KAPITEL 8 / *CHAPTER 8*
DETERMINATION OF NETWORK ATTACKS USING NEURAL NETWORK TECHNOLOGIES
ВИЗНАЧЕННЯ МЕРЕЖЕВИХ АТАК З ВИКОРИСТАННЯМ НЕЙРОМЕРЕЖНИХ ТЕХНОЛОГІЙ

DOI: 10.30890/2709-2313.2021-07-08-003

Introduction

Formulation of the problem. Intrusion-Detection Systems (IDS) are used to detect network attacks in real time. In the information and telecommunication system (ITS) of railway transport, the problem of a large volume of network traffic arises, since standard approaches to data processing cease to be effective. One of the most effective approaches to classifying a large amount of data is the use of neural network technology. This approach allows detecting not only already known network attacks, but also detecting new ones.

Analysis of recent research. The most promising areas are IDS, built on the basis of neural networks (NN): multilayer perceptron (Multi Layer Perceptron, MLP); Radial Basis Function Network (RBF), Self Organizing Maps (SOM) and Adaptive-Network-Based Fuzzy Inference System (ANFIS). For example, [23] analyzes only DDoS attacks using TCP, UDP and ICMP protocols due to their popularity among attackers, in [24] some threats were detected in the network based on the analysis and processing of parameters of network connections using the stack of TCP/IP protocols, using a neural network configuration 19-1-25-5 (19 – the number of initial neurons ; 1 – the number of hidden layers; 25 – the number of hidden neurons; 5 – the number of resulting neurons), but other types of attacks also require research.

At the present stage, on the one hand, there are more and more works [1, 12-14, 17-18] that use a combined approach to solving the problem. For example, [14] proposes a new ensemble classifier that uses RBF and fuzzy clustering to increase detection accuracy, reduce false positives, and provide a higher detection rate for infrequent attacks. In [1] the approach with use of neural networks, immune systems, neurofuzzy classifiers and their combinations is considered. The essence of hybrid approaches is to implement various schemes of combining basic classifiers, which allow eliminating shortcomings in their operation separately. However, at the same time an important disadvantage of such techniques is the lack of universality of their application. In [12] to improve the efficiency of IDS it is proposed to use the method of coincidence, based on the fact that different types of NN (MLP, RBF, SOM) can detect different attacks, but erroneous triggers also do not always occur on the same network packets. Analysis using different types of NN. In addition, each type of neural network has its advantages, the disadvantages that need to be considered or additional research.

On the other hand, attempts are being made to use NN at different levels. For example, [5] considered a new approach to building a multilevel network intrusion detection system, which consists in the fact that groups of similar parameters between network interactions are fed to the inputs of individual first level modules, each of which represents a hierarchical structure of several different NN type and performs detection of anomalies on a given group of parameters. The results of the first level modules are fed to the input of the second level solver, which makes the final decision on the presence of the attack and its classification. According to this approach, the probability of identifying known attacks was 91 %, detection of intrusions, information about which was not available during training, was 86 %. However, the developed prototype has a relatively significant probability of error of the second kind of 18 %, analysis and correction of the causes of these errors is promising for further study.

The purpose of this work is to develop a method for determining network attacks. To achieve this goal ***the following tasks*** were solved:

– to determine the optimal parameters of some types of neural networks, which will provide a sufficiently high level of reliability of detection of intrusions into the computer network;

– conduct research on different approaches (single-level and two-level) to detect network attacks;

– explore a combined version of network attack detection that combines the use of neural network technology with immunology.

8.1. Problem statement and database selection

Attacks are divided into four main categories [19]: DoS, U2R, R2L, Probe.

A Denial of Service (DoS) attack is an attempt to harm by making the target system, such as a website or application, inaccessible to ordinary end users. Typically, attackers generate a large number of packets or requests, which ultimately overload the target system. An attacker uses multiple hacked or controlled sources to carry out a «distributed denial of service» (DDoS) attack. There are six classes of DoS attacks: Back, Land, Neptune, Pod, Smurf, Teardrop.

U2R (User-to-Root) attacks involve obtaining the privileges of a local superuser (network administrator) by a registered user. There are four classes of U2R attacks: Buffer_overflow, Loadmodule, Perl, Rootkit. R2L (Remote-to-Local) attacks are characterized by an unregistered user gaining access to a computer from a remote computer. There are eight classes of R2L Attacks: Ftp_write, Guess_password, Imap, Multihop, Phf, Spy, Warezclient, Warezmaster. Probe attacks are about scanning network ports for confidential information. There are four classes of Probe attacks: IPsweep, Nmap, Portsweep, Satan.

Specific types of network attacks are presented in the KDDCup database [20], which contains about 5,000,000 connection records. A connection is a sequence of TCP packets over a limited period of time, the start and end points of which are clearly defined and during which data is transmitted from the sender's IP address to the receiver's IP address using a specific protocol.

As an architectural solution of the attack detection module, five neural networks of the multilayer perceptron type are proposed [25]: NN1 - to determine the category of network attack (DoS, R2L, U2R, Probe) or the fact that there was no attack; NN2… NN5 - to identify the network attack class, if any (each of these four neural networks corresponds to the attack category and is able to identify classes that belong only to this category). In fig. 1 shows the structure of a hypothetical complex that uses such a solution.

