FACTA UNIVERSITATIS
Series: **Electronics and Energetics** Vol. 35, Nº 2, June 2022, pp. 269-282
https://doi.org/10.2298/FUEE2202269P

Original scientific paper

WK-FNN DESIGN FOR DETECTION OF ANOMALIES IN THE COMPUTER NETWORK TRAFFIC

Danijela Protić[1], Miomir Stanković[2], Vladimir Antić[3]

[1]Center for Applied Mathematics and Electronics, Belgrade, Serbia
[2]Mathematical Institute of SASA, Belgrade, Serbia
[3]Center for Applied Mathematics and Electronics, Belgrade, Serbia

Abstract. *Anomaly-based intrusion detection systems identify abnormal computer network traffic based on deviations from the derived statistical model that describes the normal network behavior. The basic problem with anomaly detection is deciding what is considered normal. Supervised machine learning can be viewed as binary classification, since models are trained and tested on a data set containing a binary label to detect anomalies. Weighted k-Nearest Neighbor and Feedforward Neural Network are high-precision classifiers for decision-making. However, their decisions sometimes differ. In this paper, we present a WK-FNN hybrid model for the detection of the opposite decisions. It is shown that results can be improved with the xor bitwise operation. The sum of the binary "ones" is used to decide whether additional alerts are activated or not.*

Key words: *WK-FNN, anomaly detection, weighted k-nearest neighbor, feedforward neural network*

1. INTRODUCTION

Due to the enormous increase in computer applications in the last few decades, the need for protection of the computer networks has multiplied [1]. Intrusion detection systems (IDSs) are the main defense of the network infrastructure, used to detect attacks or to indicate anomalies in the behavior of the computer network. The signature or misuse IDSs proactively detect the presence of known maliciousness. The most practical method to detect signature of malicious content is to measure the similarity between detected pattern of current network activity and the already known patterns of various types of malicious attacks [2]. The anomaly detection is performed by detecting changes in system behavior or usage patterns [3]. The identification of anomalies in the network is essential to diagnose

Received October 11, 2021; received in revised form December 6, 2021
Corresponding author: Danijela Protić
Center for Applied Mathematics and Electronics, Belgrade, Serbia
E-mail: adanijela@ptt.rs

attacks or failures that seriously affect the performance and security of the computer network [4, 5].

The goal of an anomaly-based IDS is to proactively detect any activity or an event on a host computer or network that shows a deviation from a normal network behavior [2]. In order to provide suitable solution for the detection of anomalies in the computer network, the concept of normality is fundamental. The idea of normality is usually introduced through a formal model that expresses the relationship between the variables involved in the dynamics of the system, so that an event is recognized as abnormal when its degree of deviation in relation to the profile or the behavior of the system, specified by the normality model, is high enough [6].

In the last few decades, machine learning has started to play an important role in anomaly detection [6, 7, 8]. In supervised machine learning, anomaly detection can be thought of as a kind of binary classification, since the data sets for training and testing the models contain binary labels: one for normal observations and one for abnormal observations. It should be noted that the troubleshooting data set can be quite unbalanced in detecting anomalies. Therefore, it is important to use some data transformation algorithms prior to supervised learning. In this article we propose a three-step algorithm that removes all irrelevant features from the Kyoto 2006+ dataset and normalizes the instances so that the influence of one feature cannot dominate the others. After pre-processing is completed, there were nine features left to train two binary classifiers, namely the weighted k-Nearest Neighbor (wk-NN) and the Feedforward Neural Network (FNN). The classifiers show a high precision in decision making but, in some cases their decisions are different. The proposed WK-FNN hybrid model recognizes the opposite decisions based on a bitwise exclusive or (xor) operation between the outputs of the classifiers. The binary sum of the opposing decisions is used as the basis for the additional warnings. Two alerts are combined. Trigger alert reacts to the opposite decisions and threshold-based alert allows users to prioritize alerts that are rated as critical.

2. LITERATURE REVIEW

Since the nature of the features and the number of instances determine the applicability of anomaly detection techniques, the analysis of the high-dimensional data sets becomes a challenge for researchers [9, 10]. In the last few decades, researchers have investigated the intrusion detection systems for various purposes and on the different datasets. In [11] and [12] the authors compare the DARPA98, KDD CUP ’99, NSL-KDD, Kyoto 2006+ and CAIDA datasets. In addition, the authors in [13] have compared a signature-based and anomaly-based classification and examined the ISCX2012, CIC-IDS-2017 and CSE-CIC-2018 datasets in the context of the feature selection and the attack types. In [14] the authors describe the functionality of the ADFA-LF and ADFA-WD datasets and compare them with the DARPA98, KDD Cup ’99, NSL-KDD and CIC-IDS-2017 datasets. The datasets are simulated or captured from real computer network traffic, and differ in size, number of features, purpose, type of attacks, etc. The main characteristics of the above datasets are summarized in the Table 1.

