Applied Intelligence (2022) 52:3527–3544
https://doi.org/10.1007/s10489-021-02621-x

A lightweight intelligent network intrusion detection system using OCSVM and Pigeon inspired optimizer

Hadeel Alazzam[1] · Ahmad Sharieh[1] · Khair Eddin Sabri[1]

Accepted: 15 June 2021 / Published online: 7 July 2021

Abstract

Due to the widespread of Internet services, all around the world, service providers are facing a major problem defending their systems, especially from new breaches and attacks. Network Intrusion Detection System (NIDS) analyzes network packets and reports low-level security violations to system administrators. In large networks, these reports become unmanageable. Moreover, state-of-the-art systems suffer from high false alarms. A NIDS should be anomaly-based to have the ability to discover zero-day attacks. Most NIDSs proposed by researchers that were based on such techniques suffered from high false alarms. This paper introduces an intelligent lightweight IDS that has a low false alarm rate while maintaining a high detection rate. The proposed NIDS is a fusion between two main subsystems that work in parallel. Each subsystem is trained using One-Class Support Vector Machine (OCSVM). One of the systems is trained over normal packets, while the other is trained over attack packets. The results of both subsystems are combined to give a good judgment for each packet that passes through the network. The proposed NIDS has been evaluated and compared with state-of-the-art systems using three popular IDS datasets (KDDCUP-99, NSL-KDD, and UNSW-NB15) in terms of detection rate, accuracy, f-measure and false alarms. The results show that the proposed NIDS outperformed the examined IDSs proposed by the previous researches.

Keywords Cybersecurity · Detection system · Network intrusion · KDDCUP-99 · UNSW-NB15 · NSL-KDD · Pigeon inspired optimizer

1 Introduction

Nowadays, individuals, small businesses, or an international organizations are highly relied on computer systems and the Internet than ever before. Personal data posted on the public through social media can result in identity theft. Information theft is the fastest and most expensive cybercrime [3, 26]. Also, the industrial environment is a target for cyber attacks, where the attack disrupts or destroys an entire infrastructure by stopping an automated process, causes physical damage, and even threaten people's lives [58].

✉ Hadeel Alazzam
hdy9160095@ju.edu.jo

Ahmad Sharieh
sharieh@ju.edu.jo

Khair Eddin Sabri
k.sabri@ju.edu.jo

[1] Computer Science Department, The University of Jordan, Amman, Jordan

Cybersecurity describes the discipline, tools, mechanisms, guidelines, risk management, actions, and best practices that can be used to protect the cyber environment of organizations and user's property [30, 61]. As the number and sophistication of cyber-attacks increase, organizations have to take serious steps to protect their sensitive data especially the information related to safeguarding the national security, financial records, or health information. In 2013, the nation's top intelligent officials reveal that digital spying and cyber-attacks eclipsing the terrorism regarding threat a national security [37, 45].

Intrusion detection and prevention systems are a defense layer used to filter out possible attacks and provide a reactive mechanism against the attackers. Traditional intrusion detection systems (IDSs) are unable to dominate the tremendous sophisticated development of cyberattacks [7, 44]. An IDS is a dynamic defense layer that has the capability of adapting the dynamic behavior of network traffic and can be effective to detect all new types of attacks [42, 59].

This paper introduces a novel lightweight intelligent network-based IDS. The proposed system is lightweight since it does not use all the available data for training, instead of that, the system only trained over a small representative dataset produced by k-means clustering. Also, the proposed system composed of two main subsystems that use the One-Class Support Vector Machine (OCSVM) to develop the system. One of the subsystems is trained over normal data to detect anomalies, while the other is trained over the attack data. The subsystem that is trained on attacks aims to detect known attacks and decrease the number of false alarms raised by the first subsystem. Both subsystems will work in parallel and vote for each packet passes the network.

The main contributions of this paper can be summarized as follows:

- Summarize the state-of-the-art works related to Network-IDS.
- Produce a small representative dataset using k-means clustering.
- Propose a novel lightweight Network IDS, which is a fusion of two main subsystems that use the OCSVM.
- Analyze the run time complexity of the proposed Network-IDS.
- Evaluate the proposed IDS and compare its results with the state-of-the-art works in terms of detection rate, false alarms, accuracy, and f-score using three popular IDS datasets (KDDCUP-99, NSL-KDD, and UNSW-NB15)

The rest of this paper is organized as follows. Section 2 summarizes the state-of-the-art related works, Section 3 introduces the proposed IDS and the methodology used to develop it, Section 4 analyzes the run time complexity of the proposed system, Section 5 presents the achieved results and Section 6 concludes the paper.

2 Related works

Over the last 30 years, considerable attention has been conducted to provide efficient network intrusion detection system (NIDS). Remember that NIDS is used to monitor the network traffic, determine the suspicious behavior, and alert the system administrator.