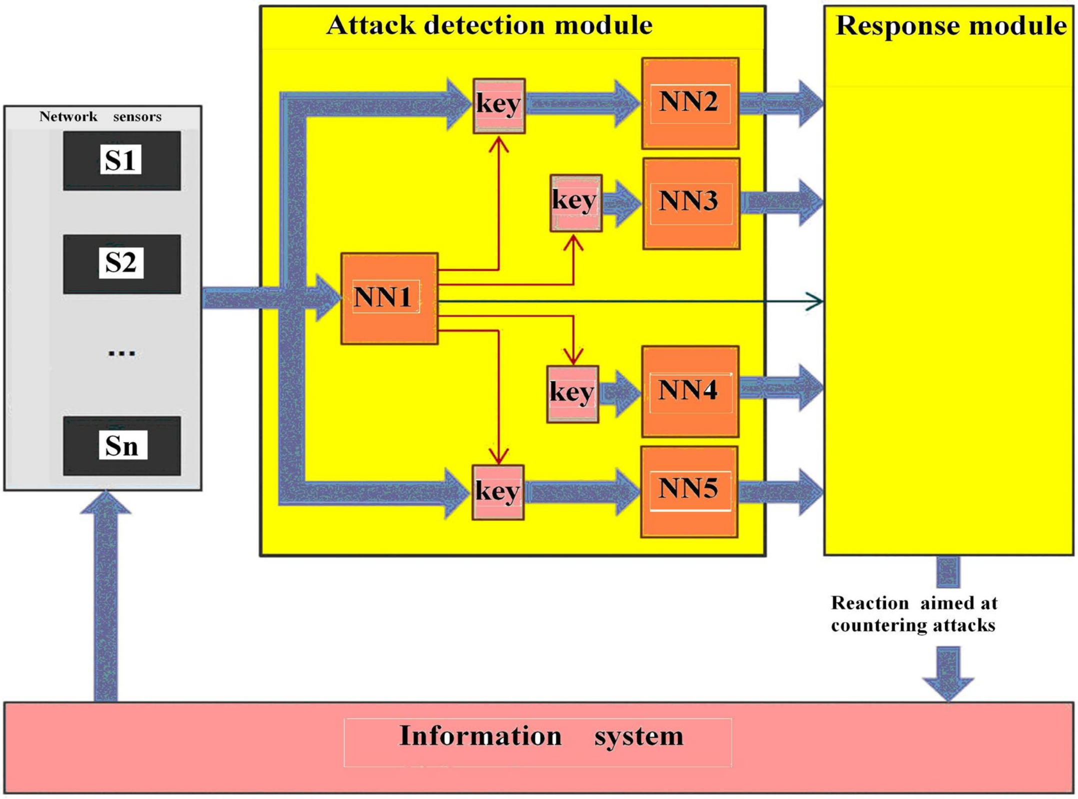


Figure 1 – Structure of hypothetical complex [25]

The complex includes a network attack detection module, which receives connection data from network sensors and outputs the result to the response module. The signal from NN1, which detects the category of network attack, through the key turns on one of the neural networks NN2… NN5, which will determine the class of network attack according to the category. The simulation results on other neural networks (attack detection accuracy) are summarized in table 1.

Table 1 – NN simulation results [25]

NN	NN1	NN2	NN3	NN4	NN5
Configuration	41–1–132–5	41–1–160–5	41–1–111–5	41–1–8–5	41–1–107–5
Accuracy, %	91.03	98.93	94.77	–	97.35

The table shows that the best result is achieved when determining the type of attacks of the DoS and Probe classes, slightly worse – for the R2L class. For the U2R class, it was not possible to configure the NN4 neural network to obtain acceptable results. This is due to the small number of records (52 in total) in the KDD Cup 99 database that belong to the U2R class.

8.2. Research of the type of neural network to determine attacks

8.2.1. Multilayer perceptron

The author's certificate [10] is received on a technique of detection of threats to a computer network by means of a multilayer neural network. As means of realization use of various neuropackages or drawing up of program model is possible [2-3, 8-11, 24-25]. So, for example in [3], on the created software model it is established that the best indicators of the quality of detection of network attacks of the DoS category in multilayer receptors with one hidden layer of neurons. In fig.2 shows that the best indicators have NN with the following activation functions: logistic function on the hidden layer; Softmax function on the resulting layer.

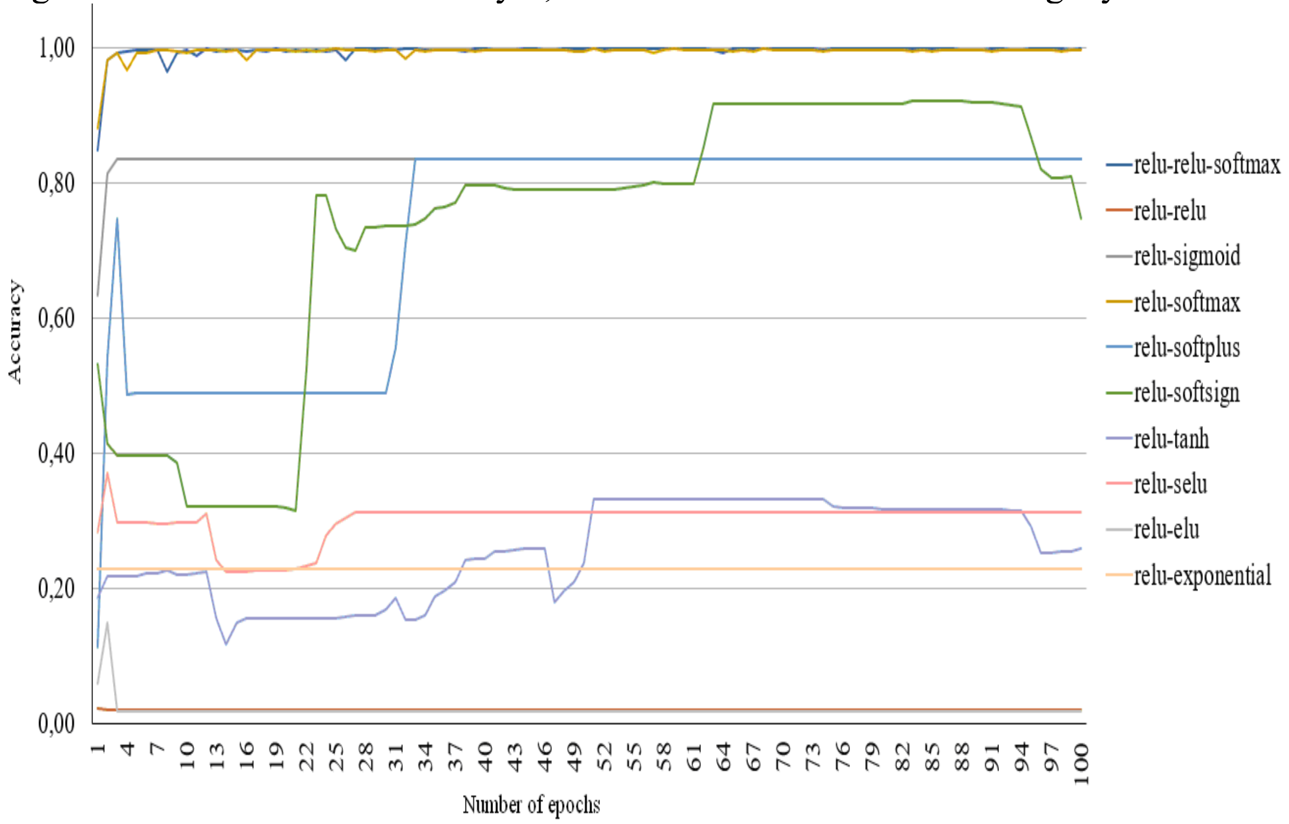


Figure 2 – Investigation of MLP-DOS accuracy from the number of learning epochs by different activation functions [3]

The study of the accuracy of MLP-DoS from the number of learning epochs for different numbers of hidden neurons is shown in fig. 3. The figure shows that the best quality indicators showed NN configuration 29-1-25-6. The obtained structure of MLP-DoS, where y0 corresponds to Back, y1 – Land, y2 – Neptune, y3 – Pod, y4 – Smurf, y5 – Teardrop.

