International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

ENHANCING SOC CAPABILITIES :ACTIVE AND PASSIVE ATTACK DETECTION USING FIREWALL TECHNOLOGIES

R.Saritha[1], Vasam Jahnavi[2], Terala Sudhansh[3], Sabavath Sai Kiran[4], Pashapu Siddartha Reddy[5]

[1] Associate Professor, Dept. of CS, Sri Indu College of Engineering and Technology, Hyderabad,
[2 3 4] Research Student, Dept. of CS Sri Indu College of Engineering and Technology, Hyderabad

Abstract: This paper explores the optimization of firewall parameters for attack detection using machine learning techniques, focusing on improving network security in dynamic environments. Traditional firewall systems often face limitations in detecting malicious traffic due to static rule sets and high false positive rates, particularly in real-world scenarios with evolving attack patterns. To overcome these challenges, this study applies neural networks with the rectified linear unit activation function (ReLU), which enables precise attack detection and real-time firewall policy adjustments. The proposed 5-5-4 neural network model, tested using real-world datasets, achieved an accuracy of 96.3%, outperforming alternative configurations. The analysis evaluated three scenarios: normal conditions, active attacks, and post-policy adjustment, confirming the effectiveness in enhancing detection and mitigation capabilities. The results highlight the potential of machine learning, particularly neural networks, as a robust tool to improve network security. This approach enables future integration of IoT and real-time threat monitoring.

Index *Terms: DDoS attack, firewall, machine learning, neural networks, Orange*

1. INTRODUCTION

THE growing importance of security in computer networks highlights the need for robust monitoring of end devices, network infrastructure, and firewall systems. Firewalls play a vital role in this landscape by controlling network traffic and regulating packet transmission between zones (network areas), with the ability to inspect packets when necessary.

Firewalls are generally classified into two main types: edge firewalls, which manage Internet access, and data centre firewalls, which secure access to server infrastructures.

The manuscript was received on October 1st, 2024.
Dragan JEVTIĆ, Infrastructure of Serian Railway, Nemanjina6, Belgrade, Serbia, dragan.jevtic@srbrail.rs, Information Technology Studies, University of Criminal Investigation and Police Studies, Cara Dušana 196, Belgrade, Serbia
Petar ČISAR, Information Technology Studies, University of Criminal Investigation and Police Studies, Cara Dušana 196, Belgrade, Serbia, petar.cisar@kpu.edu.rs

While edge firewalls focus on controlling external connections, data centre firewalls are specifically designed to protect internal resources within the data centre environment.

From an implementation perspective, firewalls can be categorized as software-based, hardware-based, or cloud-based solutions. Each type offers the same level of protection but comes with specific limitations and differences. Beyond their implementation, it is crucial to consider the types of firewall device in relation to their intended functionality. The primary categories include packet filtering firewalls, stateful inspection firewalls, and proxy firewalls. In addition, specialized solutions such as application firewalls, web application firewalls, and virtual infrastructure firewalls are available.

A firewall's primary function is to filter network traffic based on predefined policies. These policies are shaped by organizational requirements, network architecture, and configuration needs, while also taking into account real-time network conditions. Policies are defined using attributes such as source and destination ports, network address translation (NAT), bytes sent and received, packets sent and received, and specific actions to be taken [1].

This paper aims to demonstrate the optimization of an Internet firewall using machine learning (ML). For this purpose, a dataset collected from an Internet firewall was analysed using the Orange software package [2]. Orange was selected for its extensive graphical components and algorithms, making it a powerful tool for data analysis and modelling.

Machine learning, along with neural networks, is increasingly recognized as providing exceptional results in this field. Neural networks, inspired by biological nerve cells, are designed to meet the computational needs of systems that use this technology. These networks can be classified into cellular, layered, and fully connected structures [3].

Developing neural networks involves several key stages. The process begins with the dataset, followed by the design of the system, which includes tasks such as dataset preprocessing, defining the network topology, setting parameters, and selecting activation functions. This stage encompasses loading and filtering the dataset,

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

specifying the type of network along with its topology, link type, link order, and weight range. It also involves defining the characteristics of individual nodes and determining the system's dynamics, including the initial weight scheme, activation equations, calculations, and the learning algorithm. After the design stage, the system must be trained and finally tested. The test dataset should be left aside. If we use a multilayered perception of a neural network, certain errors that occur during classification can be reintroduced back into the network to modify the network parameters.

Machine learning algorithms can be divided into three categories: supervised learning, unsupervised learning, and reinforcement learning. Supervised learning, the most essential form of machine learning, is characterized by data consisting of pairs of descriptions of what is being learned and what needs to be learned. On the other hand, unsupervised learning is defined by the absence of predefined labels or outcomes, focusing instead on identifying patterns or structures within the data. However, reinforcement learning is employed when a sequence of actions is required to find a solution to a problem. The most widely used model is the supervised learning model where the input variable is derived from an input dataset, and the focus is on identifying patterns within the data. Figure 1 illustrates the main classification of machine learning algorithms.

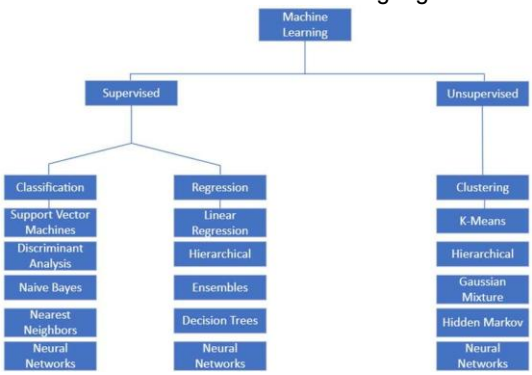


Figure 1: Classification of machine learning algorithms [1].

Neural networks are a branch of machine learning that is key to deep learning based on artificial intelligence. Neural networks (NN) consist of mutually densely connected processing elements (neurons) organized according to specific architectures. NN shows information processing with learning and generalization characteristics based on the training dataset. It is applied mainly to problems that require a clearly defined function that connects the input and output data [4].

This paper deals with neural networks with supervised learning. The general principle of the operation of neural networks is shown in Figure 2.

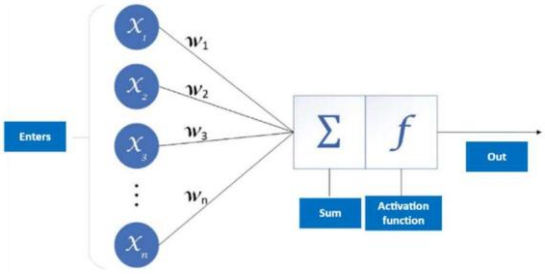


Figure 2: Artificial neuron [5].

Supervised learning defines the outputs with a specific mapping of the input (x) and output (y). Model creation begins by feeding a large amount of data to the input, which starts the model's training. A large amount of input data allows the model to be more accurate. The high accuracy of the model will enable us to have adequate prediction of the output values by providing test data to the model.

Each neuron has its input, which is connected through synapses. The inputs, shown as x_i where i=1...n, can be new unprocessed input data or data sent from another neuron. The strength of the connection between neurons and individual input data is not arbitrary but is precisely defined by synaptic weights. These weights play a crucial role in the transfer function, where the input multiplied by its synaptic weight enters. The result of the transfer function is then compared to a threshold value, triggering an activation function that gives a result of 1 or 0, the final output of the neuron [6].

The inputs received by neurons, are followed by activation function. Activation functions introduce non-linearity to the neural network, allowing it to model more complex classifications. These functions vary depending on the mathematical formulas used, determining how the input data is processed. The choice of activation function is crucial in the development of neural networks, as it directly affects the learning speed, as well as the accuracy and performance of the network [5]. Figure 3 provides examples of activation functions, including the sigmoid, rectified linear unit activation function (ReLU), and the hyperbolic tangent (tanh).