Snort is a lightweight NIDS developed as in [52]. It is a cross-platform tool that can be used by small TCP/IP networks, and detects suspicious traffic or known attacks. Snort is a packet sniffer and logger. These features rules can make a content pattern matching to detect attacks and probes and provide the network administrator with enough information about suspicious behavior to facilitate the process of making a decision. In addition, snort can deal rapidly when any new attack emerges, in the time that, the security vendors are slow to release the new attack signature. Snort is useful when no budget is available to deploy NIDS. It is free to use in any environment, under the license of the General Public License (GNU99).

In [10] they proposed an IDS based on hybrid supervised and unsupervised Neural Networks (NN). The authors used the unsupervised NN for training normal packets and used a supervised NN that is based on the backpropagation for clustering and classification of the network attacks. The system was evaluated using DARPA 1998 dataset; the true positives, and false-positive rates were 73.67% and 26.53%, respectively.

Authors in [62] proposed a new approach for IDS called FC-ANN, which is based on Fuzzy Clustering and Artificial Neural Network (ANN). The authors used the fuzzy clustering to cluster the heterogeneous data into smaller homogeneous datasets, and then used several ANN models to train the small datasets. After training several models, a fuzzy aggregation model is used to aggregate the results of the previous models. The evaluation results of the proposed approach showed that the approach enhanced the predicted ratio for the low-frequent attacks.

In [28], they proposed an intrusion detection approach that is based on hierarchical clustering, and Support Vector Machine (SVM). The hierarchical clustering used to transform the dataset into a smaller size dataset with abstracted data points. This approach used the full KDDCUP-99 dataset, which contains 4 million records. The reduced dataset provided by the hierarchical clustering is used to feed the SVM. In order to have a high detection rate and a low false alarm, the reduced dataset must be representative. The training time for the SVM is significantly reduced, and the evaluation experiments show that this approach has 95.72% of average accuracy.

In [39], they proposed a cascade method using k-means clustering and C 4.5 decision tree to identify anomalies and normal in network traffic. The proposed method used k-means clustering to cluster the data into predefined k clusters using the Euclidean distance, then the C 4.5 decision tree is used to classify anomalies and normal traffic within the cluster in order to refine or tune the decision boundary within each cluster. The proposed approach is evaluated using six performance measures (true positive rate (TPR), Accuracy, false positive rate (FPR), Precision, F-measure, and Area under Receiver Operating Characteristic (ROC)) against the K-means, C 4.5 decision tree, Naïve Bayes, KNN, SVM and Transudative Confidence Machines KNN methods over KDDCUP-99 dataset and achieved high detection accuracy. The authors used WEKA to conduct the experiments.

Another IDS introduced in [32] contains two main approaches. The first one used simulated annealing with SVM to optimize the parameters of SVM and selected the best set of features. The second approach used a decision tree and SVM to adjust the parameters' values and build the decision rules. The proposed approach is evaluated on KDDCUP-99 dataset, and the performance of the proposed approach outperforms the winning approach from the contest and other approaches using 23 features reduced from 41 features and 99.96% classification accuracy.

In [49], they proposed a hybrid IDS that used k-means clustering and the Radial Basis Function (RBF) kernel function of SVM. The k-means clustering used a pre-classification where the dataset is reduced into homogeneous clusters. Each cluster has a reduced set of attributes in order to minimize the complexity of the full attributes' dataset. A RBF kernel function of SVM is used to train each cluster independently with a different number of attributes that affect the classification accuracy for a certain type of attack. The hybrid approach is evaluated using the KDDCUP-99 dataset in terms of detection rate, false alarms rate, and accuracy rate. The proposed hybrid approach reduces the complexity by reducing the number of features while increasing the detection rate, accuracy and minimizes the false alarms rate.

Authors in [36] developed a false alarm filter based on K-Nearest-Neighbor (KNN) classifier to filter out the false alarm. The alarm filter used a rating mechanism to classify the incoming alarms into clusters for labeling, while an expert knowledge rates the alarms in the training process and decides the rating threshold. In addition, the authors in this study investigated the effect of the value of K for the KNN classifier for this problem and found out that the best value for this problem is five. The result achieved by this study reduced the number of unwanted alarms with affordable CPU usage.

Another study in [4] used SVM to build an IDS. The authors proposed a modified K-means clustering to resize the dataset into a smaller representative one, that can be fed into SVM, and reduce the training time. A multi-level hybrid SVM and Extreme Learning Machine (ELM) NIDS is proposed, where each layer contains one classifier to classify the packets into one category. The evaluation of the multi-level SVM and ELM on KDDCUP-99 achieved 95.75% and 1.87% for accuracy and false positive rate, respectively.

Manzoor and Morgan (2017) [34] proposed an anomaly-based NIDS using a SVM to predict the category of the network traffic as a normal or an attack. This work used a real-time stream processing framework (Apache spark) and tested using KDDCUP-99 dataset. The accuracy of the approach was 73% when tested over new attacks. The conducted work did not address the false alarm problem; also, some important metrics and the experimental protocol are not defined in this study.

