Vol. 07, No. 3 (2025) 1493-1500, doi: 10.24874/PES07.03.011

Proceedings on Engineering Sciences

www.pesjournal.net

ANOMALY DETECTION IN COMPUTER NETWORKS USING AI TECHNIQUES: A BENCHMARK STUDY

Stefan Ćirković[1]
Marjan Milošević

Received 15.07.2024.
Revised 24.12.2024.
Accepted 24.01.2025.

Keywords:

Anomaly detection, Machine learning, Neural networks, Network security, UNSW-NB15 dataset

A B S T R A C T

Analysis of network log files is crucial for identifying anomalies and addressing issues in information systems. The process of analyzing log files with traditional approaches can be highly demanding and tedious, especially when logs are large and unstructured. In research, machine learning algorithms and neural networks are applied to analyze network log files, aiming to detect specific system attacks. The UNSW-NB15 dataset, widely used in network security research, is employed to assess the prediction accuracy of various algorithms. Results indicate that algorithms like XGBoost, Random Forest, and Extra Trees achieve high accuracy rates exceeding 97%, while neural networks achieve approximately 96% accuracy. Training and prediction times for different algorithms are also analyzed, noting that neural networks, particularly LSTM, typically require longer training times. This study provides deeper insights into the performance of various anomaly detection algorithms in network log files and offers guidance for further research in cybersecurity.

1. INTRODUCTION

In recent years, there has been a significant increase in automated cyber-attacks, leading to the development of effective information protection strategies in organizations worldwide. Traditional approaches to log file analysis have failed to keep pace with this dynamic environment, nor have they been sufficiently effective in detecting increasingly sophisticated attacks and rapid changes in their patterns. Additionally, dealing with large and unstructured datasets further complicates this challenge.

Every security device, such as application firewalls, anti-spam devices, intrusion prevention systems (IDS), and authentication servers, generates vast amounts of real-time logs. These logs represent valuable sources of information about network activities. However, manual analysis of these logs becomes nearly impossible due to their sheer volume, leading to the potential oversight of many potentially dangerous activities.

This study aims to explore how various machine learning algorithms and neural networks can contribute to anomaly detection in network log files. Using the well-known UNSW-NB15 dataset, widely used in network security research, the performance of different algorithms in predicting security attacks will be evaluated. Training and prediction times of various algorithms will be presented to understand their practical applicability in real-world scenarios. The results of this research are expected to have a significant impact on

Corresponding author: Stefan Ćirković
Email: stefan.cirkovic@ftn.kg.ac.rs

improving information security and network infrastructure, providing guidelines for implementing the most effective solutions.

2. LITERATURE REVIEW

This chapter provides an overview of relevant studies, research, and theories related to machine learning algorithms and anomaly detection in network logs.

In the study by Shushlevska (2022), various machine learning algorithms (Naive Bayes, Logistic Regression, Decision Tree, and Random Forest) were implemented and compared on the UNSW-NB15 dataset to evaluate their effectiveness in detecting network traffic anomalies. Khan et al. (2022) offer a review of ensemble methods for addressing imbalanced classes in classification, exploring the effectiveness of different combinations and their contributions to improving performance on imbalanced datasets. Hanif (2020) investigates the efficiency of the Extreme Gradient Boosting (XGBoost) algorithm compared to logistic regression. Srivastava et al. (2023) provide an overview of different classification models, including traditional models like logistic regression, decision trees, random forests, and deep neural networks. Abellan et al. (2017) examine a modification of the basic decision tree classifier within the Random Forest algorithm, using a new criterion that leads to the creation of a new decision model. Ampomah et al. (2020) compare the efficiency of ensemble learning models based on decision trees. The NVIDIA website (2024) describes the XGBoost algorithm for machine learning, renowned for its efficiency and wide application in various prediction and classification tasks. It explains how XGBoost operates, its key features, advantages over other machine learning algorithms, and provides practical examples and use cases, enhancing understanding of its application and effectiveness. Moustafa and Slay (2015) addresses the lack of comprehensive network-based datasets for evaluating intrusion detection systems, proposing the creation of the UNSW-NB15 dataset containing real and synthesized network attack scenarios. Sak et al. (2014) explore the application of Long Short-Term Memory (LSTM) in distributed training, focusing on recurrent neural networks. Le et al. (2019) utilize LSTM neural networks for accurate flood prediction in Vietnam. Gao and Glowacka (2016) investigate recurrent neural network structures for learning long-term dependencies, introducing Deep Simple Gated Unit (DSGU) and Simple Gated Unit (SGU). Nayeem et al. (2015) employ artificial neural networks for predicting different classes, emphasizing the feed-forward backpropagation algorithm.
Balboa et al. (2024) compare the performance of logistic regression and machine learning models in predicting evacuation decisions.