In fig. 4 presents a study of MSLE from the number of learning epochs for the obtained structure of MLP-DoS. Based on low MSLE scores, it is concluded that the best optimization method for MLP-DOS is the adadelta method. To achieve a sufficient level of quality of the algorithm is enough 25 epochs of learning.

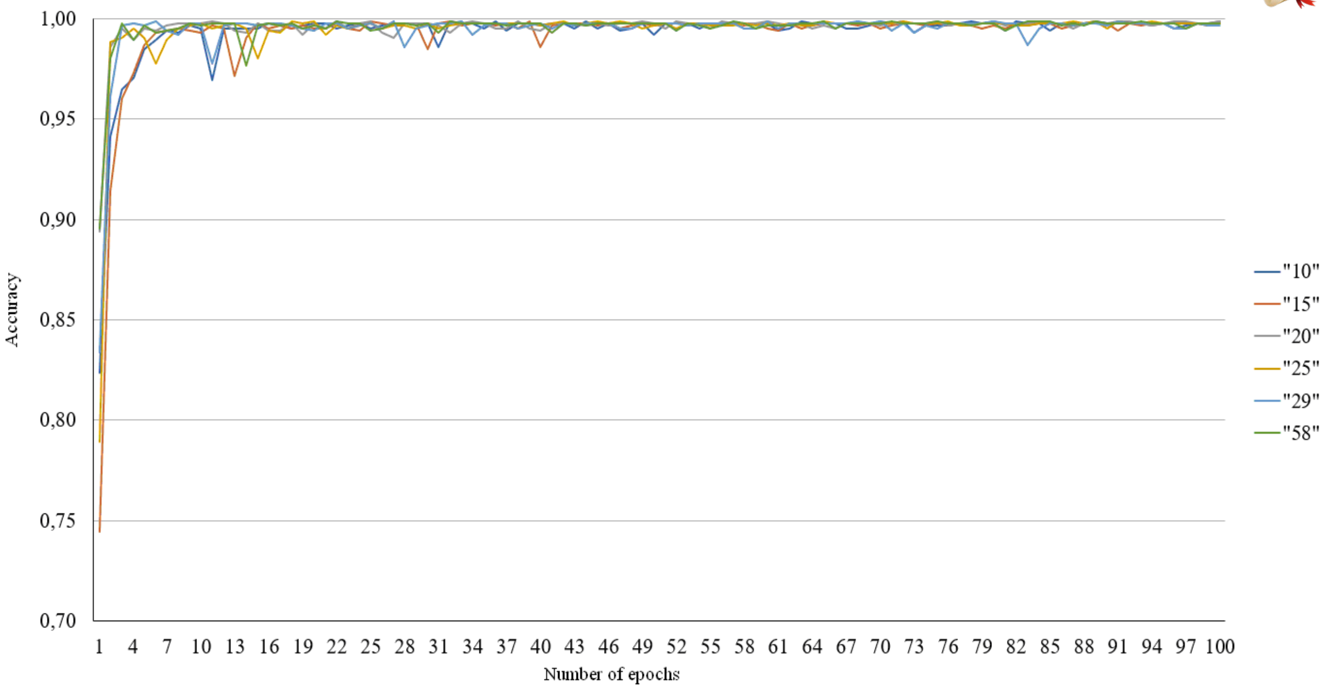


Figure 3 – Study of the accuracy of MLP-DoS from the number of learning epochs for different numbers of hidden neurons [3]