In [56], they proposed a comprehensive approach for intrusion detection in high-speed big data networks. The approach employed the machine learning techniques for classifications (Naïve Bayes, Random Forest, and Logistic Regression), and performed a feature selection approach using branch-and-bound, and information gain algorithms. The system was evaluated using a dataset generated by the Information Security Centre of Excellence at the University of Brunswick (ISCX-UNB) and achieved high accuracy and lower false alarm rate compared to the state of the art that used the same dataset However, their works did not clarify the bulk synchronous parallel framework they used, and how they deal with the big data challenges.

In [35], they proposed a network-based detection approach for the internet of things (IoT) botnet attacks. The method used a deep auto-encoders technique, which is a type of neural network. The authors collected the data from nine IoT devices and infected with known IoT-based botnets Mirai and BASHLITE attacks. A module for each IoT device has been trained independently of other devices; the neural network (NN) (auto-encoder) is trained to reconstruct its input after some compression. The auto-encoder is trained over normal traffic only to be capable of reconstructing the normal traffic, and misconstruction the anomalous traffic. If a significant reconstruction error appears, the input traffic can be classified as anomalies. In order to minimize the FPR, the abnormality decision was based on a sequence of instances within determined window size, where the window size is the minimum number of the sequence. The results of this study were promising, the FPR hits zero, but it is not easy to build a model for each IoT device in a network.

In [23] the authors proposed a NBSVM IDS. The proposed system used the Naive Bayes transformation technique to select features and generate new high quality dataset. The model trained by SVM on the generated dataset. The proposed system evaluated using three datasets; NSL-KDD, UNSW-NB15 and CICIDS2017 in terms of accuracy, detection rate and false alarms.

In [13] an anomaly detection approach have been proposed. The proposed approach used an enhanced PSO, Gravitational Search Algorithm, and Random Forest Classifier (PSOGSARFC). The PSOGSARFC approach used the diversity strategy to enhance the random forest in detecting anomalies. The proposed approach evaluated using NSL-KDD and UNSW-NB15 in terms of precision, recall, f-measure and accuracy.

Another study in [64] used cross-layer of Conventional Neural Networks (CNN) and Long Short-Term Memory(LSTM) networks to build intrusion detection model for advanced metering infrastructure. The CNN is used to recognise regional features, while LSTM used to recognise periodic features. The proposed model evaluated using KDDCUP-99 and NSL-KDD in terms of accuracy, detection rate and false positive rate.

Table 1 summarize the machine learning techniques used in IDS along with their pros and cons. Despite the extensive research effort, the IDS still suffers from serious issues such as high false alarms. The traditional intrusion detection methods become not appropriate due to a variety of network traffic, and the high complexity of the intrusion problem [65]. Introducing machine learning algorithms into an IDS becomes a concern for the researchers. In this paper, a novel Network-IDS is proposed, the proposed system addresses the challenge of false alarms while maintains a high detection rate.

3 The proposed intrusion detection system

The methodology used for designing the proposed IDS uses the Cross-Industry Process for Data Mining (CRISP-DM), which is a robust and well-structured methodology [40]. The CRISP-DM methodology consists of six main stages as shown in Fig. 1.

3.1 Problem understanding

With the widespread and availability of the Internet in our lives and the increasing number of cyber-attacks, there is a need for IDS that can deal with such a challenge [19]. Detecting anomalies is a good way to raise alerts and it is more efficient to discover zero-day attacks than the signature-based system, but it suffers from high false alarms [27]. There are many challenges for designing intelligent network-based IDS such as the huge amount of data that has to be analyzed, the difficulty to determine the boundaries between normal and attacks [54]. In this research, an intelligent network-based IDS that benefits from both benign and attacks log will be proposed. The proposed system will be used anomaly-based techniques to detect zero-day attacks; also, it will use the log of signatures of existing attacks to reduce the number of false alarms raised by the anomaly-based system. Both systems will work in parallel to produce intelligent lightweight IDS. Moreover, the proposed system uses a k-means clustering algorithm to produce a small representative dataset to deal with the amount of data challenge.

3.2 Data understanding

There are three datasets that will be used to design and evaluate the proposed intelligent IDS. KDDCUP-99, NSL-KDD, and UNSW-NB15 are the most popular benchmarks used by researches to evaluate and design IDS over the last years [38].

KDDCUP-99 is the most popular dataset designed for detecting intrusion in a military environment. However, it does not contain recent types of attacks since it was designed in 1999 [57]. Also, the KDDCUP-99 was used in KDDCUP 1999 competition, which makes it a popular benchmark for research to evaluate and compare their IDS. KDDCUP-99 suffers from several challenges that have been discussed in [57].

In 2013, Revathi and Malathi [51] developed a new version of the KDDCUP-99, named NSL-KDD which solved the challenges mentioned in [57]. Both KDDCUP-99 and NSL-KDD have the same set of features. Some features in KDDCUP-99 are symbolic, while the rest are continuous. Table 2 presents the features of both KDDCUP-99 and NSL-KDD dataset with their category and data type.