This literature review provides a comprehensive overview of current research and applications of various machine learning techniques in anomaly detection.

3. THEORETICAL FOUNDATIONS

3.1. Machine Learning Algorithms

3.1.1. Gradient boosting (GBM)

Gradient Boosting is a machine learning technique used for both classification and regression. In regression, Gradient Boosting uses mean squared error, while in classification, it employs log-loss, also known as the gradient boosting method (GBM). The idea is to minimize the loss by adding successive models that are trained to correct the errors of the previous models. This process allows for the gradual improvement of model performance to achieve better predictions (Khan et al., 2022).

3.1.2. Extreme Gradient Boosting (XGBoost)

XGBoost is an implementation of Gradient Boosting Decision Trees (GBDT), known as one of the most efficient algorithms for supervised learning. It can be used to solve both regression and classification problems (Hanif, 2020). It offers parallel execution of the algorithm, which allows for faster data processing and more efficient model training (NVIDIA, 2024).

XGBoost can be mathematically represented as an optimization problem, where the objective function is formulated by the following equation (1):

$$\text{obj} = \sum_{i=1}^{n} L(yi, \hat{yi}) + \sum_{k=1}^{K} \omega(\text{fk}) \qquad (1)$$

$$\text{where is } \omega(\text{fk}) = \gamma T + \frac{1}{2}\lambda\|w\|^2 \qquad (2)$$

where L is the differentiable convex loss function that measures the difference between the prediction $\hat{yi}$ and the target value yi, while ω constrains the model complexity (Hanif, 2020).

In the gradient boosting algorithm, boosting is viewed as an optimization problem where the goal is to minimize the loss function of the classification model by adding one weak model at each step. The algorithm continuously reduces the errors of previous models in the direction of the gradient to create a new model (Ampomah et al., 2020).

The XGBoost algorithm has the following characteristics (Ampomah et al., 2020): (a) a regularized model to prevent overfitting, (b) an algorithm that is aware of sparsity and can handle different types of sparsity patterns in the data, (c) a distributed weighted quantile sketch algorithm for efficiently handling weighted data, (d) a column block structure that enables parallel learning, (e) a cache-aware prefetching algorithm for fast

retrieval and storage of gradient statistics, (f) out-of-core computation blocks that allow working with large datasets that cannot fit into memory.

3.1.3. Rendom Forest (RF)

Random Forest (RF) is a type of ensemble technique defined as a combination of multiple models. Ensemble techniques encompass two basic approaches: bagging and boosting. Bagging, also known as bootstrap aggregation, involves using multiple prediction models of the same type to reduce variance rather than bias. This approach is often used to overcome overfitting issues in prediction models, with all predictions carrying equal weight. In bagging techniques, training is performed by randomly sampling data from the dataset with replacement for all models (Srivastava et al., 2023).

