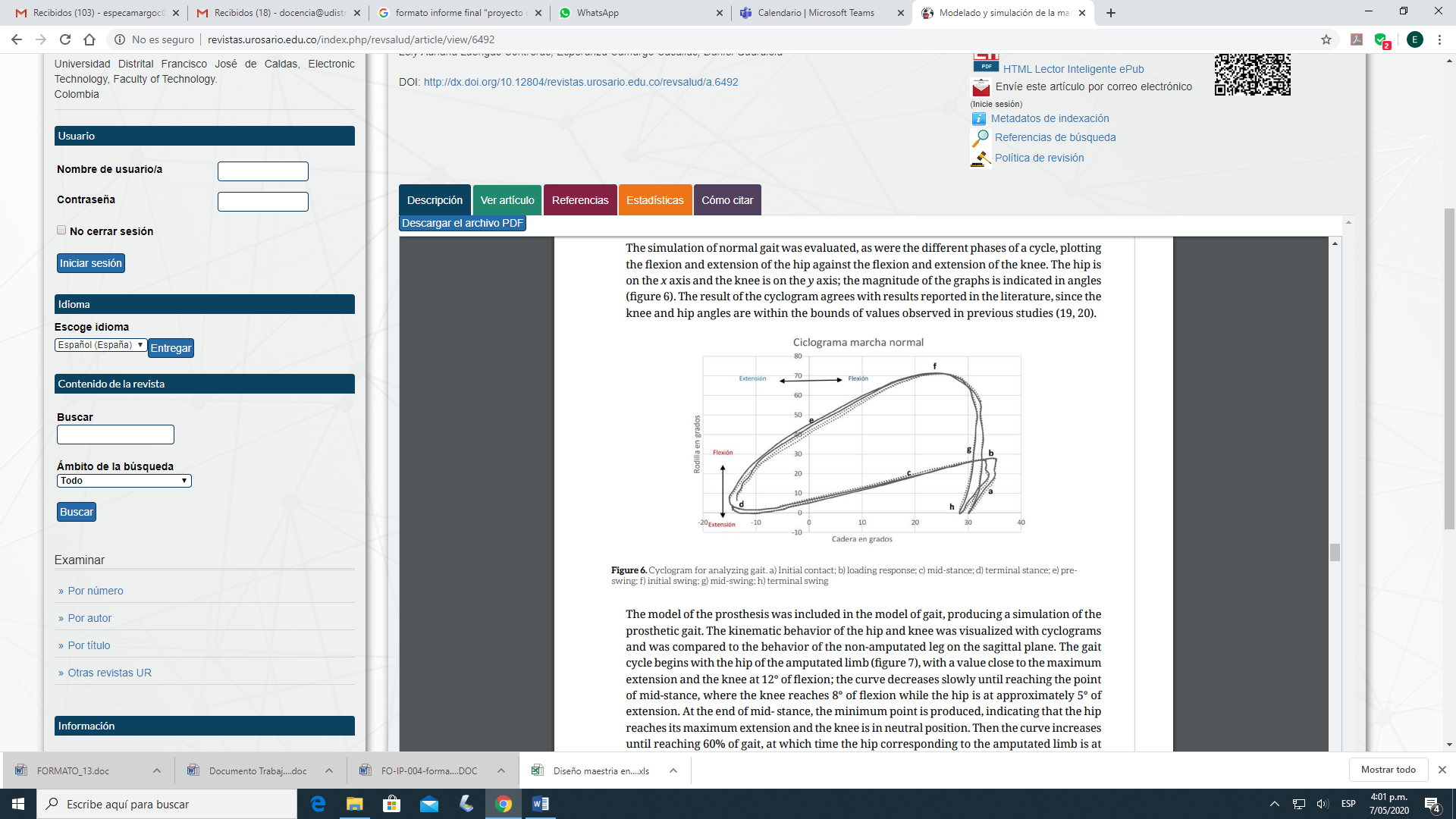


Figura 6. Ciclograma para analizar la marcha. a) Contacto inicial; b) Respuesta de carga; c) Apoyo media; d) Apoyo terminal; e) Prebalanceo f) Balanceo inicial; g) Balanceo medio; h) Balanceo terminal. (Camargo, 2018)

En el caso de las tablas se sigue un criterio similar a las figuras, el tamaño mínimo de la letra al interior de la tabla es 11.

