Detección de ciberataques mediante aprendizaje profundo

Christian Garzón
GITA Lab
Faculty of Engineering
University of Antioquia UdeA
Medellín, Colombia
christianc.garzon@udea.edu.co

***Index Terms*—cyber-attack, threats, deep learning, RNN, LSTM, imbalance dataset, balance dataset, perceptron.**

I. CONTEXTO

La seguridad es un factor esencial en cada actividad humana, especialmente en lo que se refiere a la información privada. Un activo de información puede encontrarse digitalmente, cambiando al concepto de ciberseguridad. Además, esta información pertenece a un individuo o a una organización, y parte de ella debe permanecer confidencial. Según Kaspersky [1], la ciberseguridad es la práctica de defender los dispositivos informáticos como ordenadores, servidores, teléfonos móviles, dispositivos móviles, redes y datos. Sin embargo, ¿de qué se defiende la ciberseguridad? Las ciberamenazas se han ido desarrollando desde los orígenes del ordenador; hoy en día, se dirigen a las redes, las aplicaciones, la información y las operaciones. Como explica Kaspersky, en el ámbito de la ciberseguridad se desarrollan y utilizan a diario muchos ciberataques; los actores están motivados por el dinero, la política, el terrorismo y otros [1].

La seguridad de la información ha pasado a ser un objetivo inestimable para cualquier organización, aunque implique un aumento de los precios. Actualmente, los sistemas de comunicación tradicionales poseen ciertas vulnerabilidades, las cuales podrían ser usadas maliciosamente por un tercero; ahora bien, los dispositivos de ciberseguridad, también tradicionales, podrían aumentar los retrasos adicionales y agregar un tráfico duplicado a la red. Lo anterior, es una situación que no puede ser escalable en determinadas infraestructuras.

Hoy en día, el IoT (en inglés Internet of Things) y aplicaciones industriales utilizan las tecnologías de comunicación tradicionales (basadas en TCP/IP o Ethernet) para supervisar y controlar sus sistemas. Evidentemente, esto hace que hereden indirectamente vulnerabilidades críticas de ciberseguridad. Sin embargo, nuevas técnicas y modernas herramientas, que se enfocan en la detección y mitigación de ciberataques mediante la clasificación de los flujos de red, incluyen el uso de algoritmos de machine learning, los cuales podrían alcanzar un mejor rendimiento, incluso a menor costo. Hoy en día, los ataques más sofisticados y peligrosos pueden aprovechar las brechas en las redes, y cuanto más se hayan desarrollado, mejores sistemas de defensa deberá utilizar el objetivo atacado. Los principales ataques son:

- MALWARE.
  - Virus.
  - Troyanos.
  - Spyware.
  - Ransomware.
  - Botnets.
- PHISHING.
- MAN-IN-THE-MIDDLE.
- DENIAL-OF-SERVICE.

II. OBJETIVO DE MACHINE LEARNING

Para realizar la clasificación que indique si se está bajo un ataque o no (clasificación binaria), es necesario tener un contexto de los flujos de red previamente analizados. Los mencionados flujos de red pueden verse como una serie de entradas que generan diversas salidas, es decir una secuencia de flujos. Por lo anterior, se hace necesario el uso de técnicas de modelamiento secuencial.

III. DATASET

El dataset a usar se puede encontrar en [2], este contiene una serie de datos benignos y malignos (flujos de ataques) que son procesados con CICFlowMeter a partir de estampas de tiempo, IP fuente y destino, puertos fuente y destino, protocolo y ataque; a partir de estos se genera una cantidad de 80 características. Este dataset está etiquetado según el ataque. Los ciberataques usados en este dataset fueron Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet y DDoS. El tamaño del dataset varia según el día de prueba, lunes a viernes, para un total de unas 51.1 Gb con distribución Lunes: 11 Gb, Martes: 11 Gb, Miercoles: 13 Gb, Jueves: 7.8 Gb y V: 8.3 Gb; lo anterior en formato pcap, pudiendo trabajarse también en formato .csv que implicaría una reducción considerable en el tamaño del dataset a procesar. Las etiquetas y la cantidad de muestras de cada una se especifican en la figura 1.

Class Labels	Number of instances
BENIGN	2359087
DoS Hulk	231072
PortScan	158930
DDoS	41835
DoS GoldenEye	10293
FTP-Patator	7938
SSH-Patator	5897
DoS slowloris	5796
DoS Slowhttptest	5499
Bot	1966
Web Attack – Brute Force	1507
Web Attack – XSS	652

Fig. 1. Clases de etiquetas en dataset CICIDs2017 [2].