The Random Forest classifier uses the bagging technique where the basic model is a decision tree. RF consists of multiple trees, where each tree predicts its classification, and the final decision is made based on the majority vote of the trees. The fundamental idea behind RF is simple yet powerful - the wisdom of the crowd. Combining different, independent trees allows for better results compared to individual models. Lower correlation between trees is crucial for RF's success, as it allows the trees to complement each other and compensate for potential errors of individual trees (Srivastava et al., 2023).

The RF classifier has several assumptions: first, there should be minimal correlation between predictions of individual trees; second, the input data features should have real values that the classifier uses for prediction, not estimated values. RF is a highly useful algorithm due to its training speed and high classification accuracy. It works efficiently with large datasets and can be applied in various fields such as banking, medicine, land management, and marketing (Srivastava et al., 2023).

3.1.4. Extra-Trees (ET)

The Extra-Trees classifier creates a group of decision trees that are not pruned. Pruning is the process of removing unnecessary branches from a decision tree to reduce model complexity and prevent overfitting. In this case, the trees retain all branches without adjustment. The Extra-Trees classifier uses the traditional top-down approach but randomly selects attributes and split points when dividing a tree node (Ampomah et al., 2020).

In extreme cases, it creates completely random trees that are not related to the output values of the training samples. This approach differs from other tree-based ensemble methods in two key ways (Ampomah et al., 2020):

- The algorithm splits nodes by choosing split points completely randomly.
- It uses the entire training sample instead of bootstrap replicas for growing the trees.

The final prediction is obtained by combining all tree predictions through majority voting. This means that each tree in the ensemble makes its prediction, and the final decision is based on the majority of votes. For example, if the majority of trees predict a specific class, that class is taken as the final prediction (Ampomah et al., 2020).

3.1.5. Decision Tree (DT)

Decision trees are structures used as classifiers. In situations where elements have one or more characteristics and one class variable under study, classification trees are used to predict the class value of an element based on its characteristics. In trees, each node represents a characteristic, and the branches between nodes and their descendants indicate the values of that characteristic. Tree leaves typically represent the exact class to which the element belongs (Abellan et al., 2017).

3.1.6. Logistic Regression (LR)

Logistic Regression (LR) is a statistical technique used to predict binary outcomes based on independent variables. LR is applied to assess the probability of a specific event occurring or not, where there are only two possible outcomes: either evacuation will occur (denoted as Y = 1) or it will not (denoted as Y = 0). A key element of logistic regression is the logit link represented by formula (3), which combines the probability of the event p, shown in formula (4), with a linear combination of independent variables. This allows for estimating the probability of the event happening, taking into account different characteristics of the independent variables (Balboa et al. 2024).

$$logit(p) = \ln\left(\frac{p}{p-1}\right) = b0 + b1 \times X1 + b2 \times X2 + \cdots . + bn \times Xn \quad (3)$$

$$p = \frac{e^{b0+b1*X1+b2*X2+\cdots .+bn*Xn}}{1 + e^{b0+b1*X1+b2*X2+\cdots .+bn*Xn}} \quad (4)$$

The fundamental concept in logistic regression is the odds ratio, which represents the ratio of the probability of an event occurring (p) to the probability of it not occurring (1-p). Model coefficients (b0, bi) are estimated to calculate this ratio for each independent variable. To facilitate interpretation, the odds can be transformed into probabilities using the exponential function. This process allows predictions of the probability of an event based on given values of the independent variables (Balboa et al. 2024).

3.2. Neural networks

3.2.1. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to model temporal sequences more accurately, capturing long-term dependencies compared to conventional RNNs (Sak et al., 2014). LSTM can be considered an evolution of RNNs, introduced by Hochreiter and Schmidhuber. To address the limitations of traditional RNNs, LSTM adds additional interactions within each module (or cell). It is a specialized type of RNN capable of learning long-term dependencies and retaining information over extended periods by default. Organized in a chain-like structure, each repeating module has a different internal structure. Instead of a single neural network like standard RNNs, LSTM has four interconnected layers with a unique communication method (Le et al., 2019).