Ejemplo:

Como se muestra en la Tabla 1, los valores de *t* para un nivel de confidencia del 95%, depende de los grados de libertad asociados con la desviación estándar.

|  |  |
| --- | --- |
| **Grados de Libertad** | **Nivel de confidencia** |
| 2 | 4.303 |
| 4 | 2.776 |
| 6 | 2.447 |

Tabla 3. Variables de medición. Fuente: El Autor

***Ecuaciones y Fórmulas.***Las ecuaciones deben ser elaboradas en un editor de ecuaciones apropiado. Si utiliza Word, use o bien Microsoft Editor de Ecuaciones o MathType. Numere las ecuaciones consecutivamente, deben estar centradas, por orden de aparición, con números arábigos entre paréntesis justificado al margen derecho. Las ecuaciones al igual que las figuras y tablas deben ser anunciadas en el texto del documento antes de aparecer en el mismo.

Ejemplo:

La curva de frecuencia (1) de los valores medios está centrada alrededor del valor límite medioy tiene un factor de escala

|  |  |
| --- | --- |
|  | (1) |

***Citaciones***Las citaciones son referencias a textos realizados por otras personas, las podemos clasificar en tres grandes grupos:

* Formales: donde se transcriben literalmente textos. Es imprescindible que éstas sean colocadas entre comillas e indicar su fuente.
* Conceptuales: en este caso se reproduce el concepto utilizando palabras propias, es una síntesis personal de una idea basándose en otra idea, es necesario referenciar usando normas APA o IEEE.

***Referencias*** Las referencias bibliográficas constituyen una parte importante del texto donde se detallan todas las fuentes consultadas en el trabajo que se está presentando. Es importante detallarlo adecuadamente para que posteriormente otros lectores puedan realizar búsquedas de esas referencias.

Las referencias deberán ser ordenadas considerando el apellido del primer autor, en caso de que el autor tenga más de un artículo o trabajo referenciado en el informe se debe ordenar por el año de publicación. Para mayor información sobre la forma de referenciar se les sugiere que revisen las normas APA o IEEE.

***Especificando las referencias***

Considere los siguientes ejemplos que se detallan a continuación para realizar las referencias. El autor debe dejar un espacio entre una y otra referencia.

**Referencias de libros:**

Apellido autor, Iniciales nombre autor, (Año), *Título en cursiva,*Ciudad y país, Editorial.

Hacyan, S., (2004), *Física y metafísica en el espacio y el tiempo. La filosofía en el laboratorio*, México DF, México: Fondo nacional de cultura económica.

**Referencia de páginas web:**

Apellido, A. A. (Fecha). Título de la página. Lugar de publicación: Casa publicadora. Dirección de donde se extrajo el documento (URL).

Argosy Medical Animation. (2007-2009). Visible body: Discover human anatomy. New York, EU.: Argosy Publishing. Recuperado de <http://www.visiblebody.com>.

Nombre del artículo, (s. f). En xxxxx. Recuperado el X de MES de AÑO de <http://xxx.xxxxxxxxxx.xx/xxxx/xxx>.

**Referencia revistas científicas:**

Apellido autor1, Iniciales nombre autor1., Apellido autor2, Iniciales nombre autor2., …y Apellido autorN, Iniciales nombre autor. (Año). Nombre del trabajo, revista, volumen(número del volumen), paginas.

Coruminas, M., Ronecro, C., Bruguca, E., y Casas, M. (2007). Sistema dopaminérgico y adicciones, *Rev Mukuel*, 44(1), 23-31.

Arshinder, K., Kanda, A., y Deshmukh, S. G. (2011). A Review on Supply Chain Coordination: Coordination Mechanisms, Managing Uncertainty and Research Directions. In T.-M. Choi & T. C. E. Cheng (Eds.), Supply Chain Coordination under Uncertainty SE - 3 (pp. 39–82). Springer Berlin Heidelberg. doi:10.1007/978-3-642-19257-9\_3.

**Referencias congresos y seminarios:**

Autor, A., & Autor, A. (año). Título de la ponencia, Título del simposio o congreso. Simposio o conferencia llevado a cabo en el congreso Nombre de la organización, Lugar.

Rojas, C., y Vera, N. (2013). ABMS Automatic BLAST for Massive Sequencing), 2° Congreso Colombiano de Biologia Computacional y Bioinformática CCBCOL. Congreso llevado a cabo en Manizales, Colombia.