The sigmoid function is not zero-centred and has an exponent calculation. The corresponding formula for the function is found in [8]:

$$\text{Sigmoid}(x) = \frac{1}{1 + e^{-x}} \quad (1)$$

Another problem with this function arises during the training of neural networks. The gradients, which indicate how much a parameter should be adjusted to reduce the error, gradually decrease and eventually approach zero. This factor makes the learning model difficult. Saturation of the gradients is done by normalizing the data and weights and using activation function that do not lead to rapid saturation, such as ReLU. The

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

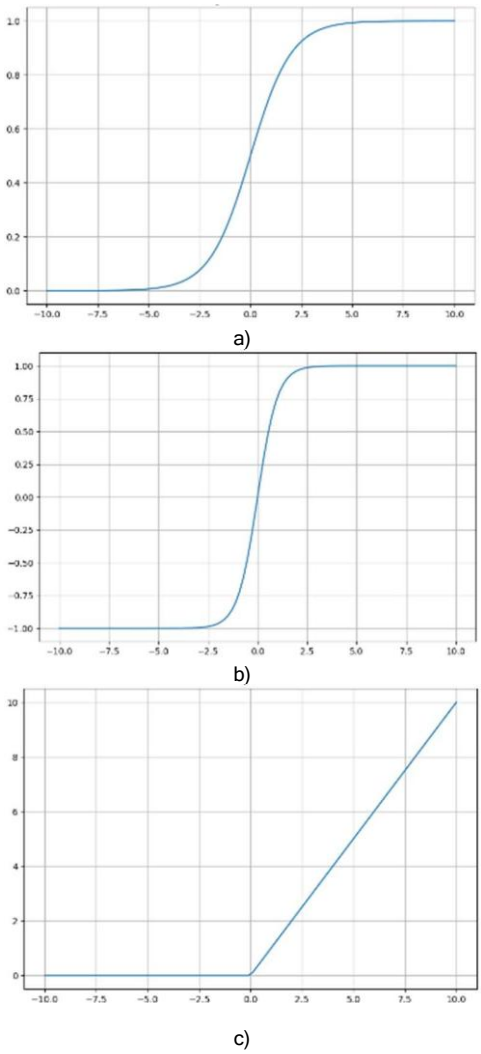
corresponding formula for the ReLU function is [9]:

$$\mathrm{ReLU}(x) = \{ \begin{matrix} 1\ if\ x > 0 \\ 0\ if\ x < 0 \end{matrix} \} \quad (2)$$

The hyperbolic tangent is an odd, monotonically increasing function. The function is centred at zero and is between -1 and 1. The corresponding formula for the function is:

$$\mathrm{Tanh}(x) = \frac{\sinh(x)}{\cosh(x)} = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (3)$$

The hyperbolic tangent function also has a problem with gradient saturation. When the input is greater than zero, the gradients will be positive or negative, causing them to disappear.



a)

b)

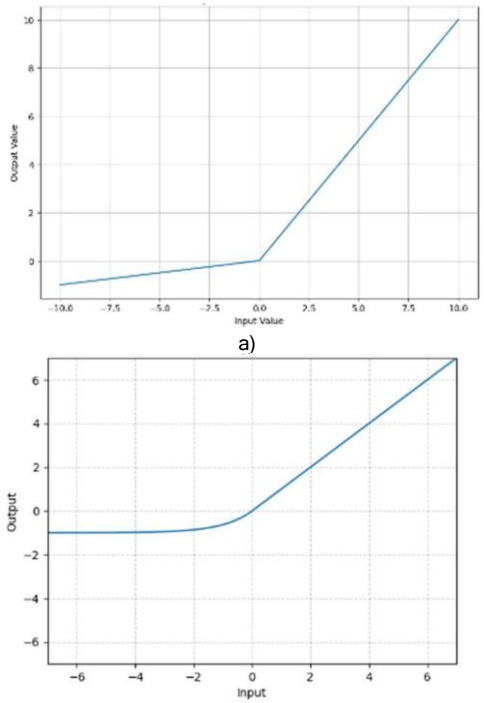
c)

Figure 3: Examples of activation functions are a) sigmoid, b) Tanh, and c) ReLU [7].

The ReLU activation function is the most used due to its simplicity in backward propagation and calculation. However, it has a major drawback: if the input is less than zero, the function outputs zero. This problem is addressed by introducing a slope constant, which prevents 'dead' ReLU, and accelerates learning by improving balance. The modified function, known as Leaky ReLU, is shown in Figure 4a.

The Maxout function uses the maximum value within a group of linear parts. Unlike ReLU, which compares the value to zero, Maxout compares it to the highest value within the candidate group. The exponential linear unit (ELU) is another variation of the ReLU function, offering improved performance for values of x <0 (Figure 4b). It shares similar properties with ReLU, but avoids the problem of dead ReLUs [3]. The corresponding formula for the ELU function is [9]:

$$\mathrm{ELU}(x) = \{ \begin{matrix} x & if\ x > 0 \\ a(e^{x} - 1)\ if\ x < 0 \end{matrix} \} \quad (4)$$



a)

b)

Figure 4: Examples of activation function a) Leaky ReLU and b) ELU [10].

The advantages and disadvantages of the activation functions are given in Table 1.

Table 1. Advantages and disadvantages of activation functions [6]

	Sigmoid	Tanh	ReLU	ELU
Reduction of gradient	yes	yes	partial	no
Limited non-linearity	no	no	yes	partial
Optimization difficulty	yes	partial	partial	no

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

	Sigmoid	Tanh	ReLU	ELU
Lack of adaptability	yes	yes	yes	yes
Computational inefficiency	yes	yes	no	partial

Neural network learning is carried out using algorithms known as backpropagation, which calculates the gradients necessary for adjusting the weight values in the network. Some of the most used algorithms include Adam, Adagrad, Nadam, Vanilla, and stochastic gradient descent among others. In this paper, the authors will focus on the Adam algorithm (Adaptive Moment Estimation), which, not only calculates its learning rate, but also incorporates momentum into the update by acting on the first-order gradient [6].

2. Related Works

The analysis of datasets, particularly general free datasets, and the use of appropriate algorithms have mainly been the focus of researchers. Along with various software packages, Attack detection systems, are commonly used to identify transmission anomalies, as discussed in numerous studies [11-23].

In his work, Maslan [11] focused on the concept of detecting Distributed Denial-of-Service (DDoS) attacks using machine learning techniques, applied to a real WEB server. According to the study, the reason for investigating this type of attack is that DDoS attacks account for 79% of all attacks in Malaysia. The techniques used in the study include naive Bayes (NB), random forest (RF), neural network (NN), support vector machine (SVM), and k-nearest neighbour (k-NN). The input dataset consists of typical attributes such as the source and destination addresses, packet size, packet type, and total number of packets. The analysis revealed that the highest accuracy was achieved using random forest and NN algorithms, reaching 98.70% using both. In comparison, other algorithms performed slightly worse, with naive Bayes achieving 97.96%, SVM 98.41%, and k-NN 97.63%. The NN used in the study consisted of two hidden layers in the 4-4 model configuration.

Studies such as Maslan [11] and Najafimehr [12] have focused mainly on traditional machine learning algorithms. For example, Najafimehr used publicly available datasets, including CICIDS2017 (DDoS subset) for training and CICDDoS2019 for testing. Two open-source datasets were used in his work: the CICIDS2017 dataset (DDoS subset) for training and CICDDoS2019 for testing. Both datasets contained 42 attributes each. Machine learning algorithms, including decision trees, random forests, naive Bayes, and support vector machines, were applied for the analysis. Among these, the random forest algorithm demonstrated the highest precision, achieving 99.6%, compared to the Decision Tree (DT) algorithm, which had a precision of 99.3%.

When comparing Maslan's study, which used real data, with Najafimehr's work with open datasets, it can be observed that real-world data tend to produce lower precision. This is due to the fact that, under real conditions, various factors come into play, including the type of network, the nature of the end node, and the specific type of attack.

In his work, Stephan [13] tried to use neural networks to set up a system for detecting attacks on a web server. The neural networks that were tested were defined with one hidden layer. When testing with 5 nodes in the hidden layer, the results obtained were 92.17% successful. Any deviation from the value of 5 nodes in the hidden layer dramatically degrades the pattern detection performance.