3.2.2. Gated Recurrent Unit (GRU)

GRU was first designed by Kyunghyun Cho in his work on neural networks (Cho et. al, 2014). This RNN structure contains only two gates. The update gate controls the information entering the memory, while the reset gate controls the information exiting the memory. Similar to the LSTM unit, GRU has gates that modulate the flow of information within the unit, but without separate memory units. GRU offers advantages over LSTM, such as a simpler architecture with fewer parameters and faster convergence during training. The GRU algorithm has become a popular choice in many deep learning applications due to its efficiency and ease of implementation (Gao & Glowacka, 2016).

3.2.3. Multilayer Perceptron (MLP)

The Multilayer Perceptron (MLP) is an artificial neural network model that uses a feedforward approach to map sets of input data to corresponding output values. MLP consists of multiple layers of nodes organized in a directed graph, with all layers interconnected. In addition to input nodes, each node represents a neuron with a non-linear activation function. For training the network, MLP employs supervised learning based on a technique known as backpropagation. This technique enables MLP to distinguish and process data that are not linearly separable (Nayeem et al., 2015).

3.3. Performance metrics

The accuracy score is used to evaluate the performance of a classification model. It represents the ratio of correctly predicted instances to the total number of instances in the dataset. It is calculated using the following formula (5).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (5)$$

The accuracy percentage varies from 0% to 100%, where a higher percentage indicates better performance. However, accuracy may have limitations, especially in situations with imbalanced classes. In such cases, the model can achieve high accuracy by predicting the majority class.

Precision, denoted as the ratio of true positive predictions to the total positive predictions, is particularly useful in scenarios where the false positive rate is high. This measure is expressed by the formula (6):

$$Precision = \frac{TP}{TP + FP} \quad (6)$$

Recall, on the other hand, measures the model's ability to capture all positive instances and becomes particularly relevant when the false negative rate is significantly high. This measure is expressed by the formula (7):

$$Recall = \frac{TP}{TP + FN} \quad (7)$$

F1 score, as the harmonic mean of precision and recall, provides a balanced assessment of model performance, especially in situations with uneven class distributions. It is expressed by the formula (8):

$$F1 - score = 2\ x\ \left(\frac{Precision\ x\ Recall}{Precision + Recall}\right) \quad (8)$$

4. METHODOLOGY

4.1. Dataset

UNSW-NB15 is a dataset consisting of network packets, created within the Cyber Range Lab at UNSW, Canberra, Australia (Moustafa & Slay, 2015). It is a labelled, imbalanced dataset that includes nine types of attacks: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. Network traffic was captured in packet form using the tcpdump tool. The simulation period was from January 22, 2015, to February 17, 2015, during which 100 GB of raw data was collected, generated in the IXIA PerfectStorm environment.

The raw data is processed using the Argus Bro-IDS tool, generating 49 attributes, including class labels. The data is organized into four CSV files, each containing a portion of the total 2,540,044 records. There is an additional table with accurate classifications called UNSW-NB15_GT.csv, as well as an event list named UNSW-NB15_LIST_EVENTS.csv. This dataset contains detailed information about each packet, including sources, destinations, timestamps, and activity types.

In this paper, the UNSW-NB15 dataset was selected because it provides depth and comprehensiveness in cybersecurity analysis. It offers detailed information

about network traffic, including various types of attacks and their characteristics. Moreover, it contains numerous attributes and a wealth of data generated through various algorithms, providing rich information for cybersecurity research. A detailed display of all features with descriptions is shown in (Table 1).

