INNOVARE
Revista de Ciencia y Tecnología
Disponible en CAMJOL - Sitio web: www.unitec.edu/innovare/

Original Article

Binary classification of malware by analyzing its behavior in the network using machine learning

Clasificación binaria de malware mediante el análisis de su comportamiento en la red mediante aprendizaje de maquina

Jean Carlo Soto[1]

Facultad de Ingeniería, Universidad Tecnológica Centroamericana, UNITEC, Tegucigalpa, Honduras

Article history:
Received: 29 October 2022
Revised: 22 March 2023
Accepted: 29 March 2023
Published: 15 April 2023

Keywords
Cybersecurity
Deep learning
Machine learning
Malware
Network

Palabras clave
Aprendizaje de maquina
Aprendizaje profundo
Malware
Red
Seguridad cibernética

ABSTRACT. Introduction. Every day we are exposed to all kinds of cyber-threats when we browse the internet, compromising the confidentiality, integrity, and availability of our devices. Cyber-attacks have become more sophisticated and cyber attackers require less technical knowledge to execute such attacks. An automated and well-defined process to counter these attacks becomes urgent. The study aim was to solve this problem. **Methods.** A model was developed to analyze the information in Packet Capture (PCAP) files and classify network connections as either benign or malicious (malware generated). This software used two methods: traditional machine learning algorithms and neural networks. Our experiments were carried out using the Intrusion Detection Evaluation Dataset (CICIDS2017), which contains labeled samples of PCAP files. We experimented using both raw and standardized data. The classification results were evaluated using recall, precision, F1-score, and accuracy metrics. **Results.** These were satisfactory for both methods, obtaining more than 95% in the F1-score and recall metric, indicating a low number of false negatives. **Conclusion**. It was found that data standardization had a favorable impact on all metrics and should be used carefully. Overall, our experiments showed that malicious network traffic can be successfully detected using automated methods achieving above 95% of F1-score in the K-Nearest Neighbors algorithm (K-NN) classifier.

RESUMEN. Introducción. Cada día estamos expuestos a todo tipo de ciberamenazas cuando navegamos por internet, comprometiendo la confidencialidad, integridad y disponibilidad de nuestros dispositivos. Los ciberataques se han convertido más sofisticados y los ciberatacantes requieren menos conocimientos técnicos para ejecutar dichos ataques. Un proceso automatizado y bien definido para contrarrestar estos ataques se vuelve urgente. El objetivo del estudio fue resolver este problema. **Métodos.** Se desarrolló un modelo para analizar la información en los archivos de Captura de paquetes (PCAP) y clasificar las conexiones de red como benignas o maliciosas (generadas por malware). Este software utilizó dos métodos: algoritmos tradicionales de aprendizaje de maquina y redes neuronales. Nuestros experimentos se llevaron a cabo utilizando el conjunto de datos de evaluación de detección de intrusiones (CICIDS2017), que contiene muestras etiquetadas de archivos PCAP. Se utilizó datos tanto crudos como estandarizados. Los resultados de la clasificación se evaluaron utilizando métricas de exhaustividad, precisión, puntuación F1 y precisión. **Resultados.** Estos fueron satisfactorios para ambos métodos, obteniendo más del 95% en las métricas de puntuación F1 y exhaustividad, lo que indica un bajo número de falsos negativos. **Conclusión.** Se encontró que la estandarización de datos tuvo un impacto favorable en todas las métricas y debe usarse con cuidado. En general, nuestros experimentos mostraron que el tráfico de red malicioso se puede detectar con éxito utilizando métodos automatizados que alcanzan más del 95% de la puntuación F1 en el Clasificador del Algoritmo de Vecinos Más Cercanos (K-NN).

1. Introduction

The International Telecommunications Union (ITU) develops the Global Cybersecurity Index (GCI). This was first launched in 2015 with 192 ITU member states and the state of Palestine to help these states to identify areas of improvement and encourage countries to act raising awareness on the state of cybersecurity worldwide. This GCI consists of 82 questions to evaluate 5 pillars: legal measures, technical measures, organizational measures, capacity development measures and cooperation measures.