Tivari [14] works with the other dataset, NSL-KDD [15]. Dataset NSL-KDD is used for network security testing and functions analysis. Using classical machine learning models, Tivari came to the following results. Comparing these studies for precision, it could be concluded that the highest accuracy could be obtained by using artificial neural networks with a value of 99.4%. The next group of models has a similar accuracy of 95% and include the SVM, Passive Aggressive Classifier, and Ridge Classifier, random forest with 94% and naive Bayes with 89%. It is also worth mentioning the Decision Tree with 79% accuracy. As mentioned so far, high values come with public data sets. Using non-public datasets, values are different.

Mahmood [16] took the example of private cloud network attacks as another example of the use of machine learning algorithms. Threats generated against cloud services cannot be classified into a separate group, but are an identical type of threat against the end user. In this example, the following algorithms were used: kNN, decision tree, SVM, Random Forest, SGD, NN, Naive Bayes, logistic regression, gradient boosting, and AdaBoost. The traffic analysis was carried out continuously for 30 days. Based on a month-long analysis, results were obtained that confirm that the neural network algorithm does not lag behind other machine learning algorithms. Regarding percentages, the following accuracy results were obtained: neural network 87.6%, random forest 86.9%, logistic regression 87.7%, decision tree 87.7%, SVM 87.6%, etc. The data obtained confirm that the results are below 99% for real datasets compared to open datasets.

Lillmond [17], who analysed the deep neural

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

network, obtained better results in analysis. A deep neural network refers to networks with multiple hidden layers. The model used in this analysis had an action output attribute with four possible values: allow, deny, drop, and reset. The allow value was 57.4%, drop 19.6%, and deny 22.9%. Analysis revealed that the deep neural network achieved 94.49% precision for the test model and 95.81% accuracy for the training model.

Thi-Thu-Houng [18] conducted a test attack using several models that included a small attack within the dataset. The attack models were denial of service (DoS), user-to-root (U2R), and remote-to-local (R2L). In this experiment, Thi-Thu-Houng used an open-source dataset, the KDD Cup [19]. The model had 80 hidden layers and 500 epochs. The best results came from the Leaky ReLU function, which achieved an accuracy of 0.97, while the ReLU function yielded the lowest result with an accuracy of 0.94.

Valentin [20] used 20,000 instances to evaluate a dataset with actions in three states: allow, deny, and reject. The training datasets were divided into negative and positive examples in an 80:20 ratio. The best performance in the neural network was achieved with 13 hidden neurons.

Habibur [21] focused on analysing firewall traffic using real data logs from a firewall, consisting of approximately 67,000 logs. The paper discussed a method involving an activation function and two hidden layers with Models 3-4. The final prediction was 0.75, with better results obtained from the random forest and SVM models. The study did not specify whether two separate datasets were used, so it can be assumed that only one dataset was analysed.

Abien [22] worked with the standard MNIST dataset [23], which included 60,000 training examples and 10,000 test cases. The paper implemented two different classification functions: softmax and ReLU. To evaluate the performance of the ReLU model, several metrics were used, including accuracy, standard deviation, recall, precision, F1 score, and confusion matrix. Both functions yielded similar results for these metrics, ranging from 0.86 to 0.89, suggesting that Abien's future work may involve exploring deep-learning variations of the ReLU model.

These studies demonstrate that neural network algorithms can perform just as well as traditional machine learning algorithms. The addition of multiple layers allows for improved precision, a topic that will also be explored in this paper.

3. MATERIALS AND METHODS

In this paper, working equipment was used for the analysis to ensure that the results are in line with practical scenarios, which may differ from laboratory-based analyses. The Check Point firewall served as the central firewall system for the study. Firewalls are critical to preventing threats and offer protection against advanced attacks. The Check Point firewall provides several key functions, including deep learning capabilities, threat prevention (such as blocking zero-day DNS and phishing attacks), and protection of Internet of Things (IoT) devices. Furthermore, the firewall supports 2.5Gbps threat prevention throughput, which improves its overall performance in real-world applications [24].

The robot system (BOT) used in this study was based on open-source software from the github.com portal [25]. Wireshark software [26] was used to detect transmission anomalies. For the machine learning analysis, the Orange software package was utilized []. Various neural network algorithms were applied, including activation functions: identity, logistic, tanh, and ReLU. The datasets used for the analysis were gathered directly from the firewall system under real working conditions.

3.1. Network Model for Attack Analysis and Detection

A network, as shown in Figure 5, was used to demonstrate how machine learning affects the operation of the firewall system and how it improves performance in terms of better detection and blocking capabilities.

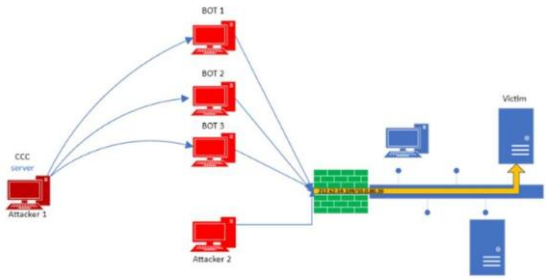


Figure 5: Layout of the analysis grid.

Figure 5 illustrates an attack on the internal zone by two actors. One machine (Attacker2) executes the attack directly, while the other control (CCC) the attack using BOT functions, engaging other users (BOTs) on the Internet to participate in the attack. The entire attack is executed using two different methods, ping flood and http flood. A BOT function is a software program that automatically performs repetitive and targeted tasks. BOT can be used for some business tasks but also for malicious purposes and could have a severe impact on the local network. There are several types of BOT attacks [27], which are as follows.

Credential stuffing is when attackers use stolen login credentials to gain access to another website. Bots circumvent existing built-in security features in web application login forms by attempting multiple simultaneous logins from various device types and IP addresses.

International journal of imaging science and engineering

ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

Web/content scraping is when bots download content from a website to use it in future attacks. A website scraper bot sends a series of HTTP GET requests, copies, and saves the information, all in seconds.

DoS and DDoS attacks are carried out with networks of Internet-connected machines, such as computers or IoT devices. Once the network is infected, attackers send remote instructions to each bot to overload the server or network, causing outages and downtime.

Brute force password cracking is an attack that uses bots to attack and infiltrate protected accounts by trying every possible password combination or cracking encryption key to gain unauthorized access to sensitive data.

Click fraud occurs when attackers target pay-per-click ads to boost the search rankings of a webpage through fake clicks.

This paper explores the use of bots to perform DDoS attacks on a local network server. The Python-based bot application includes server- and client-side code. Server-client communication (CCC and bots) typically occurs through port 5555, with options for alternative ports. Clients can launch two types of attack: http flood and ping flood, both implemented in the client-side code (see Listing 1). The server commands, outlined in Listing 2, determine which attack function is used. The attack source comprises the selected function and the target IP address.

Listing 1: Client-side code that implements http flood and ping flood attacks [23]

```
def run(self, n):
        run = 0
        #terminate = 0
        if n[3]=="HTTPFLOOD":
            while self._running and attackSet:
                url_attack = 'http://'+n[0]+':'+n[1]+'/'
                u =
urllib.request.urlopen(url_attack).read()
                time.sleep(int(n[4]))

        if n[3]=="PINGFLOOD":
            while self._running:
                if attackSet:
                    if run == 0:
                        url_attack = 'ping '+n[0]+' -i
0.0000001 -s 65000 > /dev/null 2>&1'
                        pro =
subprocess.Popen(url_attack,
stdout=subprocess.PIPE, shell=True,
preexec_fn=os.setsid)
                        run = 1             else:
                    if run == 1:
                        os.killpg(os.getpgid(pro.pid),
signal.SIGTERM)
                        run = 0
break
```

Listing 2: Server commands controlling client attack functions [23]

```
ATTACK_TARGET_HOST = "192.168.0.105"
ATTACK_TARGET_PORT = "3000"

# Type of Attacks
    #HTTPFLOOD - Floods the target system with
GET requests.
    #PINGFLOOD - Floods the target system with
ICMP echo requests.

ATTACK_TYPE = "PINGFLOOD"

#Status codes that must be set from the list below.
    # HALT - Stop attacks immediately.
    # LAUNCH - To immediately start the attack.
    # HOLD - Wait for the command.
    # UPDATE - Update Client.
ATTACK_CODE = "HALT"
```

In this paper, three datasets are defined for analysis. The first dataset represents the normal operation of the firewall system, capturing its status during periods without active attacks on specific groups or ports. This dataset serves as a training set. The second dataset focuses on detecting transmission anomalies during an active attack, making it the test set. Comparing the changes between these two datasets provides information on the firewall policies. The third dataset captures the data generated after the adjusted firewall policies are implemented, allowing the accuracy of anomaly detection and the effectiveness of the updated policies to be evaluated.