Table 1. Features found in the UNSW-NB15 dataset

#	Characteristic	Description
1.	srcip	Source IP address.
2.	sport	Source port.
3.	dstip	Destination IP address.
4.	dsport	Destination port.
5.	proto	Protocol (TCP, UDP).
6.	state	Connection state (FIN, CON).
7.	dur	Session duration.
8.	sbytes	Number of bytes sent from the source IP.
9.	dbytes	Number of bytes received on the destination IP
10.	sttl	Source Time-to-Live (TTL).
11.	dttl	Destination Time-to-Live (TTL).
12.	sloss	Number of lost packets from the source IP.
13.	dloss	Number of lost packets to the destination IP.
14.	service	Service type (HTTP, FTP).
15.	sload	Source load (bit/s).
16.	dload	Destination load (bit/s).
17.	spkts	Number of packets sent from the source IP.
18.	dpkts	Number of packets received on the destination IP.
19.	swin	Source TCP window.
20.	dwin	Odredišni TCP prozor
21.	stcpb	Source TCP sequence number.
22.	dtcpb	Destination TCP sequence number.
23.	smeansz	Average size of the source packet.
24.	dmeansz	Average size of the destination packet.
25.	trans_depth	Transaction depth.
26.	res_bdy_len	Length of the response body.
27.	sjit	Source jitter.
28.	djit	Destination jitter.
29.	stime	Session start time.
30.	ltime	Session end time.
31.	sintpkt	Average interval between packets from the source IP.
32.	dintpkt	Average interval between packets to the destination IP.
33.	tcprtt	TCP round-trip time
34.	synack	Time between sending SYN and receiving SYN-ACK.
35.	ackdat	Time between sending ACK and receiving ACK.
36.	is_sm_ips_ports	Indicator of whether the source and destination IP/port match.
37.	ct_state_ttl	Number of states with the same TTL.
38.	ct_flw_http_mthd	Number of HTTP methods during the session.
39.	is_ftp_login	Indicator of FTP login session.
40.	ct_ftp_cmd	Number of FTP commands during the session.
41.	ct_srv_src	Number of connections from the source IP to the same service.
42.	ct_srv_dst	Number of connections from the destination IP to the same service.
43.	ct_dst_ltm	Number of connections from the destination IP in the last X seconds.
44.	ct_src_ ltm	Number of connections from the same source IP address.
45.	ct_src_dport_ltm	Number of connections from the source IP to the same port in the last X seconds.
46.	ct_dst_sport_ltm	Number of connections from the destination IP to the same port in the last X seconds.
47.	ct_dst_src_ltm	Number of connections between the same IPs in the last X seconds.
48.	attack_cat	Attack category.
49.	label	Label (normal or attack).

In (Figure 1), the distribution of different attack types in the UNSW-NB15 dataset is shown. The bottom of the graph displays the attack types included, while the left side shows the number of attacks for each type. The most common attack types are Generic and Exploits, with 40,000 and 33,393 attacks, respectively. The chart also depicts the distribution between normal and malicious traffic. Malicious traffic constitutes 68.06% of the total traffic, while normal traffic accounts for 31.94%.

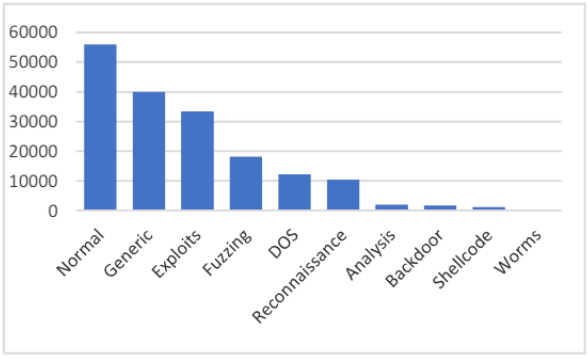

Figure 1. Types of attacks in the UNSW-NB15 dataset

4.2. Data preprocessing

The first step in data preprocessing is eliminating unnecessary features. In this case, features like *id* and *attack_cat* were removed because they do not contribute to data description or prediction of the target variable. In the next step, the SelectKBest algorithm was applied to select the most important features that contribute the most to predicting the target variable. The SelectKBest algorithm uses statistical metrics such as F-test, chi-squared test, or mutual information to assess the significance of each feature relative to the target variable. Numeric features were scaled using MinMax scaling to

normalize their values. Encoding was applied to categorical features using One-Hot Encoding technique.