[1] Autor corresponsal: jeancasoto@unitec.edu, Universidad Tecnológica Centroamericana, Campus Tegucigalpa, Honduras

Disponible en: http://dx.doi.org/10.5377/innovare.v12i1.15956

In the last GCI published in 2020, Honduras performed in the last place of the GCI in the America Region and was placed in the 178th position out of 182 countries with a score of 2.2 out of 100 (International Communication Union, 2020). This means almost every person using the internet does not actually know the risks of the ever-increasing malicious software out there.

Malware is defined as malicious software that is intentionally placed or inserted into a system to harm (Stallings, 2006) and it has been known for some time as one of the strongest threats on the internet. State-of-the-art software for virus detection and prevention (antivirus) has been quite successful.

However, antivirus providers face problems due to the large number of variations of malware that are produced daily. Ciampa (2021) highlights in the 2018 McAfee Labs threats report that the number of new malware released every month exceeds 20 million, and the total malware in existence is approaching 900 million instances. In 2019, four out of every five organizations experienced at least one successful cyberattack, and over one-third suffered six or more successful attacks (Cyberedge Group, 2020).

The organizations that oversee dealing with this type of threat increasingly require better techniques for the automated classification of malware samples in general. Malware classification is a process that was traditionally done manually (Tian et al., 2009; Gheorghescu, 2005). This became inefficient over time because of the large number of malware samples emerging daily product of the polymorphism, metamorphism, and obfuscation techniques involved in modern malware. As a consequence of the inefficiency of this process, the need of automating and standardizing this process arises. Malicious network traffic samples should be identified with the least possible margin of error by this automated process.

One of the most popular approaches for malware classification is based on content. This checks the content of the files and compares them with signatures from a database, looking for matches with previously identified malware samples. Some research works (Tian et al., 2009) concentrate on Malicious Executable Classification Systems (MECS) that distinguish between benign or malignant executables. However, this approach cannot recognize new variants of already known families without having an existing sample of these.

Another approach for malware classification is based on behavior, which is subdivided into two types: based on Central Processing Unit (CPU) and based on the data network traffic. The first one analyzes and monitors the behavior of programs on the computer. The second one analyzes and monitors incoming and outgoing data packets, connections to hosts, and others. Even though monitoring and processing system calls can be a resource intensive task (Nari & Ghorbani, 2013), most of the works using CPU-based classification are based on system calls, used to abstract, and represent malware behavior. Nari & Ghorbani (2013) proposed the behavior-based approach via data traffic network under the assumption that when a new variant of malware emerges, it will show similar behavior to its predecessor regardless of the obfuscation, polymorphism, or metamorphism used to create it. Today we can find numerous investigations (Hock & Kortis, 2015; Chockwanich & Visoottiviseth, 2019; Jabez & Muthukumar Dr., 2015; Yin et al., 2017) that show the behavior of malware in the data traffic network as an essential component.

Identifying malware by the network traffic is quite the same as the intrusion detection systems based on the network. An intrusion detection/prevention system (IDS/IPS) is a security tool that can detect malicious activity and taking preventive measures to protect both the host and the network against potential threats, which would normally pass through a traditional firewall (Ambati & Vidyarthi, 2013; Kolokotronis & Shiales, 2021). IDS/IPS are divided into two categories: Host Intrusion Detection/Prevention System (HIDS/HIPS) and Network Intrusion Detection/Prevention System (NIDS/NIPS). HIDS/HIPS are user (host) based IDS/IPS.

These are used to analyze and monitor activities in a particular machine. NIDS/NIPS detect and prevent intrusion threats by continuously monitoring data network traffic, looking for malicious and unauthorized entries that attempt to harm the basic security of the data network. These systems take automatic action to stop the intrusion by sending alerts to the administrator, dropping, or blocking malicious traffic from the source address, or terminating the connection (Kolokotronis & Shiales, 2021).