Table 2 illustrates the features grouped into three main categories (basic, content, and traffic). It worth mentioning that there are two training datasets of KDDCUP-99 available online, the first one has 4,898,431 instances, while the second version has only 10% of the total instances in version one. In this research, 10% of the corrected KDDCUP-99 training set was used to design and evaluate the proposed IDS. The instances in both datasets are labeled by five classes (Normal, probing, U2R, R2L, and DoS) [51]. Table 3 presents the data distribution of the 10% corrected KDDCUP-99 training set.

The training dataset contains 24 attack types, while the testing set has an additional 14 attack types that do not exist in the training dataset. Also, the testing dataset has a different data distribution from the training set [51].

To be aware of the relevant development, UNSW-NB15 is used to design and evaluate the proposed IDS. UNSW-NB15 was published in 2015 and it has 9 new modern attack types compared to 14 attack types in KDDCUP-99 [51]. UNSW-NB15 full dataset contains 49 features while the available training and testing sets contain only 45 features. The features fall into six categories; basic, content, flow, time, additionally generated, and labeled features [51]. Table 4 illustrates the list of UNSW-NB15 features with their category and data type.

3.3 Data preparation

Dataset preparation is an important step to do before processing and analyzing the data. Data preparation composed of data cleaning, normalization, transformation, and reduction [48]. The following illustrates the data preparation steps used in this research.

3.3.1 Data cleaning

The data cleaning is the process of removing irrelevant or duplicate records and handling missing data. Data cleaning is an important step to ensure that the data is consistent, valid and usable [5, 47]. It is important to remove duplicate

Table 1 Machine learning techniques used in IDS

Method	Overview	Pros	Cons	Example	Ref
Statistical Model	Based on the probabilities model to track network behavior.	-Ability to detect any new anomaly more than any other methods. - Does not require pre-knowledge about the system as input	- If an attack exists in a training phase, the system considers it normal. - Need a relevant time to train the system before the first alarm occurred.	Wavelet, PCA, Covariance Matrix, Filtering Model	Callegari et al. [14], Hamdi and Boudriga [25]
Clustering	Aims to group a set of objects into clusters, in order to find an outlier that is so far from other clusters.	- Ability to group large datasets into small ones. - Reduce complexity	- Depend on proximity measure that effect of the detection rate. - Not optimized for anomaly detection.	K-means, KNN	Horng et al. [28], Meng et al. [36], Ravale et al. [49], Wang et al. [62]
Classification	Consist of two main phases: training, and testing phase. A classifier is built during the learning or training phase using labeled data, then the classifier has to classify an instance as normal or abnormal at the testing phase.	- High detection rate for known attacks. - Flexibility for train and test new data with the existing	- Hard to find labeled datasets. - Unable to detect unknown attacks. - High resource consumption	Naïve Bayes, Support Vector Machine, Artificial Neural Network, Decision Tree	Al-Yaseen et al. [4], Gu and Lu [23], Manzoor and Morgan [34], Siddique et al. [56]
Finite State Machine (Finite Automata)	Consist of states, transitions, and actions. Each state caries the information of the past from the entry of the system to present (e.g. for a protocol). A transition occurred with the change in the system that exposes the condition for a transition	- Flexibility. - Can represent a wide variety of states. - High Detection Rate. - Time Consuming.	- Unable to detect unknown attacks. - Dynamic update of rules is costly.	Markov Chain	Ozkan et al. [41]
Evolutionary Method	Intelligence algorithms that inspired by natural.	- Learn and adapt like living beings. - Suitable for parallel processing. - Deal with noise without affecting the solution	- Hard to find suitable fitness function. - Hard to determine the optimal parameters.	Genetic Algorithm, Particle Swarm Optimization	Aslahi-Shahri et al. [9], Boahen et al. [13], Ghanem et al. [21], Hamamoto et al. [24]
Information Theory	A mathematical representation of information that affects the transmission and processing of information	- Highly scalable. - Very sensitive. - Law false alarm (false positive).	- Not appropriate for time series of traffic-related features. - Need highly presence of anomalies to detect it.	Entropy, Kullback-Leibler Distance	Amaral et al. [8], David and Thomas [15]