Table 1 Description of the datasets

Dataset	Type of the attacks	Features	Kind of traffic	Description
ADFA-LD and ADFA-WF	Hydra-FTP, Hydra SSH, Adduser, Java-Meterpreter, Meterpreter, Webshell.	26	From the host for normal activities, with user behavior ranging from web browsing to LATEX document preparation.	Created from the evaluation of the system-call-based HIDS; Linux and Unix OS (LD) and Windows (WF).
AWID	Attacks on 802.11 (authentication request, probe request, injection, ARP flooding).	156 features extracted from each packet	Emulated (small network, 11 clients)	WLAN traffic in packet-based format; 37 million packets in one hour captured.
CAIDA	DDoS	Network traffic traces	Real (collected on high-speed monitors)	Collected on commercial backbone link from 2008 to 2019; Does not contain diversity of attacks.
CIC-IDS-2017	Botnets, cross-site-scripting, DoS, DDoS, Goldeneye, Hulk, RUDY, Slowhttptest, Slowloris.	More than 80	Emulated (small network)	Captured over a period of 5 days; contains network traffic in packet-based and bidirectional flow-based format.
CSE-CIC-2018	Brute force, Hearthbleed, Botnet, DoS, DDoS, Web attacks, infiltration from the network inside.	More than 80	Emulated (simulated scenarios)	10 days network traffic and log files of 50 machines from the attacker side and 420 PCs and 30 servers from the victim organization.
DARPA98	DoS, privilege escalation (R2L and U2R), probing.	41	Emulated (small network)	7 weeks of network traffic in packet-based format and audit log.
ISCX 2012	Scenarios: Infiltrating the network from the inside, HTTP DoS, DDoS using an IRT bootnet, SSH brute force attack.	20	Emulated (small network)	7 days of packet network traffic observed.
KDD Cup ‘99	DoS, privilege escalation (R2L and U2R), probing.	42	Emulated (small network)	Derived from the DARPA98 dataset. Five weeks of network traffic in packet-based format
Kyoto 2006+	Attacks against honeypots (DoS, exploits, malware, port scans, shellcode).	24	Real (honeypots, and regular servers)	3 years of real packet-based network traffic; packets converted into the sessions.
NSL-KDD	DoS, privilege escalation (R2L and U2R), probing.	42	Emulated (small network)	Derived from the KDD-Cup ’99 dataset; does not contain redundant records in the training set nor duplicates in the test set.

As it is shown in Table 1, all datasets, with the exception of the Kyoto 2006+ dataset, are either simulated network data or come from actual network traffic, which is mainly

used for signature detection. The dataset is also the only one intended for anomaly-based IDS modelling. For these reasons, this study uses the Kyoto 2006+ dataset as the basis for binary classification experiments with machine learning (ML) models. Machine learning is effective in eliminating redundant and irrelevant data, increasing learning accuracy and improving comprehensibility of the results [15]. Feature selection has direct influence on the efficiency of the results and offers a way to reduce computation time, improve accuracy, and enable a better understanding of the classification models or the data. In the case of an anomaly detection, the labels assigned to the data instances are usually in the form of binary values [16]. Machine learning models can be very effective in learning normal or abnormal patterns from training data and in detection of the anomalies in the computer networks [17].

The Kyoto 2006+ dataset is captured and created in actual network traffic to classify network traffic as normal or abnormal. Since the purpose of this work is to present the hybrid classifier for improved anomaly detection in binary classification this data set is used in experiments. The Kyoto 2006+ dataset is unbalanced data set in which the amounts of normal and abnormal data are unbalanced.

In [18], the authors present a series of tests they carried out to assess the effectiveness of ML techniques in detecting anomalies and present the algorithms that gave the best results. In [19] the authors carried out experiments with 10 daily records from the Kyoto 2006+ dataset and showed that accuracy decreases slightly when the number of features is reduced from 17 to 9 and the instances range from -1 to 1. In supervised machine learning, wk-NN has the highest accuracy of a variety of machine learning models. In [20] the author proposes a method that can detect large-scale attacks in real time with weighted k-NN classifiers. The key factor in developing an anomaly-based intrusion detection system is the selection of significant features for decision-making. A good feature selection for choosing meaningful and as few features as possible plays a key role in successful anomaly-based IDS. In [21] the authors proposed a new learning algorithm for pseudo-neighbor elimination and anomaly detection based on the wk-NN model in order to minimize the effects of these distant neighbors.

In [22] the authors examine the applicability of the feedforward architecture of neural networks for traffic prediction and compare the performance of different back-propagation algorithms. The prediction is made for various random aggregates of traffic flows. The performance analysis showed the effectiveness of the proposed method for an adequate choice of the learning algorithm. In [23] the authors approached an IDS using a 2-layered feedforward neural network. In the training phase, the early-stop strategy is used to overcome the problem of overfitting in neural networks. The proposed system is assessed against the DARPA dataset. The selected connections from the DARPA dataset are preprocessed and feature range is converted into [-1, 1]. These modifications affect final detection results in particular. In [24] the authors proposed IDS model, which uses the feedforward neural network and the back-propagation algorithms along with various optimization techniques to minimize the overall computational overhead, while maintaining a high level of performance. The experimental results on the benchmark NSL-KDD dataset shows that in some cases the accuracy of the proposed IDS model is better than that of the other IDS models. Because of its high performance and low computational requirements, the proposed model was a suitable candidate for real-time implementation. In [25] the authors showed the results on the accuracy of two FNN classifiers in the short processing time when deciding on anomalies in the behavior of the complex computer networks. In [26] the authors used a PC-generated offline data set to assess the performance of two neural network-based techniques. In this data set, each

data point corresponds to a normal or anomaly class. It is assumed that the anomaly data is the intruder data, obtained by disabling some PC controllers, audio drivers, graphics drivers, etc. In this article, the authors took 15 randomly selected features from the log file, which contains 20,000 records. The authors have shown that the FNN classifiers are approximately 98% accurate.