Shipulin (2018) explains the technology behind the NIDS/NIPS systems. These works at layer 4 of the OSI (Open Systems Interconnection) model (Purser, 2004). That is, with transport layer protocols such as TCP (Transmission Control Protocol), UDP (User Datagram Protocol), and others. The goal is to identify malicious packets in data network traffic representing attack attempts. The incoming traffic is divided into its corresponding protocol, and it is decoded, decompressed, normalized, and later compared it with a set of signatures.

This research work is based on the premise that any new variant of malware behaves similarly to that of its predecessor (Tian et al., 2009), together with the fact that most malware communicates with external hosts (Nari & Ghorbani, 2013). The proposed model bases its operation on the behavior-based approach at the data network level (Nari & Ghorbani, 2013). This model parses files containing frames and packets captured from the network, known as (Packet Capture) PCAP files.

This model employs two methods for classifying malware-generated traffic samples: (a) using traditional machine learning algorithms such as K-Nearest Neighbors (K-NN) and Support Vector Machines (SVM) and (b) using various deep neural network models. For both methods, the CICIDS2017 dataset (available in https://www.unb.ca/cic/datasets/ids-2017.html) was used for training and evaluation. This dataset contains network traffic features generated by the CICFlowMeter (network traffic flow generator and analyzer available in https://github.com/CanadianInstituteForCybersecurity/CICFlowMeter). Likewise, as part of the objectives, two scenarios were contemplated for both methods: (a) non-standardizing the dataset, and (b) standardizing the dataset. The results of combining different classification methods with and without standardization were then compared.

2. Methods

This research work aimed to develop a model that can identify network traffic generated by malware. In other words, it is to automate the process of classifying data frames and to locate those whose behavior patterns indicate possible intrusions to the network. The process used two methods: (a) traditional machine learning algorithms (K-NN and SVM) and (b) various deep neural networks models. Both methods were trained and tested using the CICIDS2017 dataset (Figure 1).

For our experiments, the positive class is malign network traffic. Thus, our goal was to get the lowest possible number of false negatives, indicated by a high recall. A false negative means allowing attacks to pass through the data network, which in this context could trigger a series of catastrophic events that would compromise the confidentiality, integrity, and availability of our devices. Our results scored up to 99% for the recall metric, indicating a very low number of false negatives. In addition, Receiver Operating Characteristic (ROC) curve was applied to compare neural network models.

Finally, the results have shown that the standardization of the data can have a positive impact on the results if used in the right context. This research sought to answer the following research questions: Can we achieve good results (greater than 95% of recall) in the classification of samples of traffic generated by malware, by analyzing only its behavior of network data traffic? Will classification of malware-generated traffic samples using neural networks perform better on average than classification using K-NN and SVM? Will data standardization impact drastically the evaluation metrics?

2.1. Dataset

The CICIDS2017 dataset (Sharafaldini et al., 2018) was acquired from the official website of the University of New Brunswick (UNB). This set contains 8 files of type CSV with a total of 2,830,743 records, each record has a total of 79 features. The dataset provides samples of connections from 15 types, including benign network traffic. However, the goal of this work was to identify any network traffic generated by malware. Therefore, all malware types were grouped together to represent a single class called malign. Of all connections in the dataset, 80.32% were benign and 19.68% were malign (8.1334% DoS HULK, 5.6171% PortScan, 4.5249% DDoS, 0.3638% DoS Golden Eye, 0.2806% FTP-Patator, 0.2084% SSH-Patator, 0.2049% DoS Slowloris, 0.1944% DoS Slowhttptest, 0.0695% Bot, 0.0533% Web Attack Brute Force, 0.0230% Web Attack - XSS, 0.0013% Infiltration, 0.0007% Web Attack SQL Injection, 0.0004% Heartbleed).