The dataset used for the analysis includes the following attributes:

- **destination**: specifies the destination IP address;
- **interface direction**: indicates incoming or outgoing traffic;
- **type**: identifies whether the session is a *connection*, *log*, or *connection alert*;
- **source**: displays the source IP address;
- **product**: categorizes the session as either a *threat* or *access*;
- **blade**: specifies the firewall component involved, such as antivirus, firewall, or VPN;
- **source** port;
- **destination** port;
- **protocol**;
- **action**: determines whether the session is *accepted*, *dropped*, *detected*, or *prevented*.

The *action* attribute can take on several values, including [1]:

- **allow:** allows communication between the source and destination addresses.
- **detect:** monitors specific traffic that bypasses initial detection.
- **deny:** blocks traffic between the source and destination due to policy restrictions and sends information to a sender.

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

- **drop:** blocks traffic between the source and destination without notifying the sender. This is often preferred for blocking potentially malicious traffic.
- **prevent:** stops unauthorised or malicious traffic targeting the destination address.

Figure 6 presents the network model implemented in the Orange software package. This model is designed for the application of machine learning algorithms, which aligns with the focus of this study. The model utilises three datasets: no-attack, attack, and defence. The no-attack dataset represents the network state when no attack is occurring and is used to train the model. This dataset contains approximately 54,500 log instances. Figure 7 illustrates the numerical ratio of all three output states. The attack dataset represents the state of the network during an ongoing attack and is used to evaluate the model. This dataset contains approximately 64,100 log instances. Figure 8 illustrates the numerical distribution of all three output states. The defence dataset represents the network state after the firewall policy has been corrected. It serves as an additional analysis to evaluate the firewall system's operation, determining whether the policy has been applied successfully and whether the corrections have been effective. This dataset contains approximately 62,800 log instances. Figure 9 illustrates the numerical ratio of all three output states.

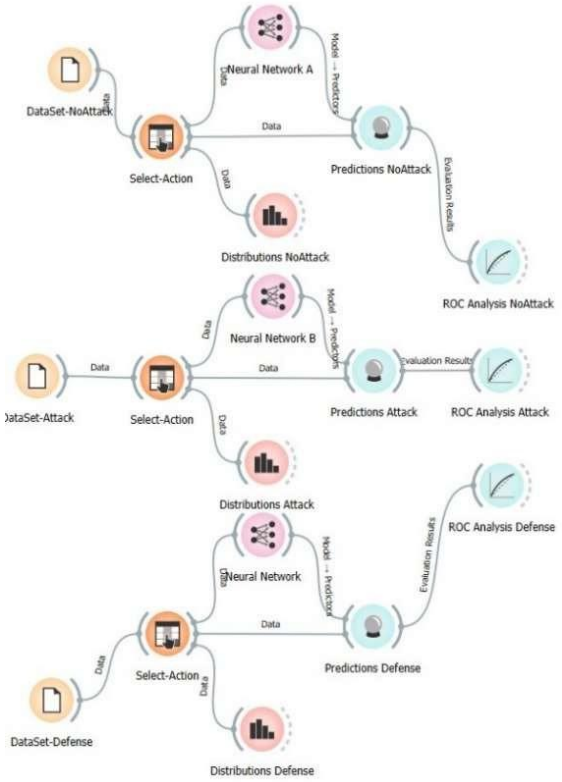


Figure 6: Network model created in the Orange package.

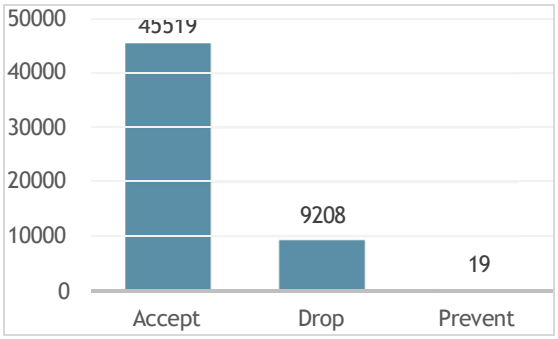


Figure 7: Numerical ratio of output states in the no-attack dataset.

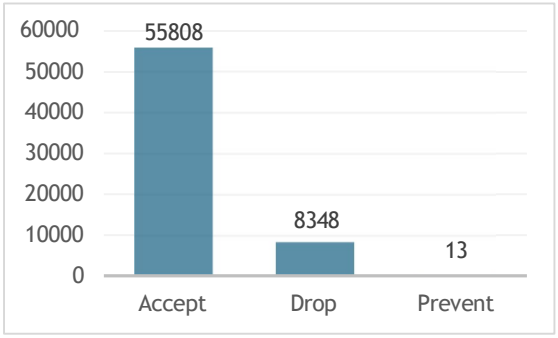


Figure 8: Numerical ratio of output states in the attack data set.

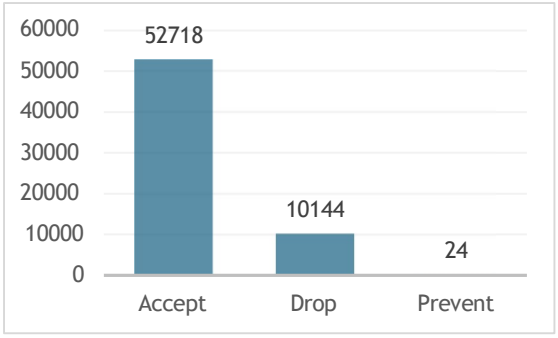


Figure 9: Numerical ratio of output states in the defence dataset.

The initial analysis of the three datasets reveals the behaviour of the *allow* exit function. Figure 10 shows the total number of all functions. The graph highlights an increase in the volume of the packets during the attack, followed by an increase in discarded packets after the defence is applied.

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

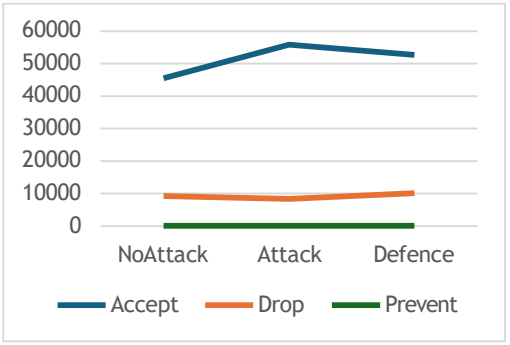


Figure 10: Total packet counts showing increases during the attack and defence phases.

Table 2 presents the percentage ratios for the allow, drop, and prevent outputs, calculated relative to the total number of output values. It shows that as the percentage of allowed packets increased from the no-attack dataset to the attack dataset, meaning the firewall did not respond initially. For smaller attacks, the firewall did not register significant concern. However, after adjusting the firewall policy, there was a reduction in allowed packets and a corresponding increase in rejected packets, as illustrated in Figure 11.