Hybrid models for anomaly detection are also the topic of various research. In scenario given in [27], the authors propose a hybrid online-offline system in which the offline model maintains the general properties of the network traffic, based on Radius Nearest Neighbor while the online model based on the support vector machine continuously learns and they work together to detect anomalies. The method is evaluated using the NSL-KDD 2009 dataset. This model achieved an accuracy of ~95% with known anomalies. It should be noted that the NSL-KDD dataset is the simulation of the computer network traffic on the middle-size American military base [11]. In order to improve the detection performance and to reduce the tendency to frequent attacks, the two-stage hybrid method based on binary classification and k-NN technique is proposed in [28]. First, binary classifiers and an aggregation module are used to efficiently identify the exact classes of network connections. Afterwards, the connections whose classes insecure, further determine their classes by the k-NN algorithm. The second step is built on the results of the first step and is a useful addition to the first step. By combining the two steps, the proposed method achieves reliable results in the NSL-KDD data set [11].

Network alerts are a critical aspect of network performance monitoring because they are designed to provide information technology (IT) administrators with quick insight into the network problems. Therefore, network alerting should be an important consideration for those choosing their network alerting tools. In [29] the author provides information on the four main types of network alerts. Real-time alerts periodically or continuously scan all areas of the network for network behavior problems. The time between each network pass is an important consideration as it determines how quickly network problems are identified. Intelligent alerts provide details about the problem, when and where it occurred, and which areas of the network are affected. Flexible delivery alerts are network monitoring notification tools that can be configured for scheduled and hourly alerts to ensure alerts are received at the right time. Critical and tiered alerts are tools for minimizing the number of network notifications. Network monitoring alerts, also known as threshold alerts, are tools that support critical and tiered alerts, so that the user can prioritize alerts that are critical or violate a preconfigured network configuration. Systems with tiered alerting assign problems to one of several categories. Alerts are processed according to the importance of the category. In [30] the authors confirm that the alert ranking classifies alerts according to the dangerousness of the alert. The alarm tactic requires that the functionality responsible for the alarm classification should not be computationally expensive, otherwise the advantages of the quick response, which is obtained by a prioritized reaction to dangerous alerts, are negated. In [31] the authors explain that not all classification algorithms equally accurate. Therefore, it is important to carefully select the criteria that can accurately classify the alerts based on the specific security needs of an organization. In [32] the authors describe the efficiency of the basic methods for rule-based alert classification and explain that engineers usually concentrate primarily on critical alerts, but not on errors and warnings. They claim that engineers should investigate more alerts. At the same time, they find a lot of time is wasted in investing in non-serious warnings (low precision), but many serious alerts are still lost. In [33] the authors divide alerts into low- and high-level alerts and point

out that high-level alert management is a potential task that helps the administrator to analyze alerts correctly and to allocate time and effort.

3. DATA COLLECTION

The Kyoto 2006+ dataset is publicly available and a widely used dataset in network-based intrusion detection research. The dataset includes more than three years of actual traffic data collected from honeypots (Solaris 8 for Intel, Windows XP (no patch, SP2, fully patched), Nepenthes, others), darknet sensors, and other systems (mail server to collect various types of mails, web crawler developed by NTT Information Sharing Platform Laboratories, Windows XP to evaluate malware activities) deployed on five different computer networks inside and outside the University of Kyoto [34]. The Kyoto 2006+ dataset is developed through deploying of honeypots in the network, but does not describe any details the types of attack [13]. In addition, the IDS Bro has been used to convert packet-based traffic into a format called sessions. IDS Bro is a signature and behavior-based analysis framework that provides detailed data on hypertext transfer protocol (http), domain name system (DNS), secure shell (SSH) communication protocol and strange network behavior [35]. Thanks to its analysis engine, it is suitable for high performance network monitoring, protocol analysis, and real-time application layer status information. The Bro event engine is responsible for receiving the internet protocol (IP) packets and converting them into events forwarded to the policy script interpreter, which then produces an output [36].

During the observation period (from 2006 to 2009) more than 50 million sessions with normal traffic, 43 million sessions with known attacks and 425 thousand sessions with unknown attacks were recorded. Each session includes 24 features, 14 out of which characterize statistical features derived from the KDD ’99 Cup dataset and 10 additional flow-based features (IP addresses, ports, and duration) [11, 37]. A feature Label indicates the presence of attacks [38]. In the original data set, there were three labels: 1 for normal sessions, -1 for known attacks, and -2 for unknown attacks. However, since unknown attacks are very rare in the dataset (~0.7%), we assigned the same label to known and unknown attacks (-1), which leads to binary classification [39].

The main problem associated with the Kyoto 2006+ is its size. In this study, this problem is solved with the pre-processing algorithm, which removes all irrelevant features (categorical features, statistical features regarding to the connection duration, and features for further analyses) and normalizes instances of the relevant features with a hyperbolic tangent function to the range [-1,1]. After the pre-processing is completed, features 5-13 remain for the evaluation of the models, and the feature Label identifies the session as normal or abnormal [19, 40]. Table 2 shows the description of the features used in the experiments.