This dataset was cleaned to deal with missing values. For those cases of missing values, the entire record was deleted. As a result, 1,358 records were deleted during this cleanup process, that is less than a 0.05% of the dataset. There are two common approaches in scaling the data: normalization and standardization. Raschka et al. (2022) argue that standardization may be more practical for machine learning algorithms due to its mean of 0 and standard deviation of 1 that facilitates learning weights. The need for data standardization is typically defined by the nature of the data itself. Here, to study the impact of the standardization approach on this dataset, we compared results with and without standardizing the data.

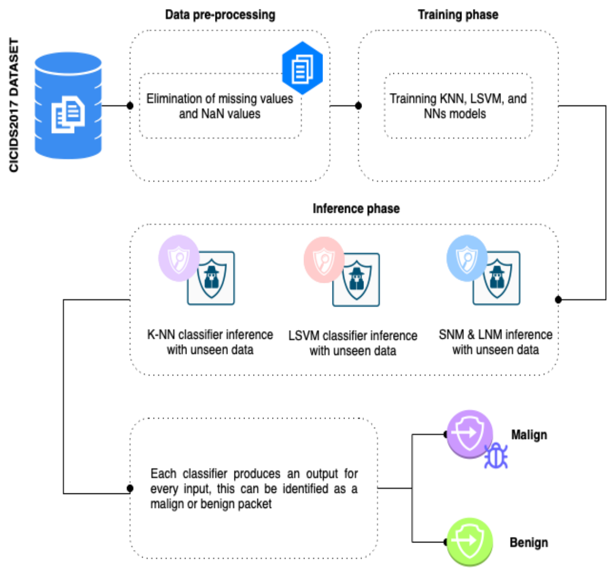
Data pre-processing
Training phase
CICIDS2017 DATASET
Elimination of missing values and NaN values
Trainning KNN, LSVM, and NNs models
Inference phase
K-NN classifier inference with unseen data
LSVM classifier inference with unseen data
SNM & LNM inference with unseen data
Each classifier produces an output for every input, this can be identified as a malign or benign packet
Malign
Benign

Figure 1. Methodology diagram.

2.2. Traditional machine learning methods

We chose traditional machine learning algorithms such as K-NN defined as a method were examples classified by their nearest neighbor (Cunningham & Delany, 2021) and SVM defined as algorithms to solve classification and regression problems (Cortes & Vapnik, 1995). These algorithms provide a baseline for other experiments.