On the processed dataset, the following machine learning and neural network models were employed with default parameters:

- Extreme Gradient Boosting (XGBoost)
- Random Forest (RF)
- Extra Trees (ET)
- Decision Tree (DT)
- Long Short-Term Memory (LSTM)
- Gated Recurrent Unit (GRU)
- Multilayer Perceptron (MLP)
- Gradient Boosting (GBM)
- Logistic Regression (LR)

Before the models were trained, the dataset was split into various training and test sets using the ShuffleSplit technique. Subsequently, the models were trained on the training sets and evaluated on the test sets using cross-validation. The test results for each performance metric (accuracy, precision, recall, F1 score) were recorded and compared, along with the training time for each model. Python programming language was used for this analysis, along with Jupyter Notebook tool, utilizing Pandas and NumPy libraries for data processing, matplotlib for result visualization, and Seaborn and sklearn (Scikitlearn) for machine learning algorithms.

5. RESULTS AND INTERPRETATION

The analysis of the training results of various machine learning algorithms and neural networks for anomaly detection in network log files, with metrics including Accuracy, Recall, Precision, and F1-Score, is presented in (Table 2). Table 3 shows the training time, prediction time, and total time for each algorithm.

The XGBoost algorithm demonstrated the best results among all tested models with an accuracy, recall, precision, and F1-score of 97.80%. In addition to high detection performance, XGBoost also excelled in training and prediction speed, with a total time of just 0.6 seconds. Both Random Forest and Extra Trees algorithms exhibited high performance with accuracy, recall, precision, and F1-score above 97.5%. However, the training and prediction time for Random Forest was slightly longer (2.9 seconds) compared to Extra Trees (2.5 seconds). The Decision Tree achieved an accuracy of 96.57%; it was slightly less precise than the previous three models but had a low training time (1.1 seconds) and zero prediction time. The LSTM and GRU neural network algorithms showed similar performance with an accuracy around 96.5%. However, their training and prediction times were significantly longer. LSTM had a total time of 134.1 seconds, while GRU had a time of 57.1 seconds, which may limit their real-time practical application. The MLP (Multi-Layer Perceptron) achieved an accuracy of 96.30% with a total training and prediction time of 17.4 seconds. Although not the fastest, it offers good performance with reasonable training time. The Gradient Boosting algorithm had slightly lower performance (95.85%) compared to other ensemble methods. The Logistic Regression baseline model showed the lowest performance with an accuracy of 92.80%, making it the least effective among all tested algorithms.

Table 2. Performance of Machine Learning Algorithms and Neural Networks for Anomaly Detection

Algorithm	Accuracy	Recall	Precision	F1-Score
XGBoost	97.80%	97.80%	97.81%	97.80%
RF	97.68%	97.68%	97.69%	97.68%
ET	97.53%	97.53%	97.55%	97.53%
DT	96.57%	96.57%	96.57%	96.57%
LSTM	96.55%	96.55%	96.55%	96.55%
GRU	96.45%	96.45%	96.45%	96.45%
MLP	96.30%	96.30%	96.33%	96.30%
GBM	95.85%	95.85%	95.86%	95.85%
LR	92.80%	92.80%	92.83%	92.80%

Table 3. Training and Prediction Times for Models

Algorithm	Time to train	Time to predict	Total time
XGBoost	0.6 s	0.0 s	0.6 s
RF	2.8 s	0.1 s	2.9 s
ET	2.4 s	0.1 s	2.5 s
DT	1.1 s	0.0 s	1.1 s
LSTM	42.3 s	91.8 s	134.1 s
GRU	51.0 s	6.2 s	57.1 s
MLP	17.4 s	0.0 s	17.4 s
GBM	34.5 s	0.0 s	34.5 s
LR	1.1 s	0.0 s	1.1 s