In this research, the notation of the instances is as follows: the number of instances in a daily record is referred to as the total number of instances, the number of instances labelled with 1 is referred to as the number of normal instances, while the number of instances labelled with -1 denotes the number of anomalous instances.

Table 2 Description of the features from the Kyoto 2006+ dataset

Feature	Description
Count	The numbers of connections whose source IP address and destination IP address are the same to those of the current connection in the past two seconds.
Same_srv_rate	% of connections to the same service in the Count feature.
Serror_rate	% of connections that have 'SYN' errors in Count feature.
Srv_error_rate	% of connections that have 'SYN' errors in Srv_count (% of connections whose service type is the same to that of the current connections in the past two seconds) features.
Dst_host_count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose source IP address is also the same to that of the current connection.
Dst_host_srv_count	Among the past 100 connections whose destination IP address is the same to that of the current connection, the number of connections whose service type is also the same to that of the current connection.
Dst_host_same_src_port_rate	% of connections whose source port is the same to that of the current connection in Dst_host_count feature.
Dst_host_serror_rate	% of connections that have 'SYN' errors in Dst_host_count feature.
Dst_host_srv_serror_rate	% of connections that have 'SYN' errors in Dst_host_srv_count feature.
Label	Indicates whether the session was attack or not; '1' means normal. '-1' means known attack was observed in the session, and '-2' means unknown attack was observed in the session.

4. WK-FNN MODEL

A classification model generally maps the input data to a specific target and determines which label to assign to the new, unlabeled data. With binary classification, a classifier assigns the input data into one of two classes. The WK-FNN hybrid model is based on two binary classifiers. The wk-NN classifier is a lazy learner who saves training data and labels, and waits for the test data. Instead of focusing on building a general model, it works on storing instances of the training data into classes. The FNN, an eager learner, creates a classification model based on the training data set before it is received data for prediction.

The basic idea of the wk-NN is to expand k-Nearest Neighbor (k-NN) algorithm which stores all instances corresponding to the training data in n-dimensional space. Predictions for a new instance x are made by searching the entire training set for the k closest neighbors and summarizing the output variable for these cases. The classification is based on calculation of a simple majority vote of each point. wk-NN extends the k-NN such that instances of the training set, which are particularly close to the new instance, have more weight in the decision than those who are more distant. The main idea is to make the distant neighbor less effective than the closest, at making decisions by majority vote, by giving more weight to the nearest point and less to the more distant [41, 42]. To do this, the distances $d_w(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{p}(x_i - y_i)^2}$ are converted into the weights. The simplest conversion

function is inverse of the distance. The closest k points are weighted with weights $w = \frac{1}{d_w(x,y)^2}$ (the weight decreases with increasing the distance).

The FNN consists of a series of layers with highly connected neurons in each layer, with the final layer producing the outputs that relate the inputs to the desired output, so that

$$y_i(\mathbf{w}, \mathbf{W}) = F_i\left(\sum_{j=1}^{q} W_{ij} f_j\left(\sum_{l=1}^{m} w_{ji} x_l + w_{j0}\right) + W_{f0}\right) \quad (1)$$

where f_j and F_i denote hidden and output layer transfer functions, m represents the number of inputs x_l, q represents the number of outputs y_i, $\mathbf{w}$ and $\mathbf{W}$ are weight matrices, and w_{j0} and W_{f0} are biases [43]. The FNN is trained through an iterative process to modify the weights so that the given inputs map an appropriate response. In this way, the inputs are classified according to the target classes. In general, FNNs have a large number of parameters which, due to the convergence to a correct set of parameter values, can lead to the estimation problems [44]. For this reason, the weights are updated according to the Levenberg-Marquardt (LM) algorithm [45, 46].

The design of the WK-FNN model is based on the wk-NN and FNN binary classifiers, which work in parallel and decide on the anomaly in the behavior of the computer network. The basic idea is to train wk-NN and FNN with the same training set and evaluate high-precision classifiers (Figure 1). Subsequently, the classification of the unknown network transfer is carried out by both classifiers. The decisions about the anomaly are transmitted to the xor block, where the result of the counter-decision is calculated. Finally, the percentage of the opposite decision triggers an alert.

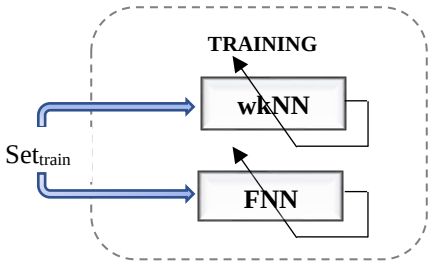

Fig. 1 Classifiers' training

The WK-FNN model is a three-layer structure. The first layer classifies the network traffic according to the both wk-NN and FNN. A bit-by-bit xor operation is carried out in the second module. The third part of the WK-FNN marks the opposite decisions (Fig. 2).