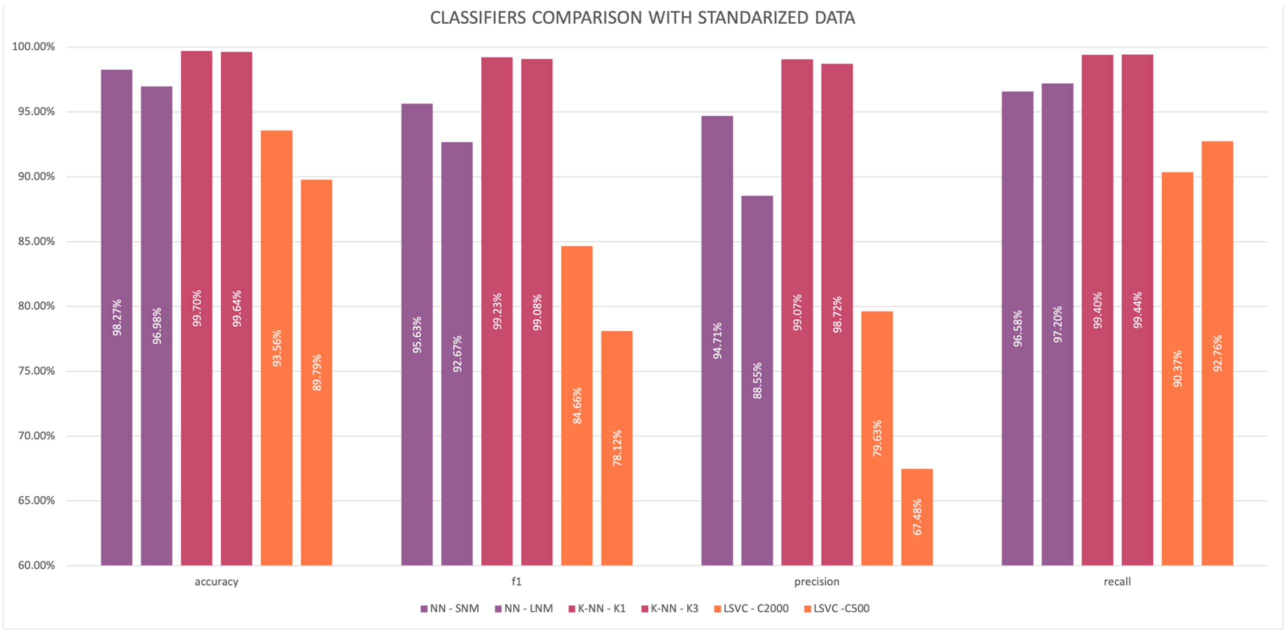

Figure 2. Best classifiers comparison using standardized data.

We used the implementation provided by the SciKit-Learn machine learning package for Python (Pedregosa et al., 2011) to solve a binary classification problem (either benign or malignant). For classification, the K-NN algorithm classifies instances based on the classes of their nearest neighbors in the training set. SVM, on the other hand, can solve classification and regression problems by projecting instances into higher dimensional spaces using kernels. In this work we used linear SVC (LSVC).

2.3. Deep learning methods

The second method used in this research aimed to experiment with various models of neural networks (NNs). That is, with two sets of hyperparameters for the designed neural network architectures. Two main architectures were developed: SNM and LNM. SNM and LNM stand for small neural model and large neural model, the small one has 2 hidden layers (78 inputs → 48 neurons → 24 neurons → 2 outputs), and the large 5 hidden layers (78 inputs → 64 neurons → 32 neurons → 16 neurons → 8 neurons → 4 neurons → 2 outputs). Both models used the cross-entropy loss function BCEWithLogitsLoss and SGD as the optimization algorithm. There is no limit for the number of epochs, but the training will stop if the validation loss does not decrease after 10 epochs have passed. A batch size of 4096 and learning rate of 0.001 were used, and for the fully connected layers, both models used sigmoid as the activation function.

3. Results

The results of the experiments with both traditional machine learning and deep networks are presented in this section. In both cases, we also varied the usage of standardized and non-standardized data.

3.1. Evaluation metrics

We aimed to classify when a connection was benign or malicious. In the context of binary classification of packet samples, when a packet is determined to be malicious, we say that it is a positive, otherwise it is a negative. However, the prediction can be right or wrong. Correct predictions are either true positives (TP) or true negatives (TN). Incorrect predictions are either false positives (FP) or false negatives (FN). In the context of cyber-attacks, a false positive does not represent a significant threat to the user, company, or organization. While a false negative would let through attacks on the data network, which could trigger a series of catastrophic events.

Accuracy is defined as the ratio between the correct predictions (TP + TN) and all predictions. Precision deals with the percentage of items identified as belonging to a given class that are of that class (focuses on false positives). Recall is about the percentage of items in a class that were successfully identified (focuses on false negatives). F1-Score is the harmonic mean of precision and recall and tries to maximize both quantities.

For the neural network experiments, the use of the Receiver Operating Characteristic (ROC) curve was implemented. This is defined as an evaluation metric for binary classification problems. It is a probability curve that plots the TP rate against the FP rate at various threshold values. The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. High AUC values mean that the model is good at distinguishing between positive and negative classes.