IV. NOTEBOOKS

El desarrollo de este proyecto se llevó a cabo mediante la implementación de tres notebooks ordenados secuencialmente, los cuales se especializan en una función especifica ilustrada a continuación:

1) Pre-procesamiento.
   a) Carga de datos y análisis inicial.
   b) Balanceo de datos y guardado de CSV's.
2) Perceptron.
   a) Carga de datos y análisis inicial.
   b) División de dataset en conjuntos de entreno, testeo y validación.
   c) Ensamblaje del modelo.
   d) Evaluación del sesgo.
   e) Entrenamiento del modelo y métricas de evaluación.
3) RNN y LSTM.
   a) Carga de datos y análisis inicial.
   b) División de dataset en conjuntos de entreno, testeo y validación.
   c) Ensamblaje de los modelo.
   d) Entrenamiento de los modelos.
   e) Evaluación de los modelos.
   f) Matrices de confusión.

Cada uno de los anteriores notebook tiene una sección que se encarga de subir las librerías necesarias para su ejecución. Se debe de tener en cuenta que en el notebook 1 (Pre-procesamiento) la base de datos a subir esta completa, donde originalmente esta se dividía en días. Sin embargo, hay una celda comentada que indica como sería el procedimiento para juntarlas en una sola. Además, en cada uno de estos notebooks se usan DataFrames para un procesamiento más limpio.

V. DESCRIPCION DE LA SOLUCION

Debido a la cantidad de ataques con los que cuenta el dataset, ya que no es especializado en uno solo, y al hecho de que al tener más muestras (con cierto grado de calidad cada una) podría haber un mejor resultado, se decide imponer dos clases al dataset BENIGNO 0 o MALIGNO 1. Por tanto, se tendrá dentro del sistema una entrada con las 80 características y una salida binaria, habiendo así un preprocesado, procesamiento y salida.

Aquellas etapas en busca de la solución se muestras a continuación:

- **PRE-PROCESAMIENTO**

Como se explicó anteriormente la base de datos [2] podrá descargarse en un formato PCAP o CSV, y vendrá dividida en ocho archivos, los cuales fueron unidos en el archivo "data.csv" para facilidad en el procesamiento. El total de muestras con el que cuenta la base CIC-IDS2017 es de 2'830.743 muestras, cada una con 80 características. La cantidad de muestras por etiquetas encontradas inicialmente se aprecian en la figura 2.

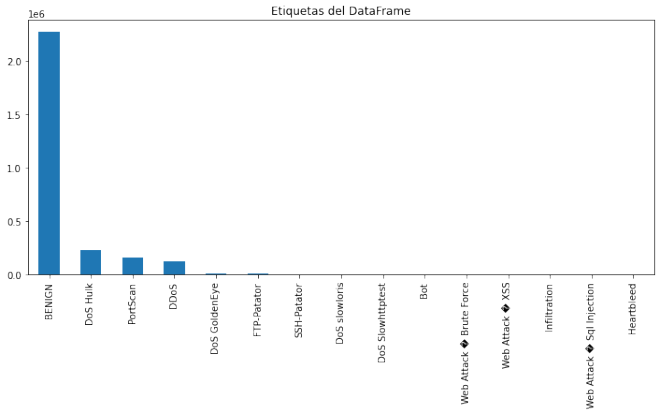

Fig. 2. Cantidad de muestras por etiqueta.

Desde este punto, puede notarse el desbalance que existe en la base de datos y la necesidad de trabajar en un balanceo o en técnicas especiales para dataset desbalancedados. Sin embargo, este proyecto busca realizar una clasificación binaria, 1 o 0, y para ello es necesario convertir, basados en la figura 2, la clase "BENIGN" en 0 y las demás clases en 1, teniendo como resultado la figura 3. Esta nueva base de datos será guardada con el nombre de "data_bin_desbalanceada.csv" y se usará en el notebook 2 Perceptron.

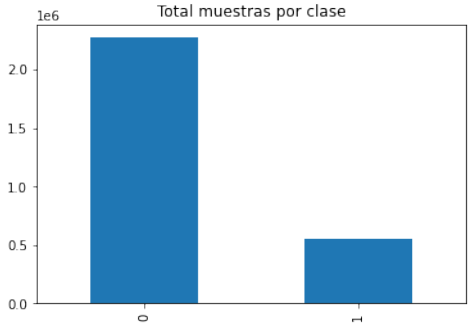

Fig. 3. Muestras por clase con desbalance.