Table 2. Percentage ratios of allow, drop, and prevent outputs calculated relative to the total number of output values

	allow/all	drop/all	prevent/all
no-attack	83%	16.8%	0.034%
attack	86.9%	13%	0.020%
defence	83.8%	15.8%	0.038%

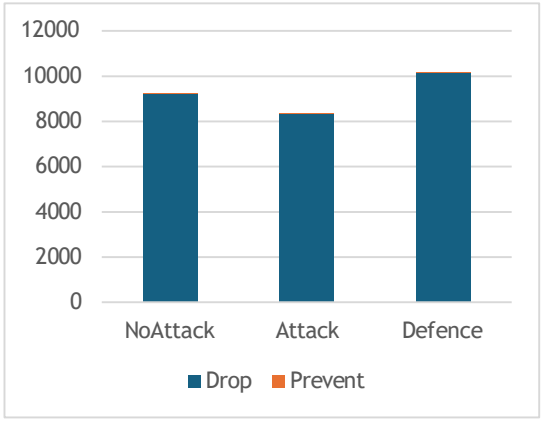


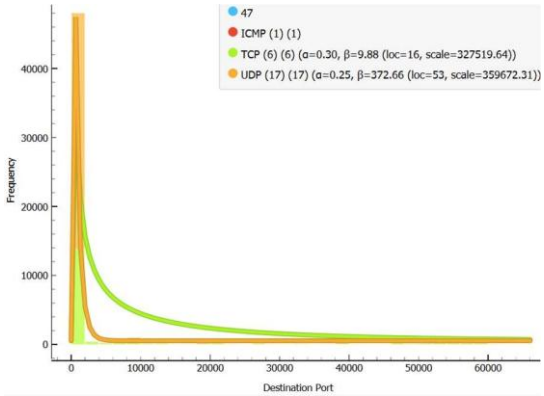
Figure 11: Change in allowed and rejected packets after firewall policy adjustment (the value of the Prevent state is quite low).

The first analysis of the neural network, focusing on anomaly detection to predict whether an attack has occurred, is performed using the distribution unit in the Orange application. For this purpose, data fitting was applied using the beta distribution. The beta distribution is used to model the behaviour of random variables constrained to finite intervals across various fields. It is defined in the interval [0, 1], with two positive parameters, *alpha* (α) and *beta* (β), which serve as exponents of the variable. The probability density function (PDF) of the beta distribution for variable x and the shape parameters α and β is given by the following equation:

$$f(x,\alpha,\beta) = \frac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1} \quad (5)$$

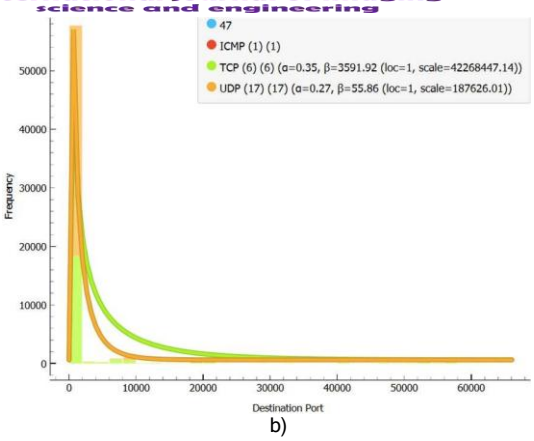
A change in several characteristics is observed in the TCP and UDP packets, as shown in Figure 12. The figures indicate values for the Generic Routing Encapsulation (GRE) protocol, labelled as value 47, and the Internet Control Message Protocol (ICMP). However, the values for these protocols are negligible. As shown in Figure 12, an increase in activity is observed in the UDP ports, suggesting additional network activity. To further confirm this, Figure 13 shows the increase in the number of UDP sessions, indicating a significant increase in UDP traffic. Given that static NAT is configured in the firewall system, the local IP address 10.0.80.20 is assigned to the public IP address (not shown for security reasons), where specific attacks are made against the public IP address. Figure 14 shows multiple connections from a single public IP address to the local server at 10.0.80.20.

A more detailed analysis of the number of sessions and associated IP addresses was conducted using Python programming. The analysis covered a 10-minute period. The results of this analysis, shown in Figures 15 and 16, highlight the findings derived from the Python scripts.



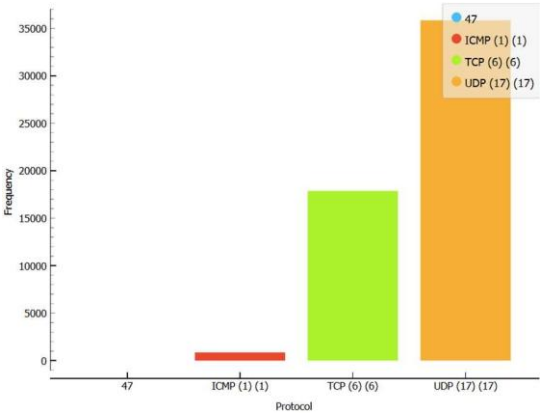
a)

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

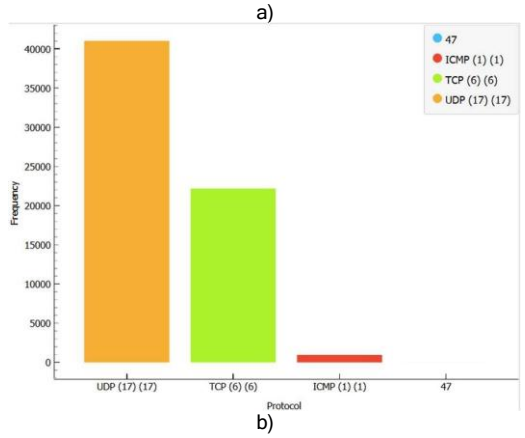


b)

Figure 12: Examples of TCP and UDP views a) before and b) after the attack.



a)



b)

Figure 13: Displays of the number of received packets a) before and b) after the attack.

	TIME	Source	Destination	Protocol
148	1.217593	10.0.80.20	[redacted]	IPv4
149	1.217593	10.0.80.20	[redacted]	IPv4
150	1.217593	10.0.80.20	[redacted]	IPv4
151	1.217593	10.0.80.20	[redacted]	IPv4
152	1.217593	10.0.80.20	[redacted]	IPv4
153	1.217593	10.0.80.20	[redacted]	IPv4
154	1.217593	10.0.80.20	[redacted]	IPv4
155	1.217593	10.0.80.20	[redacted]	IPv4

Figure 14: Display of connections on the local server.

Destination	
067-078-135-202.biz.spectrum.com (67.78.135.202)	1
3.215.163.101	1
3.222.69.75	1
3.223.146.149	1
3.248.30.115	1
...	
[redacted]	1084
host_192.168.1.2 (192.168.1.2)	1745
ns1.ptt.rs (212.62.32.1)	2259
dns.google (8.8.4.4)	10704
dns.google (8.8.8.8)	11735

Name: Source, Length: 1682, dtype: int64

Figure 15: Display of the public IP address as destination (attack victim).

Source	
1.190.205.113	1
205.210.31.70	1
205.210.31.68	1
205.210.31.49	1
...	
[redacted]	973
[redacted].dynymic.isp.telekom	994
Zeleznice_LDAP_server_2 ([redacted])	1306
host_10.0.4.145 ([redacted])	1710
host_192.168.1.15 (192.168.1.15)	2716

name: Source, Length: 1766, dtype: Int64

Figure 16: Display of the public IP address as source (attacker).

When examining the events on the incoming firewall, it is observed that there is activity originating from a specific public IP address directed at the public IP address of the server exposed to the Internet. The next step is to analyze the server using various numerical parameters. When Wireshark data are transmitted during the ongoing attack, a notable similarity can be observed in the number of packets between the local IP address and the public IP address on the Internet, as shown in Figure 17.

Detecting a high volume of sessions requires an analysis of the use of the protocol to block potential attacks on the firewall. Figure 18 reveals that the analysis identified the ICMP protocol as being used in a ping flood attack.

Source	
10.0.80.100	1
108.137.69.146	1
121.233.168.23	1
128.201.219.13	1
16.78.30.255	1
...	
fe80::8f71:98ef:cf4a:c4c6	1211
10.0.80.105	1228
10.0.80.240	8394
[redacted]	60524
10.0.80.20	64809

name: Source, Length: 1766, dtype: Int64

Figure 17: Number of sessions display between the server and the attacker.

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

```
Protocol
OpenVPN                1
LSD                    4
CDP                    8
DCERPC                24
BROWSER               25
DHCPV6                30
LOOP                  47
UDP                   79
NBNS                 130
SNMP                 188
STP                  236
LLMNR                388
SSDP                 422
ICMPV6               530
ARP                 1545
MDNS                3961
TLSV1.2             4012
ICMP                4523
TCP                10230
IPV4               115074
name: Source, dtype: int64
```

Figure 18: Display of the server-side protocols.