6. COMPARISON WITH RELATED STUDIES

In comparison to related studies, the XGBoost algorithm in this study has achieved outstanding performance. In the study by Shushlevska et al. (2022), various algorithms were used for anomaly detection, yielding the following results across four key metrics: Naive Bayes achieved an accuracy of 70.62%, precision of 68.78%, recall of 85.39%, and F1-score of 76.19%. Logistic Regression demonstrated an accuracy of 70.59%, precision of 65.98%, recall of 96.19%, and F1-score of 78.27%. Decision Tree attained an accuracy of 86.12%, precision of 82.20%, recall of 95.46%, and F1-score of 88.33%. Random Forest achieved an accuracy of 87.09%, precision of 81.77%, recall of 98.52%, and F1-score of 89.36%.

In this study, various machine learning algorithms and neural networks were used, with the best result achieved using the XGBoost algorithm. Specifically, the research yielded the following results: applying the Logistic Regression algorithm resulted in an accuracy of 92.80%, precision of 92.80%, recall of 92.83%, and F1-score of 92.80%. Decision Tree showed an accuracy of 96.57%, precision of 96.57%, recall of 96.57%, and F1-score of 96.57%. Random Forest achieved an accuracy of

97.68%, precision of 97.68%, recall of 97.69%, and F1-score of 97.68%. The high accuracies in this study are the result of careful selection of relevant features, first by eliminating insignificant features that do not affect the target variable, and then by applying the powerful SelectKBest algorithm to identify the most important features.

In comparison to the study by (Meftah et al., 2019), which also utilized the UNSW-NB15 dataset with various machine learning algorithms, Support Vector Machine achieved an accuracy of 82.11%, Gradient Boost Machine reached 61.83%, while Logistic Regression achieved 77.21%. The ability of the XGBoost algorithm to significantly outperform Support Vector Machine, Logistic Regression, and Gradient Boost Machine may stem from its effectiveness in modeling complex, non-linear relationships between data and the target variable.

Furthermore, in the study (Kim et al., 2021), the LSTM neural network algorithm achieved an accuracy of 90%, but with significantly longer training and prediction times compared to the results of this study (96.55% and 134.1 seconds). These results highlight the high performance of LSTM networks, while also emphasizing the practical challenges due to the required training time, which is consistent with this research.

These studies emphasize that the approach used in this study, employing advanced data preprocessing techniques and algorithms for selecting the most important features, can significantly enhance anomaly detection in network logs. This study contributes to existing knowledge and opens new possibilities for further research in this field.

7. CONCLUSION

This study investigated the application of various machine learning algorithms and neural networks in anomaly detection, with a particular emphasis on the importance of logs in cybersecurity. The theoretical foundations of the algorithms used are extensively described to provide readers with insights into each of them. The research methodology includes the described dataset, data preprocessing process, and the application of the SelectKBest algorithm for selecting significant features.

The research results are presented methodically, with interpretation of the obtained results and their comparison with similar studies. Special focus is placed on XGBoost, which displayed the best metrics among all models. This study highlighted the high potential of applying these models to enhance cybersecurity, opening paths for further research that integrates these models into real-world security applications.

This research proved the importance of network datasets availability for research in this area. The future work is oriented towards comparison of techniques detection efficiency between different kinds of datasets and also on further improvement of the techniques that showed up to have the best detection rate and application in more complex scenarios, with dominantly encrypted traffic.