Aún con el procedimiento realizado la etiqueta 0, la de datos benignos, seguirá siendo mayoritaria respecto a la 1, la de malignos, por casi cuatro veces la cantidad de datos. Por tal razón debe de realizarse un muestreo aleatorio de la clase 0, con el fin de balancear los datos y obtener una base de datos similar a la de la figura 4 respecto a la cantidad de muestras. Esta nueva base de datos será guardada con el nombre de "data_bin_balanceada.csv" y será necesaria para correr el notebook 3 RNN y LSTM.

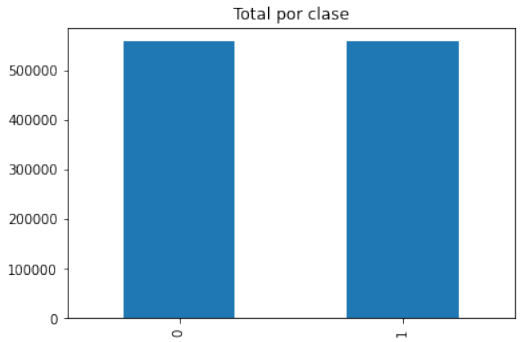

Fig. 4. Muestras por clase balanceadas.

- **IMPLEMENTACION DE MODELOS**

Se contará con tres modelos, los cuales son perceptron simple, Recurrent Neural Networks (RNN) y el Long-Short Term Memory (LSTM). Las salidas de los tres modelos serán binarias e indicarán simplemente si se trata de un ataque o no, pudiendo ser un aviso para un operador de seguridad en cierta red y trabajar así como un IDS (Intrusion Detection System).

- **EVALUACION DE LOS MODELOS**

La medición del desempeño de los modelos fue posible haciendo uso de métricas como:

PRECISION: El número de positivos que fueron reportados e identificados de forma correcta.

$$\frac{TP}{TP + FP} \tag{1}$$

RECALL: Indica el porcentaje de positivos reales que se identificaron teniendo en cuenta los falsos negativos.

$$\frac{TP}{TP + FP} \tag{2}$$

PRC: Precision-Recall Curve indica la relación entre las anteriores métricas Precision y Recall, detallando que tan bajas son las tasas de falsos positivos y falsos negativos.

MATRIZ DE CONFUSIÓN: Una métrica de desempeño que permitirá visualizar el desempeño del método usado. En esta se busca que los valores de la diagonal principal sea mayores notoriamente a los demás, tal como se expresa en verde en la tabla I.

TABLE I
MATRIZ DE CONFUSIÓN

		EXPECTED	
		True Positives TP	False Positives FP
P R E		False Negatives FN	True Negatives TN

VI. ITERACIONES

Los tres modelos a usar contarán con un tamaño de entrada equivalente al total de características de la base de datos, ochenta. Además tendrán una capa densa de 40 y 50 neuronas para el perceptron simple y RNN-LSTM respectivamente, al igual que un dropout de 40% y 50%. Por la naturaleza binaria del problema se uso una activación sigmoide.

En el caso de la compilación el modelo de perceptron simple usó un optimizador Adam y una función para minimizar la perdida de BinaryCrossentropy; de los cien epochs citados hubo un early stoping al 81. Por parte de RNN y LSTM en ambos se uso Adam y una función de perdida mean_squared_error y teniendo en cuenta su complejidad solo se usarán 5 epochs en ambos.

VII. RESULTADOS

La tabla II muestras los resultados obtenidos en el proceso de validación de los modelos, entre ellos destacando el de la RNN. Al tener una mayor precisión y una menor perdida.

TABLE II
RESULTADOS EN VALIDACIÓN DE LOS MODELOS

	Accuracy	Precision	Recall	PRC	Loss
Dense Perceptron	0.942	0.858	0.845	0.931	0.149
RNN	0.982	0.976	0.989	0.996	0.0194
LSTM	0.969	0.956	0.984	0.993	0.0274

También se procede a mostrar los resultados de las matrices de confusión en las figuras 5, 6 y 7. Para la matriz de confusión de la figura 5, las del perceptron, puedo notarse como en la diagonal principal hay valores del orden 10^5, mientras que los demás valores son del orden 10^4. Estos resultados son medianamente buenas respecto a la simplicidad del modelo.