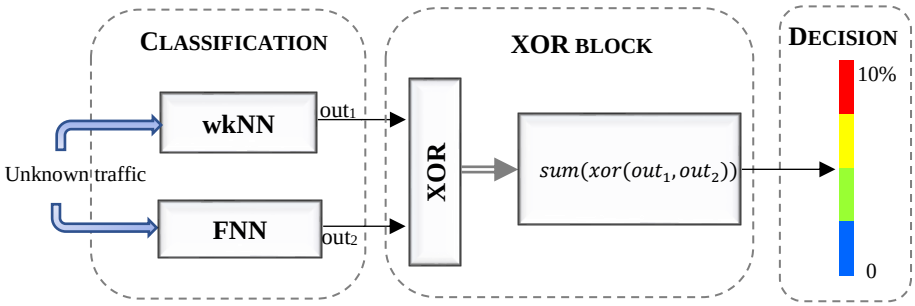

Fig. 2 WK-FNN model

Through the classification, both classifiers decide about the unknow network traffic and the outputs of each of the classifiers (decision about normal network behavior or the anomaly) are then passed on to the XOR block, where the 'exclusive or' bitwise operation is performed. The different/opposite decisions are recognized by performing the xor logical operation on the classification results, which is logically true (1) if one of the outputs is, but not both, non-zero. Otherwise the result is logical false (0).

The sum of the different decision in the decision block is then calculated as follows

$$sum_{out} = \sum_{k=1}^{length(dataset)} xor(out_{1k}, out_{2k}) \quad (2)$$

where out_{1k} and out_{2k} represent the k-th results of the classification, and $out_k = xor(out_{1k}, out_{2k})$. The result is then passed on to the decision-making engine. The opposite decisions indicated by bit-by-bit xor operation can generate different types of alerts, depending on the organizational structure and information security requirements such as confidentiality, integrity and data availability. The alerts can be sent to the network administrator or to the other IDS. It should be noted that the additional anomaly alerts are separate from regular IT alerts. Therefore, it is necessary to define the anomaly alert promotion rule in order to generate an IT alert based on the anomaly alerts.

The promotion rule of the WK-FNN model is based on the ratio between the number of opposing decisions and the total number of decisions of the classifiers, expressed as a percentage. A decision is presented based on a linear scale threshold. The basic idea behind the decision is that the number of contradicting decisions is low if the two classifiers are really highly accurate. Otherwise, the results will not be reliable. Instead of making additional decisions about what is normal or abnormal, the decision block points out the difference in the classifiers' decisions. If both choose 'normal' or 'anomaly' their decisions are not different. Otherwise, their decisions will differ, and the result on xor operation will be binary one. As 'normal' traffic refers to binary 1 (Label equals 1) and 'anomaly' refers to the binary 0 (Label equals to -1), the number of opposing decisions refers to the sum of all decision, because the same decisions result in zero after performing the xor operation. The number of opposing decisions sum_{out} is given in the Eq. 2. Divided by the total number of the decisions, given as $length(dataset)$, the resulting value shows the percentage of the opposing decisions.

For this experiment, we have divided the priority levels of the alert scale into five different categories: negligible/insignificant alerts (whitelisted), potential threats (they have no direct influence on network traffic and the network structure), warnings (provide information about the risks), silent alarms (critical with ticketing) and high priority alarms (signal an attack). The scale is linear and divided into five groups with two percentage ranges, from 0 to 10 %. It should be noted that the scale can be chosen differently, depending on the needs of the organization security.

5. Experimental Results

The experiments are carried out on three days of the computer network traffic recorded at Kyoto University computer network in February 2007. All models are selected based on

processing time and memory usage and are simulated in the MATLAB Classification Learner platform for Windows 64-bit OS installed on an Intel Core i7 processor with 2.7GHz CPU and 16 GB RAM memory. The wk-NN model is trained by approximating an instance by the weighted sum of 10 k-nearest neighbors. Weights are calculated based on inverse distances. The FNN with one hidden layer, nine inputs, nine nodes in the hidden layer and one output node is trained with the LM algorithm. In order for the LM algorithm to work correctly, the activation function of the hyperbolic tangent is used for each node, since it is differentiable, centered around 0, and its output range is [-1, 1]. The weights are initialized to the small random numbers, because the optimization begins as a gradient descent (GD) algorithm, which speeds up the convergence of the LM algorithm and minimizes the wrong approximations [20]. The WK-FNN model design is tested as follows:

- Each of the three daily sets, consisting of 57278, 57279 and 58317 instances, is divided into the two subsets: Set_1 of 75% of the instances is used to train and test the classifiers, while Set_2 of 25% of the instances is used for the WK-FNN tests;
- Set_1 is then divided into two groups of instances: 70% are used to train the classifiers and the remaining instances are used to test both the models;
- Set_2 is used for testing the opposite decisions of the classifiers – the results are passed on to the xor module;
- The sum of all contradicting/opposing decisions is sent to the alarm detector.

In summary, 52.5% and 22.5% of instances of each daily record are used to train and test the classifiers, respectively, and 25% of the instances are used to verify the WK-FNN model. The performance of the classifiers is calculated in term of accuracy (ACC). ACC represents the ratio between the number of correctly classified instances to the total number of instances, given as follows [47]

$$\mathrm{ACC} = \frac{TP+TN}{TP+TN+FP+FN} \tag{3}$$

A true positive (*TP*) result indicates that the anomaly has been correctly identified as "anomaly". A true negative (*TN*) means that the IDS has correctly classified the normal behavior as "normal". A false positive (*FP*) indicates a misclassification of the normal behavior of the network as an "anomaly". A false negative (*FN*) indicates abnormal behavior of the network that has been mistakenly assigned to the "normal" class. The accuracy results for the classifiers and the number of opposing decisions recognized by the WK-FNN model are shown in Table 3. The opposing decisions are calculated for 25% of the instances from each daily record.