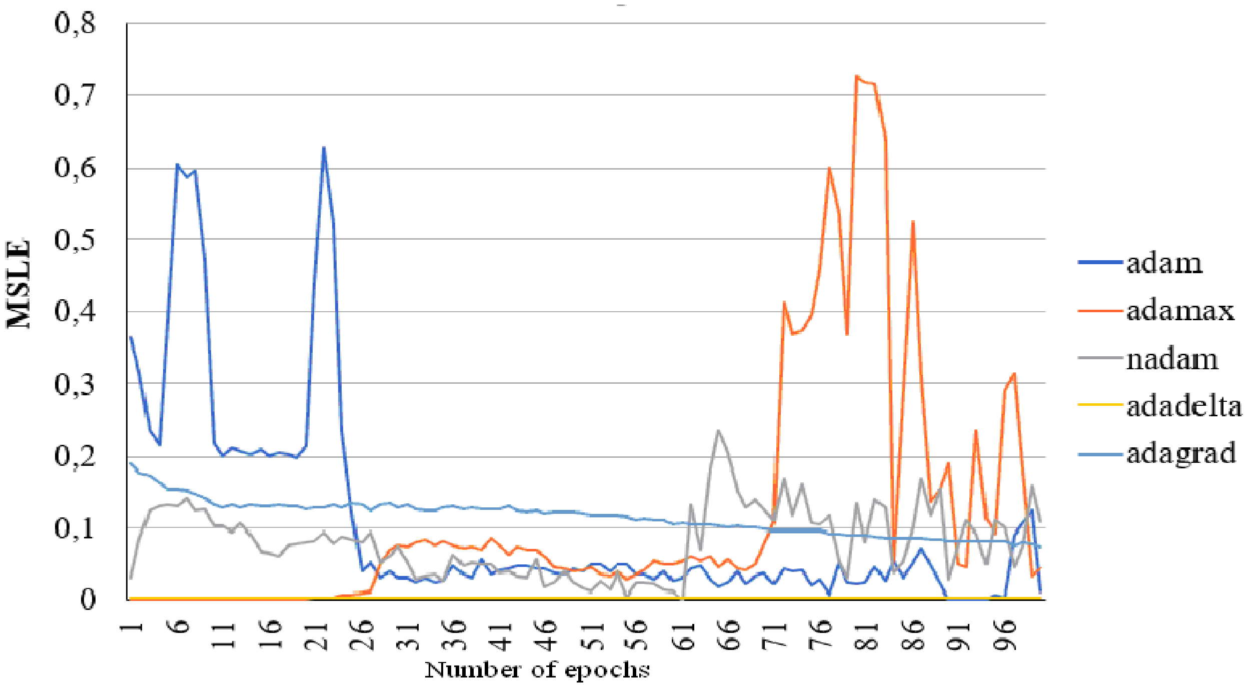


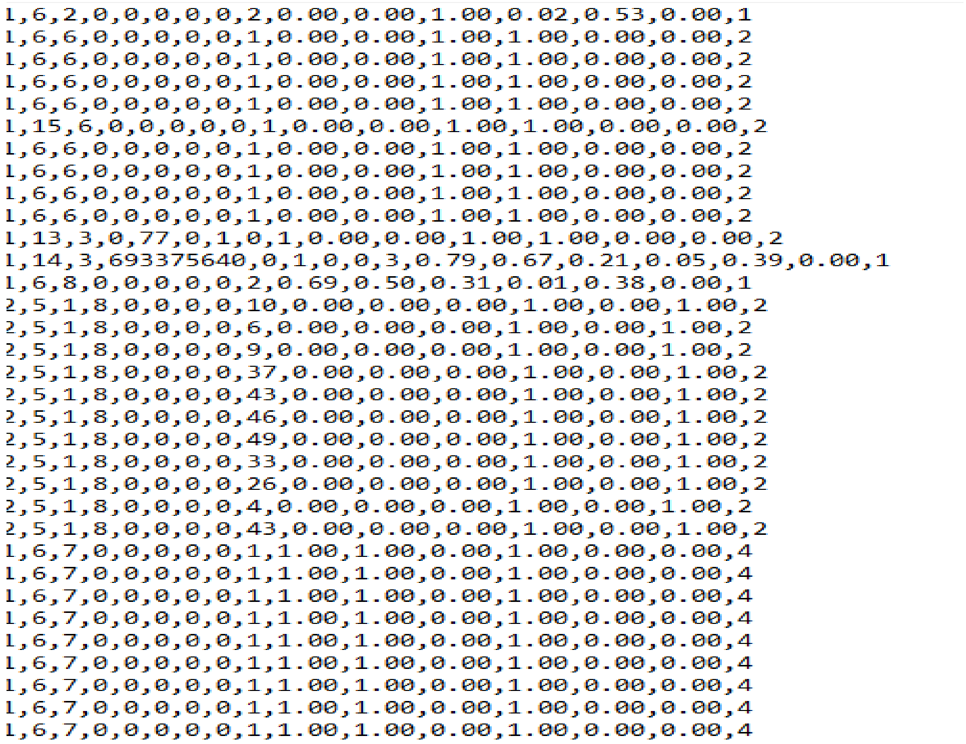
Figure 4 – MSLE study of the number of learning epochs by different methods of optimizing MLP-DoS learning [3]

8.2.2. Self-organizing map

The Kohonen network (self-organizing map) has only two layers: an input and an output composed of radial neurons of an ordered structure (the output layer is also called a topological map layer or «screen»). The neurons of the source layer are located in the nodes of a two-dimensional grid with rectangular or hexagonal cells. The number of neurons in the grid determines the degree of detail of the result of the algorithm, and, ultimately, the accuracy of the generalizing ability of the map depends on it. Thus, having a map in front of us and knowing information about some of the objects under study, we can fairly reliably judge the objects with which we are little familiar. The SOM learning process includes the following stages: assigning weights as small random numbers; choice of learning speed parameter; determining the best matching unit; weight update.

In [2, 7] created software models «SOM» on C++ and Python respectively. For example, in [7] to determine the classes of network attacks of the Probe category, a Kohonen network was created, consisting of 15 input neurons (according to the selected parameters) and 900 results, which are presented in the form of a two-dimensional map with 30 columns and 30 rows. The software implementation is made in Python. MiniSom (implementation of a self-organizing map based on Numpy) was used as the main framework for SOM.

The Matplotlib library is selected for information display, which includes the Matplotlib.pyplot module, which contains functions for graphical information display. Numpy (open source library) is selected as auxiliary libraries, which includes multidimensional arrays with high-level functions for working on them. Metrics from Sklearn were used for data analysis. The Sklearn.metrics module includes evaluation functions and performance metrics. A sample of 400 records was used to teach SOM, a fragment of which is shown in fig. 5.



```
1,6,2,0,0,0,0,0,2,0.00,0.00,1.00,0.02,0.53,0.00,1
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,15,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,6,6,0,0,0,0,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,13,3,0,77,0,1,0,1,0.00,0.00,1.00,1.00,0.00,0.00,2
1,14,3,693375640,0,1,0,0,3,0.79,0.67,0.21,0.05,0.39,0.00,1
1,6,8,0,0,0,0,0,2,0.69,0.50,0.31,0.01,0.38,0.00,1
2,5,1,8,0,0,0,0,10,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,6,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,9,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,37,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,43,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,46,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,49,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,33,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,26,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,4,0.00,0.00,0.00,1.00,0.00,1.00,2
2,5,1,8,0,0,0,0,43,0.00,0.00,0.00,1.00,0.00,1.00,2
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
1,6,7,0,0,0,0,0,1,1.00,1.00,0.00,1.00,0.00,0.00,4
```

Figure 5 – Fragment of the training sample

For testing, the sample consisted of 205 records, the control sample - of 60 records. The KDDCup database was used as the data source. As a result of the neural network, a map was obtained (fig. 6), each figure of which reflects a certain type of attack: Portsweep (red); IPsweep (green); Satan (blue); Nmap (yellow).