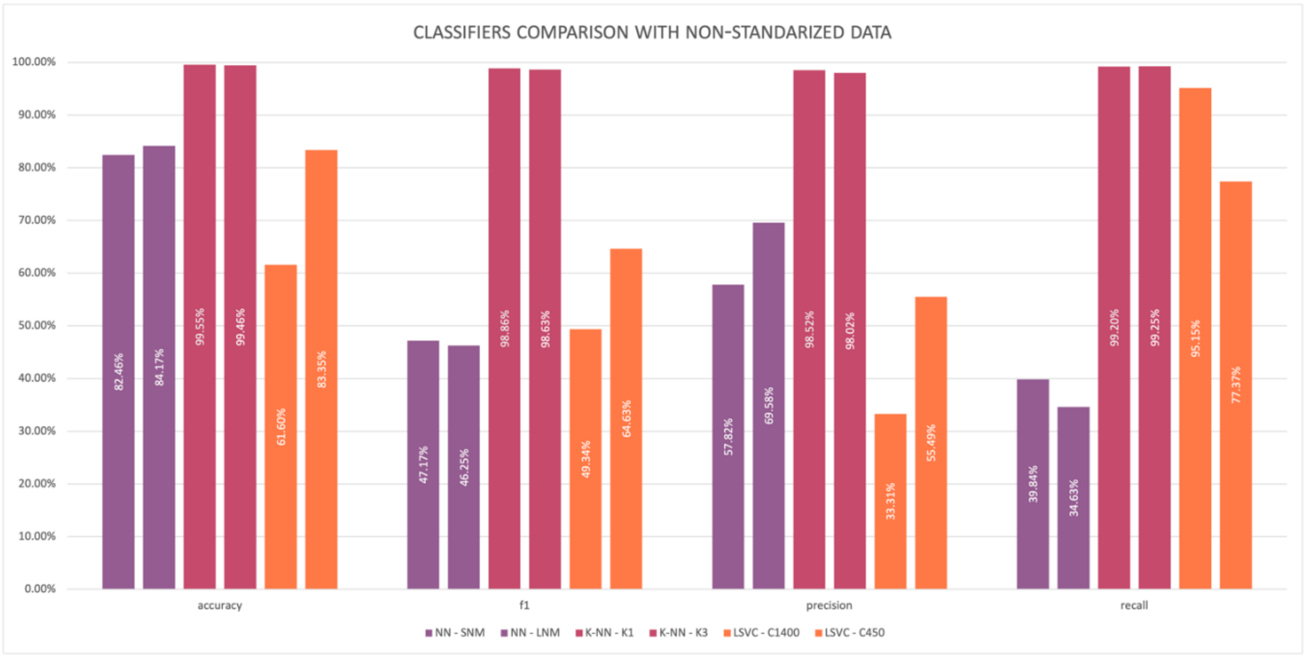

Figure 3. Best classifiers comparison using non-standardized data.

3.1.1. K-Nearest Neighbor

In order to find the best performance for a specific K in a limited interval, eight different values for K were used: K = {1, 3, 5, 7, 9, 11, 13, 15}. K-NN classifier performs slightly better when trained with standardized data. This classifier with non-standardized and standardized data performed above 95%, but still got a slightly better performance when using standardized data (Figures 2 and 3).

3.1.2. Support vector machines

The linear kernel SVC experiment was performed with 100% of the data. It consisted of 23 different values for C to find the best performance for a specific C. The values of C used were the following: C = {0.001, 0.01, 0.10, 1.00, 10.00, 50.00, 100.00, 150.00, 200.00, 250.00, 300.00, 350.00, 400.00, 450.00, 500.00, 600.00, 700.00, 800.00, 1000.00, 1200.00, 1400.00, 1600.00, 2000.00}. This classifier also performed better when trained with standardized data. LSVM classifiers performed less than 70% in the F1-score metric with non- standardized data but performed above 75% with standardized data with the best two C values for this classifier (Figures 2 and 3).

3.1.3. Neural networks

The NNs models showed a similar behavior like the traditional machine learning classifiers, based on the F1-score they got better performance when trained with standardized data. Both models performed above 90% in the F1-score, but SNM performed slightly better, reaching above 95% in this metric (Figures 2 and 3).

4. Discussion

The best performance was chosen based on the F1-score due to the classifier's context. Although other works based their choice on the accuracy, we've prioritized a lower number of FN indicated by a higher recall. Nevertheless, we sought to achieve high precision. Therefore, F1-score gave us the best of both.