Table 3 Accuracy of the classifiers and the number of opposing decisions

Instances	Accuracy [%]		Opposing decisions [%]	Opposing decisions [instances]
	FNN	wk-NN		
57278	99.3	99.5	8.08	1157
57279	99.3	99.3	3.18	456
58317	99.0	99.1	0.67	98

In Table 3, the Opposing decisions [instances] = sum_{out} (every binary 1 triggers an alert), and the ratio of the number of opposing decisions and the total number of decisions (anomaly score) is given with the Opposing decisions [%] = $\frac{sum_{out}}{length(dataset)} \cdot 100\%$. It can be

seen that there is no relationship between number of instances in the daily set and the Opposing decisions. The anomaly score ranges from 0 to 10 % and is used as the threshold value for the additional alert. There are some concerns about the priorities and the percentages associated with the conflicting decision. A higher percentage of the different decisions indicates the greatest uncertainty in the classification and the incomplete knowledge of the event, which is not related only to the decision of the classifiers [48]. In general, uncertainty in decisions can arise from the following sources: (1) data errors (uncertainties about past events), (2) forecast errors (uncertainties about future events) and (3) model (residual) errors (differences between what is observed and what the model shows). The WK-FNN supports resolving the uncertainty by calculating the percentage of the opposing decisions of the classifiers, but cannot determine the probability of a certain event occurring. It is designed to provide the warning for the conflicting decisions of the classifiers. Then the decision makers, knowing all the possible versions of the resolved issues use this auto-generated alert to resolve information security related issues in their organization. The alert scale presented in this paper was chosen to indicate the low probability of serious effects on network security with a small percentage of similar decisions and the high probability of an attack on the computer network with a high percentage of opposing decisions. It should be noted that there are other options for selecting the decision criteria, the threshold value ranges, and the alert colors, which can be modified depending on the additional protection requirements and the sensitivity of the information to the potential threats. In [49] Multicriteria decision-making (MCDM) is presented. The authors examined the changes in the measurement scale and the formulation of criteria. In [50] the authors proposed the evaluation metrics to measure the effectiveness of collaborative decisions based on the likelihood of trust in collaborative decision-making processes. In [51] the author proposes a prioritization of alerts, which can be achieved by integrating several methods. In the experiments presented in this paper, the percentages of the opposing decisions are divided into five alert groups (See Table 4).

Table 4 Linear threshold scale and ranges of the opposing decisions

Threshold range [%]	Alert grouping and colouring	Opposing decisions [%]		
		0.67	3.18	8.08
0-1.99	Negligible alert (Black)	*		
2-3.99	Potential threat (Blue)		*	
4-5.99	Warning (Green)			
6-7.99	Silent alarm (Orange)			
≥ 8.00	High priority alarm (Red)			*

Although the simple manual rules cannot always adequately capture the complex and interactive patterns of factors that influence the priority of the alerts the manual rules proposed here were used for classification of the five different priority levels. Generally, the levels can be divided into three main categories: (1) errors/failures (negligible alert, potential threat), (2) warnings and (3) critical level (silent alarm, high-priority alarm) [32]. A negligible alert means that the alert is whitelisted (the lowest probability of serious effects on network security and the smallest percentage of similar decisions). The administrator can exclude certain activities which generate alerts, based on the user analytics. For the purposes of this research, the percentage of the negligible alert is used to

be less than 2%. Potential threat means that the alert may result from network disturbances and has no negative impact on business. The warning displays the known alert sources, acts as an information aggregator, provides information about the risks, and generates a hazard message. The silent alarm signals the high-level discrepancy in the decision of the classifiers and causes the ticket to be issued (the proof of authentication or authorization must be verified). A high priority alarm signals the highest probability of the attack (the highest percentage of opposing decisions). The ranking list can be adopted after a few other factors relating to the dataset (label, total number of instances) and metrics (accuracy, precision, recall), and depending on the changes to the system, new types of the alerts can be added [32]. The ranking can be combined with methods that reduce or reclassify a given list of rankings [52].

6. CONCLUSION

This article introduces the design of a WK-FNN hybrid model that warns of opposing decisions about anomalies in the computer network. The model consists of a classification module, an XOR block and a decision-making engine. In the classification module two high-precision binary classifiers work in parallel. The classifiers take into account 9 features with the normalized instances to decide whether the network traffic is abnormal or not. The results of the decisions of the classifiers are passed on to the XOR block, where the exclusive or binary operation is carried out. Binary 1 triggers an additional anomaly alert which is sorted into one of the predefined alert groups. The results show the presence of additional alerts related to the negligible alert, potential threat, and high-priority alarm.