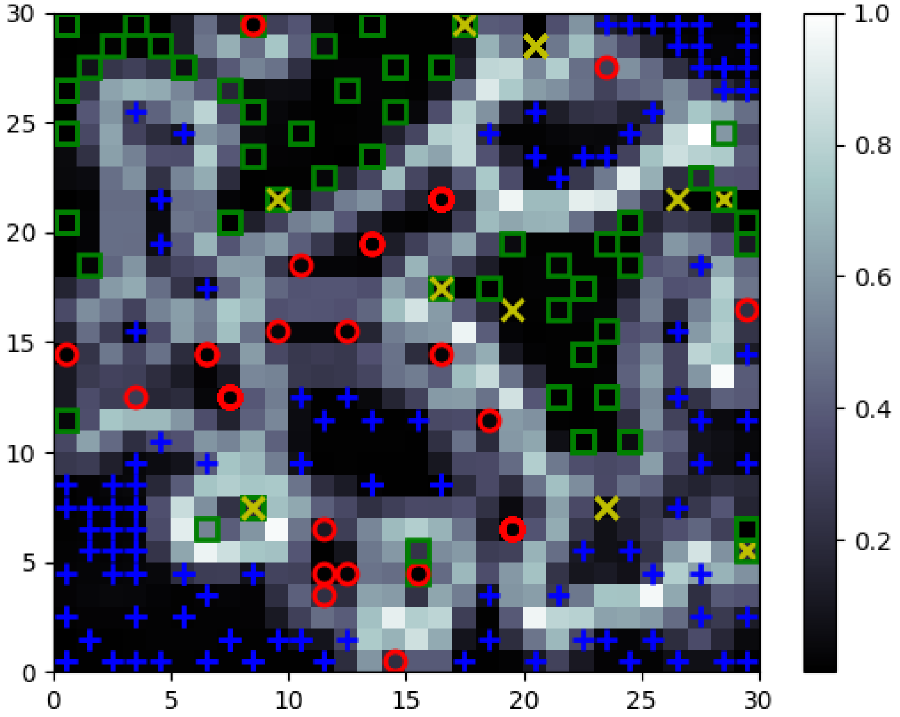
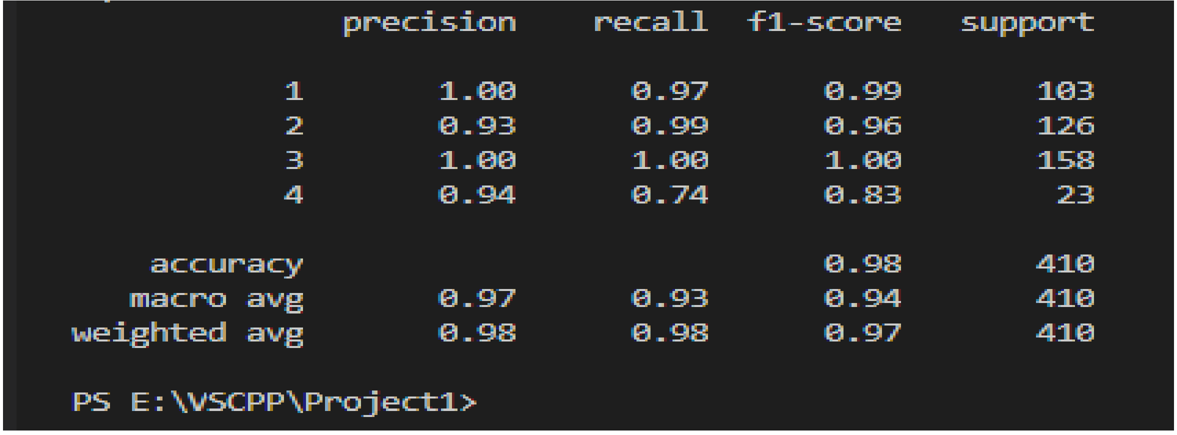


Figure 6 – Distribution of attacks on the map

In fig. 7 presents the results obtained on the software model.



```
              precision    recall  f1-score   support

           1       1.00      0.97      0.99       103
           2       0.93      0.99      0.96       126
           3       1.00      1.00      1.00       158
           4       0.94      0.74      0.83        23

    accuracy                           0.98       410
   macro avg       0.97      0.93      0.94       410
weighted avg       0.98      0.98      0.97       410

PS E:\VSCPP\Project1>
```

Figure 7 – The results obtained on the program

The number of epochs is 250 thousand. The average error was 0.05 and suggests that the neural network well determines the classes of network attacks. The neural network showed the lowest accuracy in determining the attack class Nmap. That is, NN detects network attacks of the Portsweep, IPsweep, and Satan classes well, but errors can occur when defining the Nmap class. To solve this problem, it is recommended to increase the concentration of Nmap class network attacks in the training sample. The results of the research were presented at conferences [7].

8.2.3. Neural fuzzy network

With the help of MatLAB Fuzzy Logic Toolbox ANFIS, a fuzzy network of configuration 4-5-8-16-1 was created (fig. 8) [4], where 4 is the number of neurons in the input layer, which corresponds to the categories of DoS, U2R, R2L and Probe attacks; 5 – the total number of layers of the fuzzy network (input, inputmf, rule, outputmf, output); 8 – the number of neurons in the first hidden layer (inputmf), which depends on the number of input variables and the number of terms (was the attacks or not); 16 – the number of neurons in the second hidden layer by the number of rules (rule); 1 – the number of neurons in the resulting layer (output).