With two key variables, the type of attack and the IP address from which the attack originates, the firewall system can be configured to block the attack. The process of setting the policy begins at the firewall, where you choose to apply the policy between the outside and inside zones. In the first step, for the predefined Check Point firewall solution, the following parameters must be set:

- **policy name**: This attribute does not affect the policy's functionality. However, when many policies follow the "top-down" execution principle (where the first named policy is executed first), the policy name plays a significant role in the execution order.
- **source and destination addresses**: These fields directly impact the policy's functionality. Specifically, sessions between the specified IP addresses will be blocked, as recorded by the firewall device. Care must be taken to avoid conflicts with IP addresses from other policies, ensuring the proper functioning of certain services.
- **protocol settings**: The final part of the policy defines which protocols are involved, whether they should be blocked, and whether a log entry should be created when the policy is triggered.

After explaining the firewall system, this work will explore the concept of the neural network and evaluate the effectiveness of the protection mechanism it provides.

3.2 Attack Analysis through Machine Learning Algorithms

In the following sections of the paper, the impact of predefined datasets on the neural network will be explored, which plays a crucial role in the operation of this network.

Weights and Biases**:** Weights are parameters that determine the strength of connections between neurons in different layers of the neural network. Each connection between neurons has an associated weight, which is updated during training through optimization algorithms. The weight value is learned during this process and plays an important role in the performance of the network. Bias, on the other hand, is a parameter that helps the model better understand the data. It is added to the weighted sum of inputs in each neuron, allowing the network to account for discrepancies between the predicted and actual outputs. Like weights, biases are also learned during the training process and contribute to the network's performance optimization.

Number of neurons: Neural networks can consist of multiple layers, having different structures, which influencing the model's ability to learn complex functions. Currently, there are no definitive recommendations on the optimal number or types of hidden layers needed to achieve satisfactory results. To evaluate the data presented in this study, several models with varying numbers of neurons per layer were tested, including models with two and three layers. The accuracy results for the neural network, using the training dataset, are presented in Table 3.

The best precision was achieved using a two-layer neural network with a configuration of 5-3, as demonstrated by Čisar in [1], thus confirming the effectiveness of the two-layer model. Further analysis with a three-layer model showed improved precision compared to the two-layer model, specifically with a 5-5-4 configuration for our data. The structure of the neural network, based on the number of neurons per layer, is illustrated in Figure 19.

The parameters of the neural network are influenced by the pre-defined algorithms. For the dataset used in this study, an accuracy comparison was performed with the proposed algorithm. The results of this comparison are presented in Table 4.

Table 3. Accuracy value related to the number of neurons and layers

Number of neurons and layers	Accuracy
3-5	0.940
4-5	0.945
5-5	0.954
5-3	0.957
5-4	0.949
5-4-3	0.956
5-5-3	0.960
5-5-4	0.963
6-5-4	0.954
4-5-4	0.946
4-5-5	0.960

International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

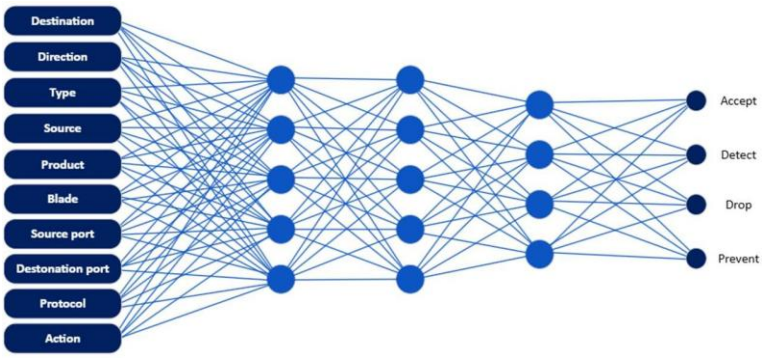


Figure 19: Structure of the neural network with the number of neurons per layer (5-5-4 configuration).

Table 4 Accuracy comparison for the proposed activation functions

Algorithm NN	AUC	CA	F1	Precision
Identity	0.950	0.942	0.966	0.935
Logistic	0.965	0.964	0.979	0.947
Tanh	0.976	0.964	0.997	0.961
ReLU	0.977	0.964	0.979	0.963

In Table 4 we have small deviations in the values for the displayed functions. There are deviations at the third decimal point, which indicates that the final values in the further examination will have small cuts in the values, for the mentioned functions. Based on the results obtained, the next phase of the analysis will focus on the ReLU function and the 5-5-4 neural network model. After defining the model, along with the number of neurons, layers, and algorithms, further network tests can be conducted to improve the performance of the firewall solution. The following characteristics were considered during this process.

Area under the ROC (Receiver Operating Characteristic) curve (AUC): The AUC represents the area under the ROC curve, which assesses the model's ability to distinguish between true positives and false positives. Values range from 0 to 1, where a value of 1 indicates that the model perfectly separates the specified classes. The AUC is always a positive number and can be calculated using the following equation:

$$AUC = \int_0^1 ROC(u)\,du \qquad (6)$$

In this case, in Table 5, an example of weighting values for all three cases is given. The average values dropped by 4.61% compared to the training data. However, after correcting for the firewall solution, it almost returned to its initial values. Figure 20 shows the ROC curve for the drop value in the three datasets. In this figure, the graph reveals a decrease in the area of the ROC curve during a system attack. These changes ar work and models coulde also reflected in Table 5. By taking prompt action on the firewall features, the system returned largely to its initial state.

Classification Accuracy (CA): CA represents the proportion of correctly classified examples related to the total number of examples. The formula for CA is as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \qquad (7)$$

where TP is true positive, TN is true negative, FP is false positive, and FN is false negative. Table 6 presents the CA values. When comparing the test model to the training model, the CA drops by 1.24%, as shown in Table 6. Similarly to the AUC parameter, there is no return to the initial values after intervention in the firewall system.

F-score (F1): The F-score measures predictive performance, particularly when dealing with unbalanced datasets. The results are presented in Table 7, for the described model. A 1.56% drop related to the training model is observed, but the values return close to the nominal after the firewall intervention.

Precision: Precision represents the proportion of true positive events among the cases classified as positive. The results are shown in Table 8.

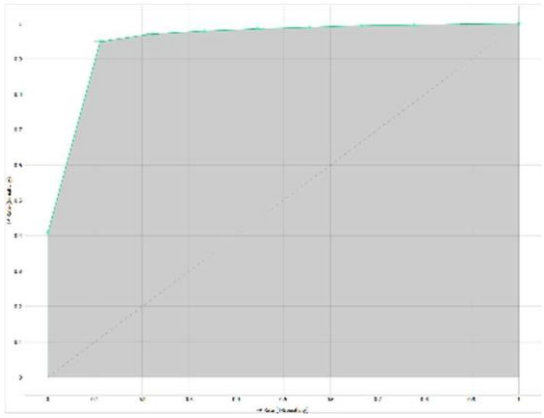
Recall: Recall represents the proportion of true positive events among all positive instances. The results are provided in Table 9.

The data was recorded in three cycles. The first cycle captured characteristics when there was no attack on the system. The second cycle recorded the state during the attack and was used for training. The third cycle, confirmation, recorded the firewall status after the necessary corrections were made.