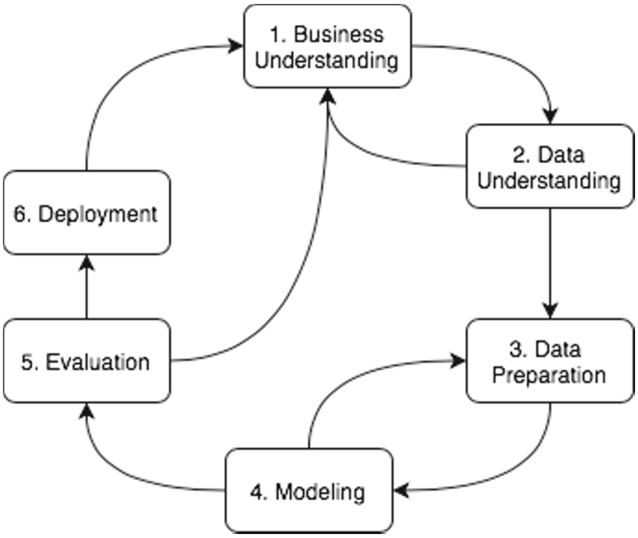


Fig. 1 CRISP-DM methodology

records in the training set to avoid the classifiers to be biased to most frequent records and prevent it from learning infrequent records. Duplicates records were removed from the KDDCUP-99 dataset, the number of training set records after eliminating duplicates is 145,584 down from 494,019 instances. Both NSL-KDD and UNSW-NB15 have no duplicate records. Also, irrelevant features have been removed from the UNSW-NB15 such as the "id" and the "attack_cat" which is redundant of the class attribute.

Another data cleaning process is to transfer the symbolic data into numeric values and label transfer. All the examined dataset (KDDCUP-99, NSL-KDD, and UNSW-NB15) have a class label which has a symbolic value such as normal or attack type. The developed IDS aims to identify the normal traffic from malicious without identifying the type of attack. All the attack's label is set to "1", while the normal instance's label is set to "0". Also, other symbolic values are transferred to numeric data such as the type of the protocol.

3.3.2 Data normalization

Data normalization is the process of transforming or scaling the data values of each feature into a proportional range [46]. Normalizing the data is an important step to eliminate the bias toward the features of larger values from the dataset [53]. All the examined datasets have been normalized into the range [0, 1] according to (1) [46].

$$X_{normalized} = \frac{(X - X_{min})}{(X_{max} - X_{min})} \tag{1}$$

3.3.3 Data splitting and clustering

The proposed IDS uses OCSVM to train the models. One of the main challenges of one-class SVM is parameter tuning [33]. In order to select the best parameter for the target dataset, the training dataset was split into training and validation. The training dataset is used to train the model, while the validation dataset is used to validate the model

Table 2 KDDCUP-99, NSL-KDD Features with their data types and categories

Category	No.	Name	Data Type
Basic	1	duration	continuous
	2	protocol_type	symbolic
	3	service	symbolic
	4	Flag	symbolic
	5	src_bytes	continuous
	6	dst_bytes	continuous
	7	Land	symbolic
	8	wrong_fragment	continuous
	9	urgent	continuous
Content	10	Hot	continuous
	11	num_failed_logins	continuous
	12	logged_in	symbolic
	13	num_compromised	continuous
	14	root_shell	continuous
	15	su_attempted	continuous
	16	num_root	continuous
	17	num_file_creations	continuous
	18	num_shells	continuous
	19	num_access_files	continuous
	20	num_outbound_cmds	continuous
	21	is_host_login	symbolic
Content	22	is_guest_login	symbolic
	23	count	continuous
	24	srv_count	continuous
	25	serror_rate	continuous
	26	srv_serror_rate	continuous
	27	rerror_rate	continuous
	28	srv_rerror_rate	continuous
	29	same_srv_rate	continuous
	30	diff_srv_rate	continuous
	31	srv_diff_host_rate	continuous
Traffic	32	dst_host_count	continuous
	33	dst_host_srv_count	continuous
	34	dst_host_same_srv_rate	continuous
	35	dst_host_diff_srv_rate	continuous
	36	dst_host_same_src_port_rate	continuous
	37	dst_host_srv_diff_host_rate	continuous
	38	dst_host_serror_rate	continuous
	39	dst_host_srv_serror_rate	continuous
	40	dst_host_rerror_rate	continuous
	41	dst_host_srv_rerror_rate	continuous
	42	Class	symbolic

Table 3 Corrected KDDCUP-99 Training Dataset Distribution

	# of Instances	Percentage %
Normal	97,277	19.69%
DOS	391,458	79.24%
Probe	4,107	0.83%
R2L	1,126	0.23%
U2L	52	0.01%
Total	494,019	100%

during parameter tuning. Figure 2 illustrates the dataset splitting process, where the validation set has 20% of the full training set. After selecting the optimal parameters for one-class SVM, the model will be tested over the testing dataset.

As known that one-class SVM works on a single class dataset. Therefore, to prepare the data for one-class SVM, the training dataset will be split up into two files: normal and attacks. After splitting the data, the class label will be eliminated. Even though SVM is robust, it has a higher run time complexity compared to other machine learning classifiers [1]. One-class SVM run time complexity is determined by the number of instances and data dimensionality. To deal with this challenge, k-means clustering is used to produce a representative small size dataset.

K-means clustering is fed with a number of clusters k, then started by selecting a random centroid for each cluster, the k-means work by assigning the data points to the nearest cluster based on the distance or the similarity between the data point and each centroid. After assigning all data points to their closest group, the centroid of each cluster will be calculated as an average of all data-points belong to that cluster, then the process of assigning the data points to the new cluster's centroid will be repeated, and the cluster's centroid will be recalculated over and over again until the centroid's values become stable [31].