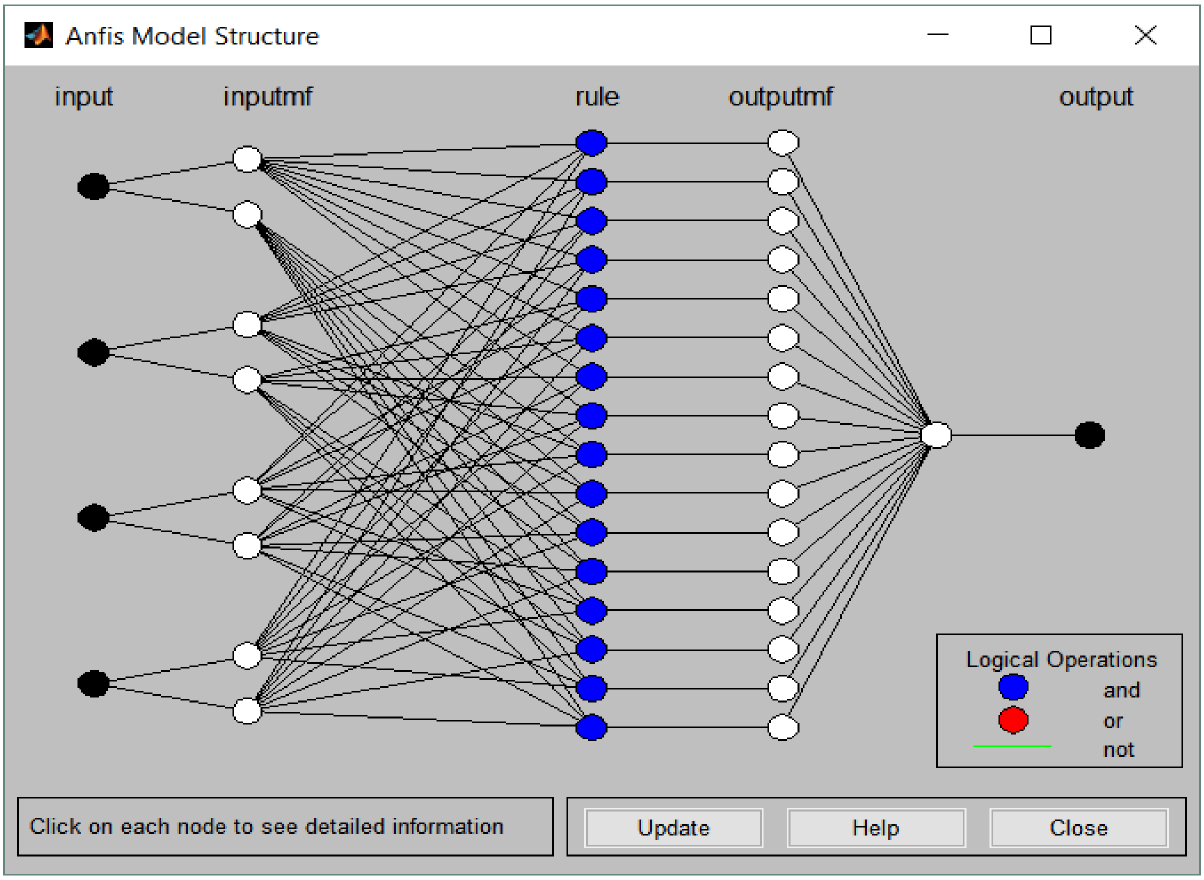


Figure 8 – Structure of the ANFIS system

The input layer contains neurons that represent the membership functions of the input fuzzy variables and perform the operation of falsification (fuzzy) of the input data. The inputmf layer consists $2 \cdot 4 = 8$ values for the rules that make up the knowledge base created as a result of model learning, these neurons can contain any implementation of the operation of the t-norm, which is a fuzzy analogue of the operation «AND» (logical operation «AND»). The third layer is the TSK generator (this is a parametric layer). The neurons of the rule layer contain the results of rule calculations taking into account the weight of each rule: $2^4 = 16$. The neurons of the outputmf layer contain the final results of rule calculations, which are grouped into fuzzy classes. The output layer is normalizing (nonparametric) and contains only one neuron, which calculates the final output of the model by performing a dephasification operation (definition) by determining the centers of fuzzy classes. At the preparatory stage, samples were prepared using the Excel package. On the created neural network the research of value of an error of training on various functions of belonging at various algorithms is carried out.

8.2.4. Hybrid attack detection: combining different classifiers

The resulting vector is found as the sum of the output vectors of the three NN: MLP, SOM and ANFIS. If the coordinate in the resulting vector is zero or one, then there are no attacks. The unit means that a false positive occurred on one of the NN. If the coordinate value of the vector is 2 or 3, then an attack of the corresponding type has occurred. The implementation of such a hybrid option for the definition of attacks: the combination of basic classifiers will eliminate the shortcomings in their operation separately.

3. Research of one-level and two-level approaches to detecting network attacks

8.3.1. Determination of neural configurations

In [22] provides an overview of existing datasets, the most common of which is the NSL-KDD database, initiated by the US Department of Defense's Advanced Research Projects Authority (DARPA) based on the KDD'99 database [21]. The data set consists of the following sets [15]: KDDTest+, KDDTrain+, KDDTrain+20% (table 2). The benchmark contains 43 parameters for each record, 41 of which relate to the traffic itself, and the last two: Label and Score. Although all attacks exist in the database, but their distribution is highly distorted, as shown in table. 1. More than half of the records that exist in each data set reflect the normal state, and the standards for the classes U2R and R2L are extremely small. This is an accurate idea of the distribution of modern attacks on Internet traffic, where the most common attacks are DoS, and attacks of U2R and R2L are almost non-existent.

Table 2 – Distribution of attacks in the NSL-KDD database

Dataset	Total	Normal	DoS	Probe	U2R	R2L
KDDTrain+20%	25192	13449 (53 %)	9234 (37 %)	2289 (9,16 %)	11 (0,04 %)	209 (0,8 %)
KDDTrain+	125973	67343 (53 %)	45927 (37 %)	11656 (9,11 %)	52 (0,04 %)	995 (0,85 %)
KDDTest+	22544	9711 (43 %)	7458 (33 %)	2421 (11 %)	200 (0,9 %)	2654 (12,1 %)

Multilayer NN as a mathematical apparatus. The number of neurons in the latent layer of Multilayer NN can be determined by a known formula, which is a consequence of the Kolmogorov-Arnold-Hecht-Nielsen theorem:

$$\frac{N_y Q}{1+\log_2(Q)} \le N_w \le N_y\left(\frac{Q}{N_x}+1\right)(N_x+N_y+1)+N_y ,$$

where N_y is the length of the output signal; Q is the number of elements of the set of educational examples; N_w – the required number of synaptic connections; N_x – the dimension of the input signal.

After estimating the required number of synaptic connections, you can calculate the required number of neurons in the hidden layer (N):

$$N = \frac{N_w}{N_x + N_y}.$$

As an example in fig. 9 shows the configuration of NN2 at the maximum number of hidden neurons: 41-1-8-4, where 41 is the number of initial neurons; 1 – the number of hidden layers; 8 – the number of hidden neurons; 4 – the number of resulting neurons ($Y_1 = 1$, then the category DoS; $Y_2 = 1$, then the category Probe; $Y_3 = 1$, then the category U2R; $Y_4 = 1$, then the category R2L; otherwise, where *i*=1,…, 4).

For all NN, the calculation of the minimum ($N_{\min}$), average (N_{avg}) and maximum ($N_{\max}$) number of hidden neurons was performed (table 3).

Table 3 – NN parameters according to the Levenberg-Marquardt algorithm [9]

NN	Appointment NN	Number hidden neurons			Optimal NN configuration	Number training standards	MSE
		$N_{\min}$	N_{avg}	$N_{\max}$			
NN1	Attack detection	33	313	593	41-1-33-23	7000	2,71
NN2	Detect the attack category	8	56	104	41-1-8-4	500	3,79
NN2-1	Detect the attack class of the DoS category	16	138	259	41-1-16-9	500	3,74
NN2-2	Detect the attack class of the Probe category	12	84	156	41-1-12-6	400	4,79
NN2-3	Detect the U2R category attack class	4	28	52	41-1-4-2	300	1,47
NN2-4	Detection of an attack class of category R2L	10	70	130	41-1-10-5	400	1,98

The results of the study of one-level and two-level approaches to identifying network attacks are presented in [9]. The author's certificate was obtained for the methodology of the corresponding research [11].