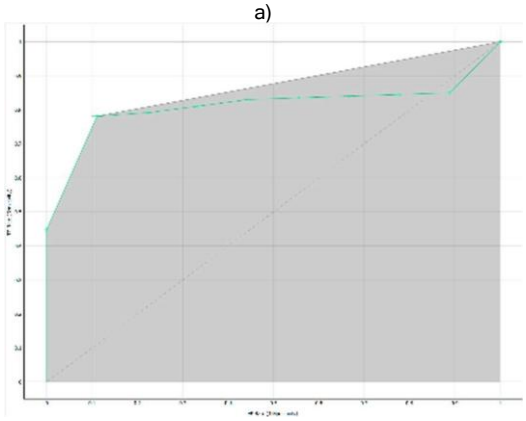
Table 5. AUC values across three dataset scenarios

Target	Training	Testing	Conformation
Accept	0.977	0.930	0.975
Detect	1	1	1
Drop	0.977	0.932	0.977
Prevent	1	1	1
Average	0.977	0.932	0.977

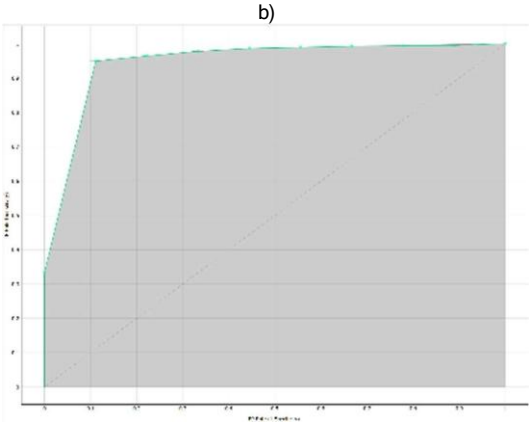
International journal of imaging science and engineering
ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar



a)



b)



c)

Figure 20: Values of the ROC curve for the drop function for a) test, b) training, and c) confirmation mode.

Table 6. CA values for datasets

Target	Training	Testing	Conformation
Accept	0.964	0.952	0.963
Detect	0.964	0.952	0.963
Drop	0.964	0.952	0.963
Prevent	0.964	0.952	0.963
Average	0.964	0.952	0.963

Table 7. F-score values showing the impact of firewall intervention

Target	Training	Testing	Conformation
Accept	0.978	0.973	0.978
Detect	0	0	0
Drop	0.886	0.791	0.878
Prevent	1	0.839	1
Average	0.963	0.949	0.962

Table 8. Precision values for datasets

Target	Training	Testing	Conformation
Accept	0.970	0.957	0.968
Detect	0	0	1
Drop	0.927	0.908	0.931
Prevent	1	0.722	1
Average	0.963	0.950	0.962

Table 9. Recall values for datasets

Target	Training	Testing	Conformation
Accept	0.987	0.989	0.988
Detect	0	0	1
Drop	0.848	0.701	0.831
Prevent	1	1	1
Average	0.964	0.952	0.963

4. *Comparing the results for other functions*

The results showed that by using the ReLU function we can optimize the firewall device. Table 10 shows the results of other functions in neural networks for the situation of an attack on the system and the Accept functionality option.

Table 10. Values of Accept results for all functions

Target	AUC	CA	F1	Precision	Recall
ReLU	0.930	0.952	0.973	0.957	0.989
Tanh	0.930	0.946	0.969	0.955	0.983
Logistic	0.910	0.949	0.971	0.958	0.958
Identity	0.913	0.936	0.964	0.937	0.994

Suppose the focus is on the values of the CA parameters, the parameter that shows the correctly classified values. Then it could be determined that the ReLU is better than Tanh, Logistic and Identity quantified in percentage differences of 0%, 0.6% and 1.7% (respectively). In Figure 21, it can also be seen that ReLU is mostly better for the other parameters, except for the Recall parameter.

Noting the value of the Drop in Table 10, for all functions, similar conclusions are reached, i.e. that ReLU gives better results compared to the Tanh, Logistic and Identity functions quantified in percentage differences of 0.6%, 0.4% and 0.7%, respectively. Defined deviations can also be seen

ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

in Figure 22.

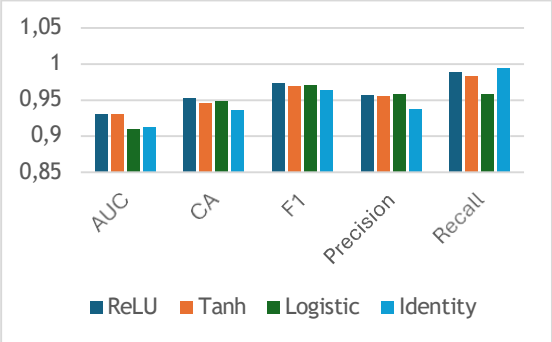


Figure 21: Values of different parameters for activation functions

Table 10. Values of Drop results for all functions

Target	AUC	CA	F1	Precision	Recall
ReLU	0.932	0.952	0.791	0.908	0.701
Tanh	0.929	0.946	0.768	0.862	0.693
Logistic	0.910	0.949	0.782	0.782	0.709
Identity	0.913	0.936	0.692	0.931	0.505

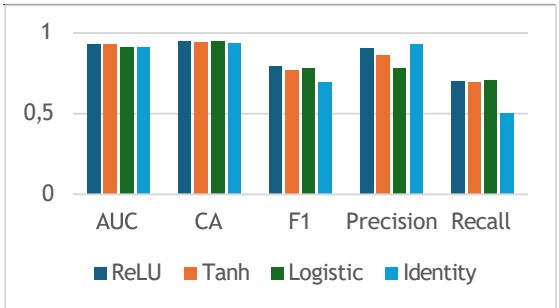


Figure 22: Values of defined deviations.

5. Comparing the results with a larger dataset

In the paper, the analysis was done with a small number of samples, up to 54,500 defined sessions. The values of some parameters are quite high. Table 11 shows all parameters within the ReLU function, but for a data set containing 1 000 000 sessions.

Table 11. Values for complete data set

Target	AUC	CA	F1	Precision	Recall
Accept	0.809	0.837	0.896	0.820	0.987
Detect	1.000	0.837	0.782	0.642	1.000
Drop	0.998	0.837	0.997	1.000	0.993
Prevent	0.967	0.837	0.000	0.000	0.000
Reject	0.753	0.837	0.357	0.834	0.227

Due to the very difficult option of comparing two data sets whose values are quite different, the comparison will be made only according to the parameter CA, i.e. according to the parameter of correctly classified examples.

Looking at the CA values in Table 11, we can state that there was a drop in correctly classified sessions by about 12%. This large value indicates that for better detection of malicious sessions, it is necessary to use a smaller data set to improve accuracy.

6. Conclusion

This paper demonstrates the application of neural networks with the ReLU activation function aiming to optimize firewall policies for detection and mitigation of network attacks. The proposed 5-5-4 model achieved the highest accuracy of 96.3%, surpassing two-layer architectures by 0.6% and the next-best three-layer architecture by 0.3%. The analysis, based on real working environment datasets, confirmed that this approach improves the accuracy of attack detection and improves the effectiveness of firewall policy adjustments, even in dynamic and complex environments.

During the testing, a decrease in performance metrics was observed: The area under the curve (AUC) decreased by 4.61%, the classification precision (CA) by 1.24%, the F1 score by 1.56%, Precision by 1.3%, and recall by 1.2% compared to the training values. After applying optimized firewall policies, most parameters returned close to their initial values, demonstrating the reliability of the proposed model in real working scenarios.

The study used datasets consisting of approximately 54,500 instances for training, 64,100 instances during active attacks, and 62,800 instances after firewall policy adjustments. This ensures the results are based on realistic conditions and it validates the practical applicability. These findings are particularly relevant in environments such as IoT systems, where quick detection and prevention of attacks are essential to maintain network security.

Although the results are encouraging, the study is limited by the use of a single dataset and software platform. Future research should focus on evaluating the model's adaptability to larger and more diverse datasets (IoT) and exploring its integration into broader cybersecurity frameworks to enhance scalability and applicability.

This study confirms the potential of machine learning, particularly neural networks, as an effective tool for improving network security. The proposed approach provides a solid foundation for further research and development aimed at creating more adaptive and robust cybersecurity solutions for increasingly complex and interconnected systems.

International journal of imaging science and engineering

ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

REFERENCES

[1] Čisar, P., Popović, B., Kuk, K., Čisar, S., Vuković, I., "Machine Learning Aspects of Internet Firewall Data," Springer, 10.1007/978-94-024-2174-3_4. 2022.

[2] Demsar, J., Curk, T., Erjavec, A., Gorup, C., Hocevar, T., Milutinovic, M., Mozina, M., Polajnar, M., Toplak, M., Staric, A., Stajdohar, M., Umek, L., Zagar, L., Zbontar, J., Zitnik, M., Zupan, B., "Orange: Data Mining Toolbox in Python," Journal of Machine Learning Research, vol. 14, no. Aug, pp. 2349-2353. 2013.