In this research, the representative subset will be produced using k-means, where the instances of the new subset will be the cluster's centroids exported from the k-means. In this case, the new subset will be relatively small and represents all the data points in the main dataset. For the attack training set, the value of k was set to 100, while the for normal training set, the value of k was set to 40. All the examined dataset has the same k values. Figure 3 illustrates the splitting and clustering process for the training dataset.

3.3.4 Dimensionality reduction using Cosine_PIO

Dimensionality reduction is the process of eliminating the number of irrelevant attributes in a dataset [60]. The dimensionality reduction can be divided into main categories: feature selection and feature extraction. Feature selection

Table 4 UNSW-NB15 Features with their data types and categories

Category	No.	Name	Data Type
Flow	1	srcip	nominal
	2	sport	integer
	3	dstip	nominal
	4	dsport	integer
	5	proto	nominal
Basic	6	state	nominal
	7	dur	Float
	8	sbytes	Integer
	9	dbytes	Integer
	10	sttl	Integer
	11	dttl	Integer
	12	sloss	Integer
	13	dloss	Integer
	14	service	nominal
	15	Sload	Float
	16	Dload	Float
	17	Spkts	integer
	18	Dpkts	integer
Content	19	swin	integer
	20	dwin	integer
	21	stcpb	integer
	22	dtcpb	integer
	23	smeansz	integer
	24	dmeansz	integer
Content	25	trans_depth	integer
	26	res_bdy_len	integer
Time	27	Sjit	Float
	28	Djit	Float
	29	Stime	Timestamp
	30	Ltime	Timestamp
	31	Sintpkt	Float
	32	Dintpkt	Float
	33	tcprtt	Float
	34	synack	Float
	35	ackdat	Float
General Purpose	36	is_sm_ips_ports	Binary
	37	ct_state_ttl	Integer
	38	ct_flw_http_mthd	Integer
	39	is_ftp_login	Binary
	40	ct_ftp_cmd	integer
Connection	41	ct_srv_src	integer
	42	ct_srv_dst	integer
	43	ct_dst_ltm	integer
	44	ct_src_ ltm	integer
	45	ct_src_dport_ltm	integer
	46	ct_dst_sport_ltm	integer
	47	ct_dst_src_ltm	integer
	48	attack_cat	nominal
	49	Class	binary

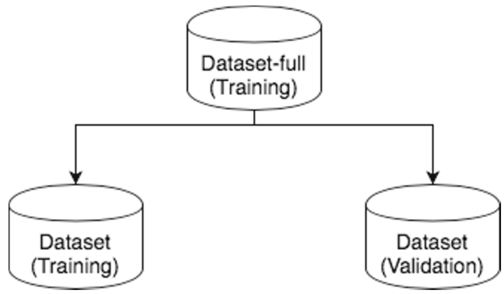


Fig. 2 Dataset splitting

considers selecting a subset of features, while feature extraction uses the existing features to produce new features by merging or using statistical analysis. In this research, a binary Pigeon Inspired Optimizer (Cosine_PIO) proposed by [6] for feature selection is used to select the optimal subset of features. The solution is represented by a one-dimensional array with a length of the number of features in the dataset, initially, the solutions are randomly generated with binary value zero or one. the "zero" indicates that the corresponding feature is absent in the current solution while the "one" indicates the presence of the corresponding feature in the solution.

The Cosine_PIO uses the cosine similarity to calculate pigeon velocities. The PIO is one of the newly developed bio-inspired algorithms. It mimics the pigeons homing skills to find their home back. The homing skills of pigeon can be expressed mathematically by two main operators: Map and compass, and landmark operators. When pigeons become closer to their destination, they rely less on map and compass operator. Equations 2–3 present the mathematical model of the map and compass operator for the cosine_PIO. (2) presents the velocity calculation for pigeon using cosine similarity.

$$V_p = Cosine\ Similarity(X_g, X_p) = \frac{X_g.X_p}{||X_g||.||X_p||} = \frac{\sum_{i=0}^{n-1} X_{p,i} X_{g,i}}{\sqrt{\sum_{i=0}^{n-1} X_{p,i}{}^2}\sqrt{\sum_{i=0}^{n-1} X_{g,i}{}^2}} \tag{2}$$

Where X_g is the pigeon with the best fitness value among the others. Equation 3 presents how the pigeons positions are updated based on their velocities.

$$X(t)_{(i,p)}[i] = \begin{cases} X(t-1)_p[i], & if(V_p) > r) \\ X(t-1)_g[i], & \text{otherwise} \end{cases} \tag{3}$$

All pigeons are evaluated according to their fitness value. Pigeons are evaluated in terms of TPR, and FPR. Equation 4 presents the fitness function used to evaluate each pigeon.