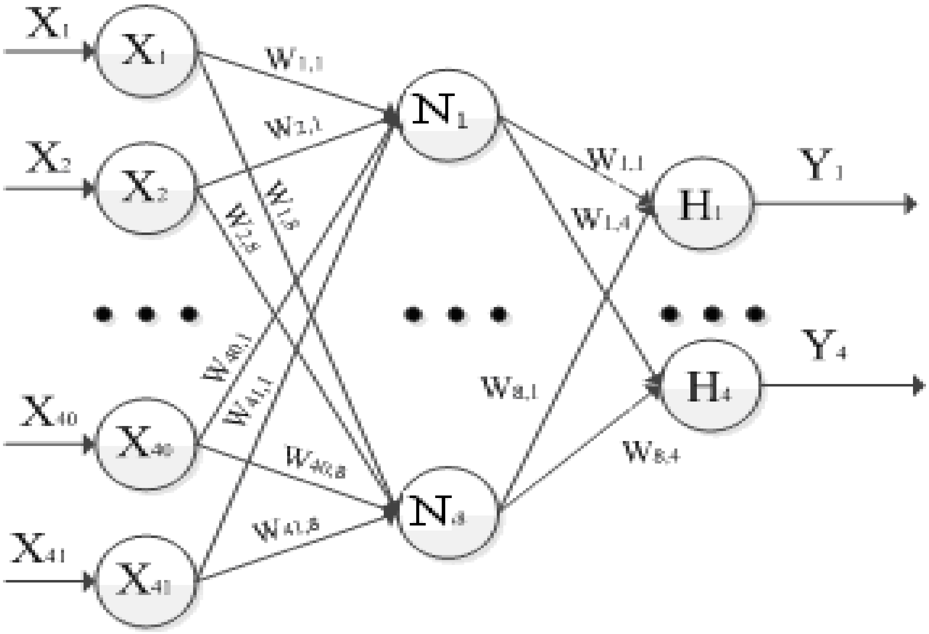


Figure 9 – NN2 configuration 41-1-8-4 [9]

8.3.2. Investigation of neural network error and learning time

On the created NN with the of the Neural Network Toolbox package of the MatLAB program the research of value of MSE from length of educational sample on training algorithms is carried out: Levenberg-Marquardt; Bayesian Regularization; Scaled Conjugate Gradient with different number of hidden neurons ($N_{\min}$, N_{avg}, $N_{\max}$).

The Levenberg-Marquardt algorithm always gives the smallest value of an error of train the received relts are added to table 2. The table shows that in all cases it is sufficient to use a minimum number of hidden neurons. For example, for NN2, the lowest value of MSE is achieved in a sample of 500 training standards.

Configuration 41-1-33-23 with a sample length of 7000 examples (first approach) was taken as NN1. To implement the second approach, the following configurations were selected: 41-1-8-4 (NN2, 500 references); 41-1-16-9 (NN2-1, 500 references); 41-1-12-6 (NN2-2, 400 references); 41-1-4-2 (NN2-3, 300 references); 41-1-10-5 (NN2-4, 400 references). Three parallel series of four possible combinations of threat detection based on the NN ensemble are considered. The average detection of network threats on NN1 (according to the first approach) is 2.21 s, according to the second approach – 0.92 s. That is, using an ensemble of five NN is about 2.4 times faster.

8.3.3. Investigation of errors of the first and second kind

The first kind of error is the number of incorrectly detected attacks (FP, False Positive). The error of the first kind on NN1 is 7.39 %, and when using an ensemble of neural networks: NN2; NN2-1; NN2-2; NN2-3; NN2-4 (second approach) – 2.17 %. The second type of error is the number of omissions of attacks (FN, False Negative). The error of the second kind on NN1 is 6.96 %, and when using the ensemble NN (second approach) – 3.91 %. During the experiments, the following results were obtained on NN (fig. 10): TP (True Positive); FP (False Positive); FN (False Negative); TN (True Negative), on the basis of which at the final stage it remains to assess the quality of solution for different approaches (table 4), using the following quality indicators.

TP 1950	**FP** 170
FN 160	**TN** 20

1st approach

TP 2150	**FP** 50
FN 90	**TN** 10

2nd approach

Figure 10 – The results of the study [9]

1) TPR – the rate of correct detection of network attacks is determined by the known formula:

$$TPR = \frac{TP}{TP + FN},$$

where TP is the number of correctly recognized anomalous connections;

FN – the number of errors of the second kind (the number of skips of attacks).

2) FPR – the rate of false positives is defined as:

$$FPR = \frac{FP}{FP + TN},$$

where FP – the number of errors of the first kind (the number of incorrectly detected attacks);

TN is the number of correctly recognized normal connections.

3) CCR – the correctness of the classification of connections is determined by the following formula:

$$CCR = \frac{CC}{TP + TN + FN + FP},$$

where CC is the total number of elements whose class has been correctly defined on a combined data set composed of normal and anomalous connections.

4) ICR – the indicator of incorrect classification is defined as:

$$ICR = \frac{IC}{TP + TN + FN + FP},$$

where IC is the number of cases of incorrect classification.

5) GPR – an indicator of generalizing ability when detecting:

$$GPR = \frac{\overline{TP*}}{\overline{TP*} + \overline{FN*}},$$

where $\overline{TP*}$, $\overline{FN*}$ – represent respectively the number of correctly recognized anomalous connections and the number of errors of the second kind, which are calculated on the unique data of the control set, strictly excluding any data of the training set.

6) OPR – retraining rate when detecting:

$$OPR = \overline{TPR} - GPR,$$

where $\overline{TPR}$ is the indicator of the correctness of detection on the unique data of the training set.

7) GCR – an indicator of the generalizing ability in the classification:

$$GCR = \frac{\overline{CC*}}{\overline{TP*} + \overline{FN*} + \overline{FP*} + \overline{TN*}},$$

where $\overline{CC*}$, $\overline{FP*}$, $\overline{TN*}$ – represent, respectively, the number of correctly classified connections, the number of errors of the first kind and the number of correctly recognized normal connections, which are calculated on the unique data of the control set, strictly excluding any data of the training set.

8) OSR – retraining rate in the classification:

$$OCR = \overline{CCR} - GCR,$$

where $\overline{CCR}$ – an indicator of the correctness of the classification on the unique data of the training set.

Table 4 – Indicators for assessing the quality of solutions for different approaches [9]

Indicator	TP	FP	FN	TN	TPR	FPR	Accuracy	Precision	Recall
1st approach	1950	170	160	20	0,92	0,89	0,86	0,92	0,92
2nd approach	2150	50	90	10	0,96	0,83	0,94	0,98	0,96

The table shows that the best results are achieved based on the use of the NM ensemble (second approach): TPR (indicator of the correctness of the definition of network attacks) is 0.96 (in comparison with 0.92); FPR (false positive rate) – 0.83 (in comparison with 0.89); accuracy – 0.94 (in comparison with 0.86), accuracy (Precision) – 0.98 (in comparison with 0.92) and completeness (Recall) – 0.96 (in comparison with 0.92) compared to the first approach on based on NN1.

8.4. Combined option for detecting attacks: a combination of neural network technology with immunology

8.4.1. Initial data for SOM

According to the source [6], 29 parameters characterizing network connections are enough to detect and classify 9 out of 22 types of attacks. Table 5 shows the relationship between the parameters in the implementation and the attributes of KDDCup [20].