Tian et al. (2009) method has outperformed other important works (Bailey et al., 2007; Shafiq et al., 2009) with a 97% accuracy. Our method outperformed those works mentioned before with accuracy above 99%. However, in this study, we focused on the recall and F1-score in which K-NN classifier obtained the best performance for non-standardized and standardized data.

The NN models used in this research did not perform better than traditional machine learning algorithms. This does not mean that traditional learning algorithms always perform better than deep learning models. However, no classifier was superior in all metrics.

The best result for the recall metric with unseen data was obtained by evaluating the classifier with standardized data (by tenths). Although the difference is minimal, it is an improvement on the metric. Likewise, based on the F1 metric, the model showed a better performance in the inference with standardized data in terms of a lower percentage of overfitting compared to the inference with non-standardized data. Although the difference is minimal, it is an improvement in the metric.

The best result for the recall metric with unseen data was obtained by evaluating the classifier with non-standardized data. However, there was a more consistent behavior with standardized data. Although the result obtained (92.76%) for standardized data is below the desired value (greater than 95%) in the recall metric, it had a better precision percentage (67.48%) compared to the precision percentage obtained for the non-standardized data (33.30%).

Furthermore, based on the F1 metric, which is in charge of maximizing precision and recall, the results showed that the classifier evaluated with unseen data obtained the best performance for standardized data. Consequently, the use of data standardization is recommended for this classifier.

The neural network model SNM obtained better performance when using standardized data. Likewise, the data obtained by the inference of the SNM model with unseen data showed that it reached a maximum of 39.84% in the recall metric with non-standardized data, compared to 96.58% with standardized data. Consequently, for the SNM model with the same hyperparameters, data standardization is strongly recommended.

Meanwhile, the LNM neural network model obtained better performance when standardized data was used. Likewise, the data obtained by the inference of the LNM model with unseen data showed that it reached a maximum of 34.63% in the recall metric with non-standardized data, compared to 97.20% with standardized data. Same as the previous case, data standardization is strongly recommended for the LNM model with the same hyperparameters.

Our best NN method outperformed Rhode et al. (2018) work when talking about accuracy abruptly. They achieved 94% accuracy with Recurrent Neural Network (RNN) in just 5 seconds, and 96% of accuracy in less than 10 seconds.

Nevertheless, it’s important to consider that our study did not include execution time to determine the best method, owing to the lack of hardware with sufficient computational power to execute the experiments. It’s not just about evaluation metrics, but execution time. As time goes by, we need better processes that require less execution time.

The large dataset we’ve used to make these experiments (over 2 million samples) compared to the less than a thousand samples used in other works (Tian et al., 2009; Bailey et al., 2007; Shafiq et al., 2009) gives us confidence about the results achieved.

5. Conclusion

A process capable of classifying connections as either benign or malicious (malware-generated) was successfully developed. This was carried out using both traditional machine learning algorithms and deep learning. Both were able to achieve above 95% in the F1-score and recall metric. This effectiveness was attained using only traffic information from the data network.

The NN models used did not perform better than traditional machine learning algorithms. However, no classifier was superior in all metrics. The choice of the best performance was made based on the F1-score due to the classifier’s context. The best performance was obtained by the K-NN classifier for non-standardized and standardized data. The impact of data standardization was a remarkable finding in the different experiments. Although it did not have a positive impact on all metrics, it improved the most relevant metrics to this research (F1 and recall). In general, we can say that the standardization had a notable impact on the evaluation metrics. In consequence, the use of data standardization is strongly recommended.

6. Acknowledgements

To Dr. Kenny Davila Ph.D for supporting and mentoring me through the development of this research work which was originally developed as a graduation project.

7. Conflicts of Interest

The author declares no conflict of interest.

8. Bibliographic References

Ambati, S. B., & Vidyarthi, D. (2013). A brief study and comparison of open-source intrusion detection system tools. *International Journal of Advanced Computational Engineering and Networking, 1*(10), 26-32.