$$Fitness\ Function = FPR + \frac{1}{TPR} \tag{4}$$

The second operator (landmark) is used to adjust the direction of the pigeons when it gets closer to the destination. All pigeons are ranked according to their fitness value, then only the top half number of pigeons is considered to calculate the desired destination in each iteration. Each pigeon adjusts its position according to the new calculated desirable destination as in Fig. 4

The mathematical model of the landmark operator is represented by (5) - (6).

$$N_p(t+1) = \frac{N_p(t)}{2} \tag{5}$$

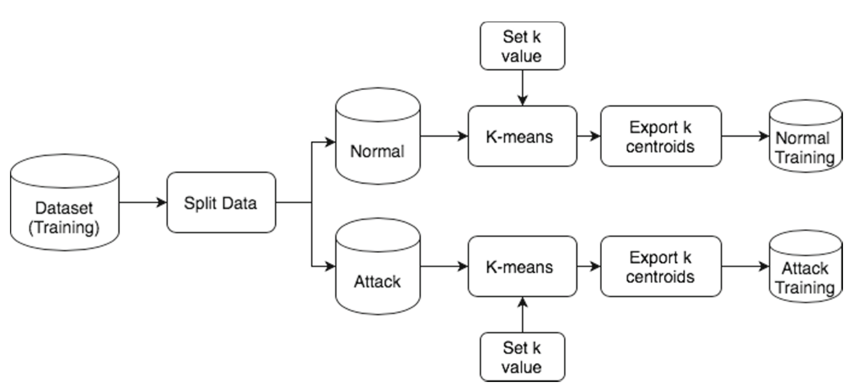
Where N_p is the number of pigeons in the current iteration t.

$$X_c(t+1) = \frac{\sum X_i(t).Fitness(X_i(t))}{N_p \sum Fitness(X_i(t))} \tag{6}$$

Where X_c is the new position desirable destination based on half the number of pigeons from the previous iteration, while X_i is the current position of $pigeon_i$. Each pigeon will update its position and velocity by (2) and (3) to X_c instead of X_g, the landmark operator will stop if the number of pigeons N_p reach one.

Figure 5 illustrates an abstraction of the feature selection process. The rest of the feature selection process will be illustrated in the model development section.

Fig. 3 Dataset splitting and clustering



3.4 Model development

The model development phase composed of three main processes: feature selection, parameter tuning, and model training. During model development, two main processes crossed which are the feature selection and parameter tuning of the one-class SVM. Two main parameters of the one-class SVM have to be tuned according to the data: gamma (γ) and nu (ν). The tuning of SVM parameters considered an open problem. The γ parameter determines the influence of the radius on the kernel. The range of γ depends on the data and the application, while the ν parameter is an upper bound for the ratio of errors in the training set and a lower bound for the ratio of support vectors [18]. The ν value range is between (0,1]. Many researchers use a grid search to find the best values for γ and ν [2, 20, 63].

In this research, the γ value was set to "scale", which is a default configuration in one-class SVM. The scale value is equal to $1/(n\_features * X.var())$, where X is the training dataset. While the ν value is set to a set of νs possible values. The operations of parameter tuning and feature selection are overlapping since the feature selection produces a new dataset that different based on the selected feature, and the values of the γ and ν depend on the dataset.

Algorithm 1 illustrates the procedure of selecting the subset of features while tuning the parameters. As we mentioned earlier, the feature selection in this paper is done using PIO. The kernel used to train the one-class SVM (OCSVM) is the RBF kernel, which is a popular kernel used for most kernelized machine learning.

Algorithm 1 Feature selection with OCSVM parameters tuning.

Input: Population size N_p, Space Dimension D, Map and compass factor R, Number of iterations nc_1, nc_2 where $nc_1 > nc_2$.
Output: Global Solution X_g, γ, ν

Initialize X_i for each Pigeon randomly.
Evaluate Pigeons $(X_1, X_2, \ldots, X_{N_p})$ by their fitness values.
Xg = best pigeon (minimum fitness)
while $(n_c >= 1)$ **do**
    Update velocity and path for each pigeon by map and compass operator.
    for each Pigeon $(X_1, X_2, \ldots, X_{N_p})$ **do**
        for each ν in νs **do**
            Train OCSVM (γ = "scale", ν = "ν", kernel= "rbf")
            Evaluate Pigeons $(X_1, X_2, \ldots, X_{Np})$ by their fitness values.
        end for
    end for
    Return best ν, γ and pigeon.
    Update X_g
end while
while $(N_p >= 1)$ **do**
    Sort pigeons by their fitness values.
    $Np = N_p/2$
    Calculate desired destination
    Update pigeons position's toward the desired destination.
    for each Pigeon $(X_1, X_2, \ldots, X_{N_p})$ **do**
        for each ν in νs **do**
            Train OCSVM (γ = "scale", ν = "ν", kernel= "rbf")
            Evaluate Pigeons $(X_1, X_2, \ldots, X_{Np})$ by their fitness values.
        end for
    end for
    Return best ν, γ and pigeon.
    Update X_g
end while