Acknowledgement: *A part of this research is presented at the 21st International Arab Conference on Information Technology, 6th of October 2020, Giza, Egypt.*

REFERENCES

[1] D. Protic, "Neural cryptography," *Military Technical Courier*, vol. 64, no. 2, pp. 483–492, 2016.
[2] J. Sen and S. Methab "Machine Learning Applications in Misuse and Anomaly Detection," 2009. Available https://arxiv.org/ftp/arxiv/papers/2009/2009.06709.pdf.
[3] D. Dasgupta and H. Brian, "Mobile security agents for the network traffic analysis," In Proceedings of the DARPA Information Survivability Conference and Exposition II DISCEX01, 2001, vol. 2, pp. 332–340.
[4] A. Kind, M. P. Stoecklin and X. Dimitropoulos, "Histogram-based traffic anomaly detection," *IEEE Transactions on Network and Service Management*, vol. 6, no. 2, pp. 110–121, June 2009.
[5] P. Čisar and S. Marvić Čisar, "EWMA statistics and fuzzy logic in function of network anomaly detection," *Facta Universitatis, Series: Electronics and Energetics*, vol. 32, no. 2, pp. 249–265, June 2019.
[6] M. H. Bhuyan, D. K. Bhattacharyya and J. K. Kalita, "Network Anomaly Detection: Methods Systems and Tools," *IEEE Communication Surveys & Tutorials*, vol. 16, no. 1, pp. 303–336, First quarter 2014.
[7] V. Hodge and J. Austin, "A survey on outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.
[8] T. Nguyen and G. Armitage, "A Survey of Techniques for Internet Traffic Classification using Machine Learning," *IEEE Commun. Surveys Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.
[9] S. Omar, A. Ngadi and H. H. Jebur, "Machine Learning Techniques for Anomaly Detection: An Overview," *International Journal of Computer Applications*, vol. 79, no. 2, pp. 33–41, October 2013.
[10] C. Jie, L. Jiawei, W. Shulin and Y. Sheng, "Feature selection in machine learning: A new perspective," *Neurocomputing*, vol. 300, pp. 70–79, 26 July 2018.

[11] D. Protic, "Review of KDD CUP '99, NSL-KDD and Kyoto 2006+ Datasets," *Military Technical Courier*, vol. 66, no. 3, pp. 580–595, 2018.
[12] B. Bohara, J. Bhuyan, F. Wu and J. Ding, "A Survey on the Use of Data Clustering for Intrusion Detection System in Cybersecurity," *Int. J. Netw. Secur. Appl.*, vol. 12, no. 1, pp. 1–18, Jan 2020.
[13] A. Thakkar and R. Lohiya, "A Review of the Advancement int the Intrusion Detection Datasets," International Conference on Computational Intelligence and Data Science (ICCIDS 2019), *Procedia Computer Science*, vol. 167, pp. 636–645, 2020.
[14] A. Khraisat, I. Gondal, P. Vamplew and J. Kamruzzaman, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, pp. 2–20, 2019.
[15] S. Khalid, T. Khalil and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," In Proceedings of the 2014 Science and Information Conference, 2014, pp. 372–378.
[16] O. Osanaiye, O. Ogundile, F. Aina andA. Periola, "Feature selection for intrusion detection system in a cluster-based heterogeneous wireless sensor network," *Facta Universitatis, Series: Electronics and Energetics*, vol. 32, no. 2, pp. 315–330, June 2019.
[17] M. Bahrololum, E. Salahi and M. Khaleghi, "Machine Learning Techniques for Feature Reduction in Intrusion Detection Systems: A Comparison," In Proceedings of the 2009 Fourth International Conference on Computer Sciences and Convergence Information Technology, pp. 1091-1095, 2009.
[18] Y. -G. Cheong, K. Park, H. Kim, J. Kim and S. Hyun, "Machine Learning Based Intrusion Detection Systems for Class Imbalanced Datasets," *Journal of the Korea Institute of Information Security and Cryptology*, vol. 27, no. 6, 2017, pp. 1385–1395.
[19] D. Protic and M. Stankovic, "Detection of Anomalies in the Computer Network Behaviour," *European Journal of Engineering and Formal Sciences*, vol. 4, no. 1, pp. 7–13, 2020.
[20] Ming-Yang Su, "Real-time anomaly detection systems for Denial-of-Service attacks by weighted k-nearest neighbor classifier," *Expert Systems with Applications*, vol. 38, no. 4, pp. 3492–3498, April 2011.
[21] J. Dhar, A. Shukla, M. Kumar and P. Gupta, "A Weighted Mutual k-Nearest Neighbour for Classification Mining," arXiv.org. Submitted on 14 May 2020. https://arxiv.org/abs/2005.08640 [cs.LG].
[22] C. Callegari, S. Giordano and M. Pagano, "Neural network based anomaly detection," In Proceedings of the 2014 IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2014, pp. 310–314.
[23] F. Haddadi, S. Khanchi, M. Shetabi and V. Derhami, "Intrusion Detection and Attack Classification Using Feed-Forward Neural Network," In Proceedings of the 2010 Second International Conference on Computer and Network Technology, 2010, pp. 262–266.
[24] B. Subba, S. Biswas and S. Karmakar, "A Neural Network based system for Intrusion Detection and attack classification," In Proceedings of the 2016 Twenty Second National Conference on Communication (NCC), 2016, pp. 1–6.
[25] D. Protic and M. Stankovic, "A Hybrid Model for Anomaly-Based Intrusion Detection in Complex Computer Networks," In Proceedings of the 21st International Arab Conference on Information Technology, 6th of October 2020, Giza, Egypt, pp. 1–8.
[26] S. K. Gutam and H. Om, "Computational neural network regression model for host based intrusion detection system," *Perspectives in Science*, vol. 8, pp. 93–95, September 2016.
[27] M. Odiathevar, W. K. G. Seah and M. Frean, "A Hybrid Online Offline System for Network Anomaly Detection," In Proceedings of the 2019 28th International Conference on Computer Communications and Networks (ICCCN), 2019, pp. 1–9.
[28] L. Li, Y. Yu, S. Bai, Y. Hou and X. Chen, "An Effective Two-Step Intrusion Detection Approach Based on Binary Classification and k -NN," *IEEE Access*, vol. 6, pp. 12060–12073, 2018.
[29] J. Griffin, "All about network alerts + Best tools," by SolarWinds on October 29, 2020. Available https://logicalread.com/network-alerts/.
[30] F. Ullah and M. Ali Babar, "Architectural Tactics for Big Data Cybersecurity Analytic Systems: A Review," *The Journal of Systems and Software*, vol. 151, pp. 81–118, 2019.
[31] S. Allier et al., "A framework to compare alert ranking algorithms," In Proceedings of the Reverse Engineering (WCRE), 19th Working Conference on. IEEE, 2012.
[32] N. Zhao, P. Jin, L. Wang, X. Yang, R. Liu, W. Zhang, K. Sui and D. Pei, "Automatically and Adaptively Identifying Severe Alerts for Online Service Systems," In Proceedings of the INFOCOM, 2020.
[33] W. Alhakami, "Alerts Clustering for Intrusion Detection Systems: Overview and Machine Learning Perspectives," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, pp. 573–582, 2019.
[34] J. Song, H. Takakura, Y. Okabe, M. Eto, D. Inoue and K. Nakao, "Statistical Analysis of Honeypot Data and Building of Kyoto 2006+ Dataset for NIDS Evaluation," In Proceedings of the 1st Work-shop on