Acknowledgement: This study was supported by the Ministry of Science, Technological Development and Innovation of the Republic of Serbia, and these results are parts of Grant No. 451-03-66 / 2024-03 / 200132 with the University of Kragujevac - Faculty of Technical Sciences Čačak

References:

Abellan, J., Mantas, C. J., & Castellano, J. G. (2017). A Random Forest approach using imprecise probabilities. *Knowledge-Based Systems, 134*, 107-123. doi: 10.1016/j.knosys.2017.07.019

Ampomah, E. K., Qin, Z., & Nyame, G. (2020). Evaluation of Tree-Based Ensemble Machine Learning Models in Predicting Stock Price Direction of Movement. *Information, 11*(6), 332. doi: 10.3390/info11060332

Balboa, A., Cuesta, A., González-Villa, J., Ortiz, G., & Alvear, D. (2024). Logistic Regression vs Machine Learning to Predict Evacuation Decisions in Fire Alarm Situations. *SSRN Electronic Journal*. doi: 10.2139/ssrn.4436056

Cho, K., van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724–1734. doi: 10.3115/v1/D14-1179

Gao, Y., & Glowacka, D. (2016). Deep Gate Recurrent Neural Network. *Proceedings of The 8th Asian Conference on Machine Learning, PMLR 63, 350-365.*

Hanif, I. (2020). Implementing Extreme Gradient Boosting (XGBoost) Classifier to Improve Customer Churn Prediction. *Proceedings of the 1st International Conference on Statistics and Analytics (ICSA 2019*), 2-3 August 2019, Bogor, Indonesia. doi: 10.4108/eai.2-8-2019.2290338

Khan, A. A., Chaudhari, O., & Chandra, R. (2024). A review of ensemble learning and data augmentation models for class imbalanced problems: Combination, implementation and evaluation. *Expert Systems with Applications, 223*, 122778. doi: 10.1016/j.eswa.2023.122778

Kim, S., Chen, L., & Kim, J. (2021). Intrusion Prediction using Long Short-Term Memory Deep Learning with UNSW-NB15. *Proceedings of the 2021 IEEE/ACIS 6th International Conference on Big Data, Cloud Computing and Data Science (BCD)*, 13 September 2021. doi: 10.1109/BCD51206.2021.9581420

Le, X.-H., Ho, H. V., Lee, G., & Jung, S. (2019). Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting. *Water*, *11*(7), 1387. doi: 10.3390/w11071387

Meftah, S., Rachidi, T., & Assem, N. (2019). Network Based Intrusion Detection Using the UNSW-NB15 Dataset. *International Journal of Computing and Digital Systems, 8*(5), 372-380. doi: 10.12785/ijcds/080505

Moustafa, N., & Slay, J. (2015). UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). *Proceedings of the 2015 Military Communications and Information Systems Conference (MilCIS)*, November 2015. doi: 10.1109/MilCIS.2015.7348942

Nayeem, M. O. G., Wan, M. N., & Hasan, M. K. (2015). Prediction of disease level using multilayer perceptron of artificial neural network for patient monitoring. *International Journal of Soft Computing and Engineering (IJSCE), 5*(4), 17-23.

NVIDIA. XGBoost. Retrieved from https://www.nvidia.com/en-us/glossary/xgboost/

Sak, H., Senior, A., & Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling. *Proceedings of Interspeech 2014, 14 September 2014*. doi: 10.21437/interspeech.2014-80

Shushlevska, M., Efnusheva, D., Jakimovski, G., & Todorov, Z. (2022). Anomaly Detection with Various Machine Learning Classification Techniques over UNSW-NB15 Dataset. *Proceedings of the 10th International Conference on Applied Innovations in IT (ICAIIT)*, March 2022.

Srivastava, R., Kumar, S., & Kumar, B. (2023). Classification model of machine learning for medical data analysis. *Statistical Modeling in Machine Learning* (pp.111-132). Academic Press. doi: 10.1016/b978-0-323-91776-6.00017-8

Stefan Ćirković
University of Kragujevac,
Faculty of Technical Sciences Čačak
Čačak,
Serbia
stefan.cirkovic@ftn.kg.ac.rs
ORCID 0009-0004-6775-1543

Marjan Milošević
University of Kragujevac,
Faculty of Technical Sciences Čačak
Čačak,
Serbia
marjan.milosevic@uni.kg.ac.rs
ORCID 0000-0003-4730-1292