[3] Petrović, M., "Osnovi veštačkih neuronskih mreža i značaj njihove primene," Zbornik radova Građevinskog fakulteta, Subotica, no. 20, pp. 47-55. 2011.

[4] Knežević, S., Mileta, Ž., Žarković, "Predviđanje proizvodnje termoelektrane pomoću neuralnih mreža," Energija, ekonomija, ekologija, vol. XXV, pp. 38-41. 2023. https://doi.org/10.46793/EEE23-4.38K

[5] Kramberger, T., Nozica, B., Dodig, I., Cafuta, D., "Pregled tehnologija u neuronskim mrežama," 10.19279/TVZ.PD.2019-7-1-04. 2019.

[6] Nikolić, M., Zečević, A., "Machine learning," Faculty of Mathematics, Belgrade, Serbia, 2019.

[7] Coraline Ada Ehmke, "How Do Neural Networks Make Decisions: A Look at Activation Functions," Accessed: Nov. 23, 2024, Available: https://www.goglides.dev/bkpandey/how-do-neural-networks-make-decisions-a-look-at-activation-functions-141e

[8] Dubey, S.R., Singh, S.K., Chaudhuri, B.B., "Activation functions in deep learning: A comprehensive survey and benchmark," Neurocomputing, vol. 503, pp. 92-108. 2022. https://doi.org/10.1016/j.neucom.2022.06.111

[9] Bai, Y., "RELU-Function and Derived Function Review," SHS Web of Conferences, vol. 144, 02006. 2022. https://doi.org/10.1051/shsconf/202214402006

[10] PyTorch Contributors, Accessed: Nov. 23, 2024, Available: https://pytorch.org/docs/stable/generated/torch.nn.ELU.html

[11] Maslan, A., Mohamad, K., Mohd Foozy, C.F., "Feature selection for DDoS detection using classification machine learning techniques," IAES International Journal of Artificial Intelligence (IJ-AI), vol. 9, pp. 137-145. 2020. https://doi.org/10.11591/ijai.v9.i1.pp137-145

[12] Najafimehr, M., Zarifzadeh, S., Mostafavi, S., "A hybrid machine learning approach for detecting unprecedented DDoS attacks," J Supercomput., vol. 78, no. 6, pp. 8106-8136. 2022. https://doi.org/10.1007/s11227-021-04253-x

[13] Stephan J, Sahab M., Abbas M., "Neural network Approach to Web Application protections", International Journal of Information and Education Technology, Vol. 5, No. 2, February 2015

[14] Tiwari S., Kumar N., Joshi K., Kumar S., "Enhancing Cyber Security: A Comparative Study of Artificial Neural Networks and Machine Learning for Improved Network Vulnerability Detections ", Advanced technologies for realizing sustainable development goals 5G, AI. Big Data, Block Chain and Industry 2.0 applications, Bentham Books, Singapore, 2024.

[15] Ghulam Mohi-ud-din, December 29, 2018, "NSL-KDD", IEEE Dataport, doi: https://dx.doi.org/10.21227/425a-3e55.

[16] Mahmood, S., Hasan, R., Yahaya, A., Hussain, S., Hussain, M., "Evaluation of the Omni-Secure Firewall System in a Private Cloud Environment," Knowledge, vol. 4. 2024. https://doi.org/10.3390/knowledge4020008

[17] Lillmond, C., Suddul, G., "A Deep Neural Network Approach for Analysis of Firewall Log Data," 10.13140/RG.2.2.27458.04808. 2021.

[18] Le, T.-T.-H., Kim, J., Kim, H., "Analyzing Effective of Activation Functions on Recurrent Network for Intrusion Detection," JMIS, vol. 3, no. 3, pp. 91-99. 2016. https://doi.org/10.9717/JMIS.2016.3.3.91

[19] Stolfo, S., Fan, W., Lee, W., Prodromidis, A., Chan, P., "KDD Cup 1999 Data," UCI Machine Learning Repository, Accessed: Nov. 23, 2024, Available: https://doi.org/10.24432/C51C7N

[20] Valentin, K., Maly, M., "Network firewall using artificial neural networks," Computing and Informatics, vol. 32, pp. 1312-1327. 2013.

[21] Rahman, M.H., Islam, T., Rana, M.M., Tasnim, R., Mona, T., Sakib, M., "Machine Learning Approach on Multiclass Classification of Internet Firewall Log Files," 10.48550/arXiv.2306.07997. 2023.

[22] Agarap, A.F., "Deep Learning using Rectified Linear Units (ReLU)," 10.48550/arXiv.1803.08375. 2018.

[23] LeCun, Y., Cortes, C., Burges, C., "MNIST Handwritten Digit Database," AT & T Labs, vol. 2, Accessed: Nov. 23, 2024, Available: http://yann.lecun.com/exdb/mnist

[24] Checkpoint, Accessed: Nov. 23, 2024, Available: https://www.checkpoint.com/

[25] Shankar Narayana Damodaran, Accessed: Nov. 23, 2024, Available: https://github.com/skavngr/netbot

[26] "Wireshark: Network Analyzer," Accessed: Nov. 23, 2024, Available: https://www.wireshark.org

[27] Cloudflare, Accessed: Nov. 23, 2024, Available: https://www.cloudflare.com/learning/bots/what-is-a-bot-attack

Dragan Jevtić, Infrastructure of Serbian Railway, Belgrade, Serbia (e-mail: dragan.jevtic@srbrail.rs)
Petar Čisar, University of Criminal Investigation and Police Studies, Belgrade, Serbia, (e-mail: petar.cisar@kpu.edu.rs), 0000-0002-8129-288X,
John von Neumann University, GAMF Faculty of Engineering and Computer Science, Kecskemét, Hungary (e-mail: csiszar.peter@nje.hu)
Kristijan Kuk, University of Criminal Investigation and Police Studies, Belgrade, Serbia, (e-mail: kristijan.kuk@kpu.edu.rs), 0000-0001-8910-791X
Vladica Stojanović, University of Criminal Investigation and Police Studies, Belgrade, Serbia, (e-mail: vladica.stojadinovic@kpu.edu.rs) 0000-0002-3819-4387

Dragan Jevtić is a PhD student at the University of Criminal Investigation and Policy Studies. Currently working as Project Manager in the IT Department, Infrastructure of Serbian Railways. His research is mainly focused on using machine learning in various situations in railway environments, ranging from detecting attacks (from the Internet and the intranet) by rapidly analysing large amounts of data to investigating anomalies in transmission or attack itself.

Petar Čisar graduated from the University of Belgrade School of Electrical Engineering and earned a PhD in Information Sciences from University of Novi Sad. He is a full professor at the University of Criminal Investigation and Police Studies, Belgrade, and an associate professor at John von Neumann University, Kecskemét. A member of the International Society for the Implementation of Fuzzy Theory in Budapest and an external member of the Hungarian Academy of Sciences and Arts, he has authored over 150 scientific papers, with 400+ independent citations. His research focuses on computer and telecommunication networks, network security, digital forensics, and AI implementations.

International journal of imaging science and engineering

ISSN: 1934--9955 www.ijise.net
Vol-20 Issue-01 Mar

Kristijan Kuk earned his M.Sc. from the Technical Faculty in Zrenjanin, University of Novi Sad, and his PhD in Informatics and Computing Science from the Faculty of Electronic Engineering, University of Niš. He is a full professor at the Faculty of Computer Science and Information Technology, University of Criminal Investigation and Police Studies in Belgrade. He has authored over 15 papers that have been published in scientific journals from SCI/E lists; two papers have been published as a chapter for Springer and two chapters for Elsevier books. His research interests include intelligent agents, data mining techniques, and secure software development.

Vladica Stojanović is a full professor at the University of Criminal Investigation and Police Studies in Belgrade. He graduated from the Department of Mathematics at the Faculty of Philosophy in Niš and completed his master's degree in 2004. He received his PhD from the Faculty of Sciences in Kosovska Mitrovica. His research focuses on statistics, probability theory, time series analysis, data analysis, and computational physics. In 2023, he was awarded the Outstanding Reviewer Award for the *Mathematics* journal.