Building Anal. Datasets and Gathering Experience Returns for Security, Salzburg, April 10-13, 2011, pp. 29–36.
[35] K. Demertzis, "The Bro Intrusion Detection System", Project: Machine Learning to Cyber Security, 2018, DOI: 10.31140/RG.2.2.35333.40168.
[36] R. McCarthy, "Network analysis with the Bro security monitor," 2014, retrieved from https://www.admin-magazine.com/Archive/2014/24/Network-analysis-with-the-Bro-Network-Security-Monitor, 7 November 2021.
[37] KDD CUP '99 dataset. [Internet] http://kdd.ics.uci.edu/dataset/kddcup'99/kddcup'99.html, 2018.
[38] M. Ring, S. Wunderlich, D. Scheuring, D. Landes and A. Hotho, "A Survey of Network-based Intrusion Detection Data Sets, " arXiv:1903.02460v2 [cs.CR] 6 Jul 2019, pp. 1–17.
[39] Y.e Maleh, "Security and Privacy Management, Techniques, and Protocols," IGI Global, USA, 2018, pp. 266–267.
[40] D. Protic and M. Stankovic, "Anomaly-Based Intrusion Detection: Feature Selection and Normalization Instance to the Machine Learning Model Accuracy," *European Journal of Engineering and Formal Sciences*, vol. 1, no. 3, pp. 43–48, 2018.
[41] M. Zhao and J. Chen, "Improvement in comparission of weighted k nearest neighbor classifiers for model selection," *Journal of Software Engineering*, vol. 10, pp. 109–118, 2016.
[42] M. Faryaneh, "Weighted k-nearest neighbors (WKNN)," MATLAB Central File Exchange, https://www.mathworks.com/matlabcentral/fileexchange/74111-weighted-k-nearest-neighbors-wknn.
[43] W. F. Schmidt, M. A. Kraaijveld and R. P. W. Duin, "Feed forward neural networks with random weights," The Netherlands, Delft University of Technology, Faculty of Applied Phisics,1992, 0-8186-2915-0/92, IEEE, pp. 1–4.
[44] D. Protic, "Feedforward neural networks: The Levenberg-Marquardt optimization and the optimal brain surgeon pruning," *Military Technical Courier*, vol. 63, no. 3, pp. 11–28, 2015.
[45] K. Levenberg, "A method for the solution of certain problems in least squares," *Quarterly of Applied Mathematics*, vol. 5, pp. 164–168, 1944.
[46] D. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *SIAM Journal in Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.
[47] C. Ambedkar and V. K. Babu, "Detection of Probe Attacks Using Machine Learning Techniques," *International Journal of Research Studies in Computer Science and Engineering*, vol. 2, no. 3, pp. 25–29, 2015.
[48] M. Kurhade and R. Wankhade, "An Overview on Decision Making Under Risk and Uncertainty," *International Journal of Science and Research*, vol. 5, no. 4, pp. 416–422, April 2016.
[49] D. Pamucar, D. Bozanic and A. Randjelovic, "Multi-criteria decision-making: An example of sensitivity analysis," *Serbian Journal of Management*, vol. 12, no. 1, 2017.
[50] A. Ramos, M. Lazar, R. F. Filho and J. j P. C. Rodrigues, "A security metric for evaluation of collaborative intrusion detection systems in wireless sensor networks," In Proceedings of the 2017 IEEE International Conference on Communications (ICC), 2017, pp. 1–6.
[51] L. Zomlot, "Handling uncertainty in intrusion analysis," *Thesis for PhD*, 2014. http://doi.org/10.13140/RG.2.1.4936.4326.
[52] T. H. Ho, J. J. Hull and S. N. Sirihari, "Decision Combination in Multiple Classification Systems," *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 16, no.1, pp. 66–75, January 1994.