Los modelos de las figuras 6 y 7 tienen en su diagonal principal valores del orden 10^5, comparados con los demás que tienen orden 10^3 se estaría hablando de mejores modelos.

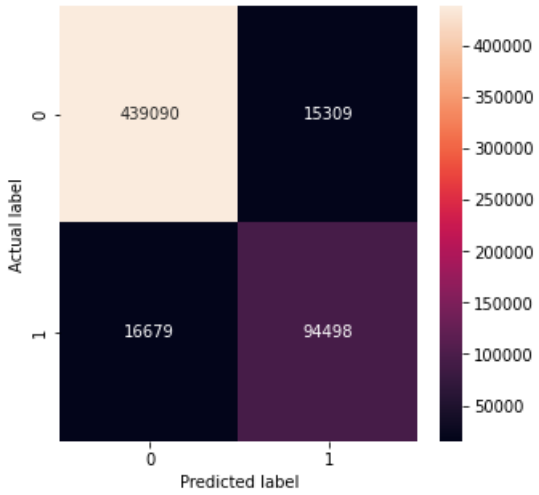

Fig. 5. Matriz de confusión de la capa de perceptrones.

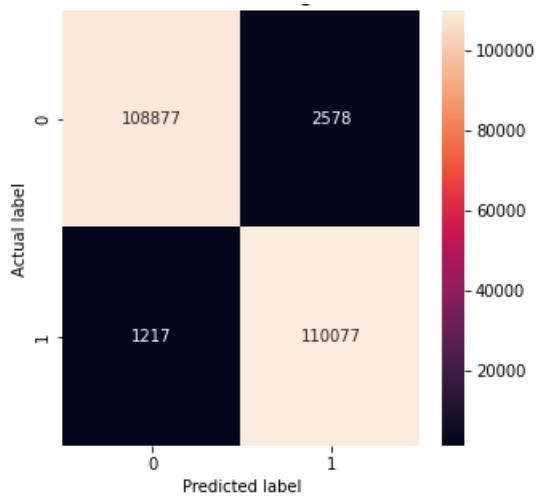

Fig. 6. Matriz de confusión de modelo RNN.

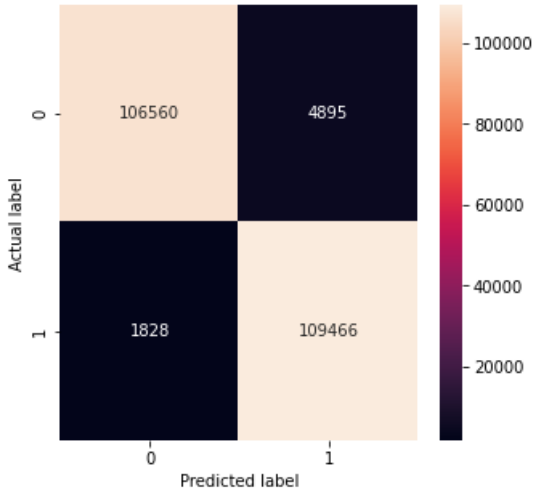

Fig. 7. Matriz de confusión de modelo LSTM.

VIII. REFERENCIAS Y RESULTADOS PREVIOS

Algunos autores ya han hecho uso de algoritmos como RNN (Recurrent Neural Network) y obtenido interesantes resultados. En [3] se obtuvieron resultados con un 0.998 y 0.919 en el F-Score para detección de incidentes y fraudes respectivamente. Por otra parte, en [4] haciendo uso de un dataset y con entradas solo basadas en IP destino y fuente alcanzó una detección del 0.912.

REFERENCES

[1] What is Cyber Security?. Kaspersky. [Online]. Available: https://www.kaspersky.com/resource-center/definitions/what-is-cyber-security.

[2] Panigrahi, Ranjit & Borah, Samarjeet. (2018). A detailed analysis of CICIDS2017 dataset for designing Intrusion Detection Systems. International Journal of Engineering & Technology. 7. 479-482.

[3] KP, Soman, et al. RNNSecureNet: Recurrent neural networks for Cyber security use-cases. arXiv preprint arXiv:1901.04281, 2019.

[4] BEN FREDJ, Ouissem, et al. CyberSecurity attack prediction: a deep learning approach. En 13th International Conference on Security of Information and Networks. 2020. p. 1-6.