Bailey, M., Oberheide, J., Andersen, J., Mao, Z. M., Jahanian, F., & Nazario, J. (2007). Automated classification and analysis of internet malware. In C. Kruegel, R. Lippmann & A. Clark (Eds.), *Recent Advances in Intrusion Detection (RAID),* (pp. 178-197). Springer.

Chockwanich, N., & Visoottiviseth, V. (2019). Intrusion detection by deep learning with tensorflow. *2019 21st International Conference on Advanced Communication Technology (ICACT)*, 654-659. IEEE Xplore. https://ieeexplore.ieee.org/document/8701969

Ciampa, M. (2021). *CompTIA security+ guide to network security fundamentals* (6th ed.). Cengage Learning.

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning, 20,* 273-297. https://dx.doi.org/10.1007/BF00994018

Cunningham, P., & Delany, S. J. (2021). K-nearest neighbour classifiers-a tutorial. *ACM Computing Surveys, 54*(6), 1-25. https://dx.doi.org/10.1145/3459665

Cyberedge Group. (2020). *2020 cyberthreat defense report.* https://cyber-edge.com/cdr/

Gheorghescu, M. (2005). An automated virus classification system. *2005 Virus Bulletin Conference*, 294-300.

Hock, K., & Kortis, P. (2015). Commercial and open-source based intrusion detection system and intrusion prevention system (IDS/IPS) design for an IP networks. *2015 13th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, 1-4. IEEE Xplore. https://ieeexplore.ieee.org/document/7558466

International Communication Union. (2020). *Global Cybersecurity Index. Measuring commitment to cybersecurity.* https://www.itu.int/dms_pub/itu-d/opb/str/D-STR-GCI.01-2021-PDF-E.pdf

Jabez, J., & Muthukumar Dr., B. (2015). Intrusion detection system (IDS): anomaly detection using outlier detection approach. *Procedia Computer Science, 48,* 338-346. https://dx.doi.org/10.1016/j.procs.2015.04.191

Kolokotronis, N., & Shiaeles, S. (2021). *Cyber-security threats, actors, and dynamic mitigation* (1st ed.). CRC Press. https://dx.doi.org/10.1201/9781003006145

Nari, S., & Ghorbani, A. (2013). Automated malware classification based on network behavior. *2013 International Conference on Computing, Networking and Communications (ICNC),* 642-647. https://dx.doi.org/10.1109/ICCNC.2013.6504162

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825-2830.

Purser, S. (2004). *A practical guide to managing information security.* Artech House.

Raschka, S., Liu, Y. H., Mirjalili, V., & Dzhulgakov, D. (2022). *Machine Learning with PyTorch and Scikit-Learn: develop machine learning and deep learning models with Python* (1st. ed.). Packt Publishing.

Rhode, M., Burnap, P., & Jones, K. (2018). Early-stage malware prediction using recurrent neural networks. *Computers & Security, 77,* 578-594. https://dx.doi.org/10.1016/j.cose.2018.05.010

Shafiq, M. Z., Tabish, S. M., Mirza, F., & Farooq, M. (2009). PE-Miner: Mining structural information to detect malicious executables in realtime. In E. Kirda, S. Jha & D. Balzarott (Eds.), *Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection (RAID).* Springer-Verlag. https://dl.acm.org/doi/10.1007/978-3-642-04342-0_7

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. *Proceedings of the 4th International Conference on Information Systems Security and Privacy - ICISS*P, 108-116. https://dx.doi.org/10.5220/0006639801080116

Shipulin, K. (2018). We need to talk about IDS signatures. *Network Security, 2018*(3), 8-13. https://dx.doi.org/10.1016/S1353-4858(18)30024-2

Stallings, W. (2006). *Cryptography and network security principles and practices* (4th ed.). Pearson Education.

Tian, R., Batten, L., Islam, R., & Versteeg, S. (2009). An automated classification system based on the strings of trojan and virus families. *4th International Conference on Malicious and Unwanted Software Conference 2009*. https://ieeexplore.ieee.org/document/5403021

Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. *IEEE Access*, *5*, 954-961. https://dx.doi.org/10.1109/ACCESS.2017.2762418