The mechanism of clonally selection simulates the behavior of B-cells in the process of immune response to an antigenic stimulus. This interaction is accompanied by the production of B-cell clones, which can undergo mutations of varying degrees depending on the strength of their binding (affinity) to any epitope of the antigen. Like the negative selection algorithm, the clonally selection algorithm belongs to a family of population algorithms in which individuals who describe the current solution of the problem are improved and replaced, and fight with each other for the right to be selected as the best candidates [1].

Table 5 – The ratio of parameters and attributes of KDDCup

Parameter	Attribute	Parameter	Attribute
x0	Duraction	x15	Srv rerror rate
x1	Protocol type	x16	Same srv rate
x2	Service	x17	Diff srv rate
x3	Flag	x18	Srv diff host rate
x4	Source bytes	x19	Dst host count
x5	Destination bytes	x20	Dst host srv count
x6	Land	x21	Dst host same srv rate
x7	Wrong fragment	x22	Dst host diff srv rate
x8	Urgent	x23	Dst same src port rate
x9	Hot	x24	Dst host srv diff host rate
x10	Count	x25	Dst host rerror rate
x11	Srv count	x26	Dst host srvb serror rate
x12	Serror rate	x27	Dst host rerror rate
x13	Srv serror rate	x28	Dst host srv rerror rate
x14	Rerror rate		

The first implementation of the clonally selection algorithm is CLONALG [16], which consists of initialization and population cycle: affinity; selection; cloning; mutation; choosing the best clone.

8.4.2. Sampling based on the clonally selection algorithm

The following approach is used to solve the problem of classifying network attacks: generation based on existing data with the addition of new data using a clonally selection algorithm.

The model of the clonally selection algorithm can be represented as follows [16]: CLONALG = $(Pr^0, f, L, N, n, \beta, d, \varepsilon)$,
where Pr^0 – initial antibody population;
f is the objective function;
L is the length of the antibody receptor;
N is the number of antibodies in the population;
n is the number of antibodies selected for cloning (with the highest affinity);
β – reproductive factor that regulates the number of clones of selected antibodies;
d is the number of antibodies to be replaced by new ones (with the lowest affinity);
ε – algorithm stop criterion.

Step 1. Initialization. Generation of the initial population of Pr^0 antibodies.

Step 2. Determination of affinity. For each $Pr_j \in Pr^t$ antibody, calculate the value of the objective function $y_i=f(Pr_j)$ and determine the affinity $g_j=affinity(y_j)$, $j \in \{1..N\}$, t – the generation number.

Step 3. Selection. Select a subset of antibodies with the highest affinity $Pr_{\{n\}}$.

Step 4. Cloning. Obtain a population of $C_{\{Nc\}}$ clones from $Pr_{\{n\}}$.

Step 5. Hypermutation. Obtain a population of altered clones $C^{*}_{\{Nc\}}$ with $C_{\{Nc\}}$.

Step 6. Determining the affinity of the population of altered clones. For each antibody $C^{*}_{j} \in C^{*}_{\{Nc\}}$, calculate the value of the objective function $y_j=f(C^{*}_{j})$ and determine the affinity $g_j=affinity(y_j)$, $j \in \{1..N_C\}$.

Step 7. Selection. Select a subset $C^{*}_{\{n\}}$ of n antibodies with the highest affinity from the population of altered clones $C^{*}_{\{Nc\}}$.

Step 8. Replacement. Replace the subset $Pr_{\{n\}}$ with $C^{*}_{\{n\}}$.

Step 9. Clonally removal. Replace the subset of $\mathrm{Pr}_{\{d\}}$ antibodies with the lowest affinity with new individuals.

Step 10. Check the stop condition of the algorithm. According to the selected criterion ε, check the fulfillment of the stop condition of the algorithm.

8.4.3. Research results

We compare the obtained values for the initial sample of 369 vectors and obtained by clonally selection of the sample of 937 vectors. TPR, FPR, CCR and ICR were calculated for the initial and obtained samples. By increasing the sample with clonally selection, we thus improve the algorithm of intrusion recognition. In addition, it significantly improves the recognition of little-known attacks, the sample of which is very few elements. False Positive (FP) is the number of incorrectly detected attacks when the normal state was taken as an attack. False Negative (FN) is the number of skips of attacks when the attack was mistaken for a normal network state. The percentage values of errors of the first and second kind at different stages are summarized in table 6.

Table 6 – The obtained values of the errors of the first and second kind

Number test data	Error first kind, %	Error second kind, %
362	0	12,98
1009	2,58	4,36
1986	1,964	5,04
4698	3,13	5,58
9931	1,96	5,49

The table shows that when the sample was increased (from 1009 to 9931 examples) approximately 9 times the errors of the first (number of incorrectly detected attacks) and the second kind decreased approximately 1.3 and 0.8 times, respectively. The results of the research were presented at conferences [2].

Conclusions

In according with the first task, a study of the type of NN to detect networks attacks. It is determined that MPL, SOM and ANFIS are suitable for finding solution (setting the network attack categories) based on the use of open databases KDDCup or NSL-KDD, but not enough to determine U2R. In addition, a hybrid option for defining attacks has been proposed: combining different classifiers (MLP, SOM and ANFIS), which will eliminate the shortcomings in their functioning separately.

In according with the second task, the detection of attacks was carried out using two approaches: a one-tier approach based on NN1 and a two-tier approach based on a complex of five NN (NN2; NN2-1; NN2-2; NN2-3; NN2-4) to determine DoS, Probe, U2R or R2L attack categories (at the first level) and determining the attack class according to the category (at the second level). Normalized data from the open NSL-KDD database were used for all NN. The best results are achieved on the basis of the use of the NN complex (the second approach): the indicator of the correctness of the definition of network attacks is 0.96 (in comparison with 0.92); the rate of false positives – 0.83 (in comparison with 0.89); reliability – 0.94 (in comparison with 0.86), accuracy – 0.98 (in comparison with 0.92) and completeness – 0.96 (in comparison with 0.92) compared to the first approach based on NN1.

3**. *In accordance with the third task***, studies of the combined option for the definition of network attacks, which combines the use of neural network technology with immunology. For example, when detecting a network attack category when increasing the sample (from 1009 to 9931 examples) for SOM by the clonally selection algorithm, the errors of the first kind (number of incorrectly detected attacks) and the second kind (number of skips of attacks) decreased approximately in 1.3 and 0.8 times, respectively. It should be noted that the error of the second kind when increasing the training data (according to the algorithm of clonally selection) was not more than 5 % in comparison with 10 % (without the use of clonally selection), which is better twice.