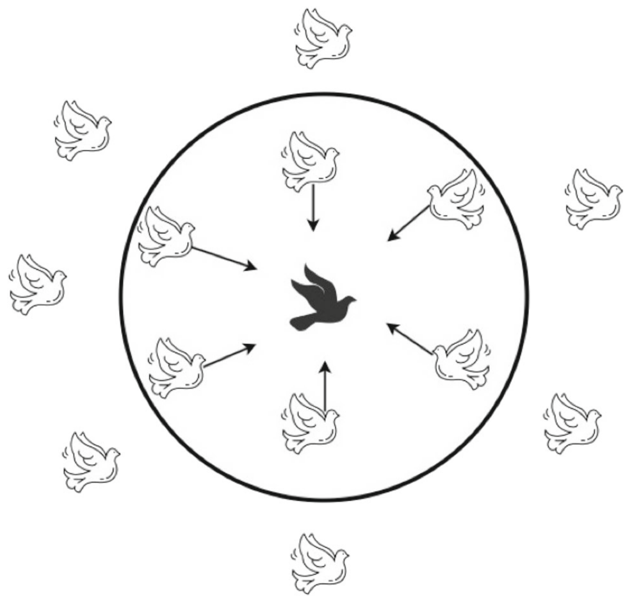


Fig. 4 The black pigeon presents the desirable destination calculated according to the half number of white pigeons in the circle [6]

Figure 6 illustrates the overall methodology design, starting from problem understanding as defined in Section 3.2 to data preparation in Section 3.3, and data splitting and clustering in Section 3.3.3, followed by feature selection using PIO and parameter tuning of the OCSVM that has been discussed in Section 3.3.4. The next section will describe the proposed IDS design based on this methodology.

3.5 The proposed IDS design

Based on the methodology used, a novel design for a network intrusion detection system is proposed. Figure 7 shows the proposed design for a network intrusion detection system. The proposed system consists of two main subsystems that work in parallel: the intrusion prevention system and anomaly detection system. Both subsystems will check up the network packet in parallel. The intrusion prevention system will be trained on attacks dataset only, while the anomaly detection system will be trained on a normal dataset only. The idea behind training the IDS on attacks is to get benefits from the attacks log and decrease the number of false alarms raised by the anomaly detection system.

The integration of the two subsystems gives a lightweight system, very fast, and robust. Both subsystems will use the OCSVM to train the model. Two votes will be taken in parallel for each packet passes the network. A flag with two bits for each packet will be added to represent the votes of both systems. The first vote is reserved for the IDS, while the second one is reserved for the anomaly detection system. The flag inputs only zero or one, zero indicates that the packet is not a part of the class, while one indicates that the packet is part of the class. There are four possible combinations for the flag bits.

Table 5 illustrates the possible combinations of the flag bits. According to Table 5, the packets with {0, 0} flag indicate that both systems did not recognize the packet to their class. In this case, this packet will be treated as an anomaly. The anomaly packet will be prevented or passed the network based on the system configuration that depends on the application type. The packet with {0,1} flag indicates that the anomaly detection system did not identify it as normal, while the intrusion prevention system identifies it as an attack, therefore it will be determined as an attack. The third combination is {1,0} which means that the anomaly detection system recognizes it as normal while the IDS did not recognize it as an attack, so it will be determined as a normal packet. The last combination is {1, 1}, which indicates that the two systems have a contradiction. The anomaly detection recognizes it as a normal packet, while the intrusion prevention recognizes it as an attack. In this case, the packet will be determined as an attack, since the intrusion prevention has a stronger vote than the anomaly detection system.

The anomaly packets will be filtered out into a log file to be labeled by the system administrator to make the system up to date with recent attacks and network behavior. Figure 7 illustrates the proposed network IDS design.

3.6 System evaluation

The proposed NIDS will be evaluated according to the detection rate, false alarms, f-score, G-mean, and accuracy. The evaluation process of the proposed system will be measured in two phases, the first phase will evaluate the subsystems independently, and then the second phase will evaluate the overall system. Also, the proposed system will be compared to the-state-of-the-art works using three datasets (KDDCUP-99, NSL-KDD, and UNSW-NB15).

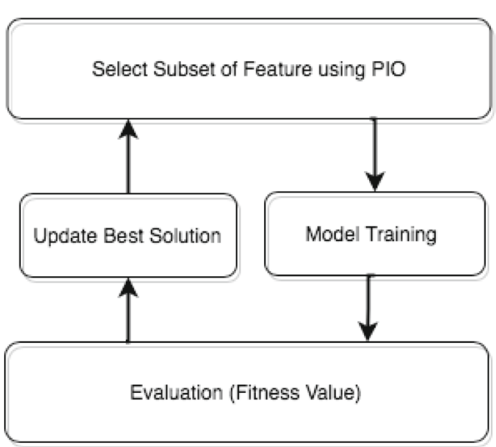


Fig. 5 Feature Selection Process

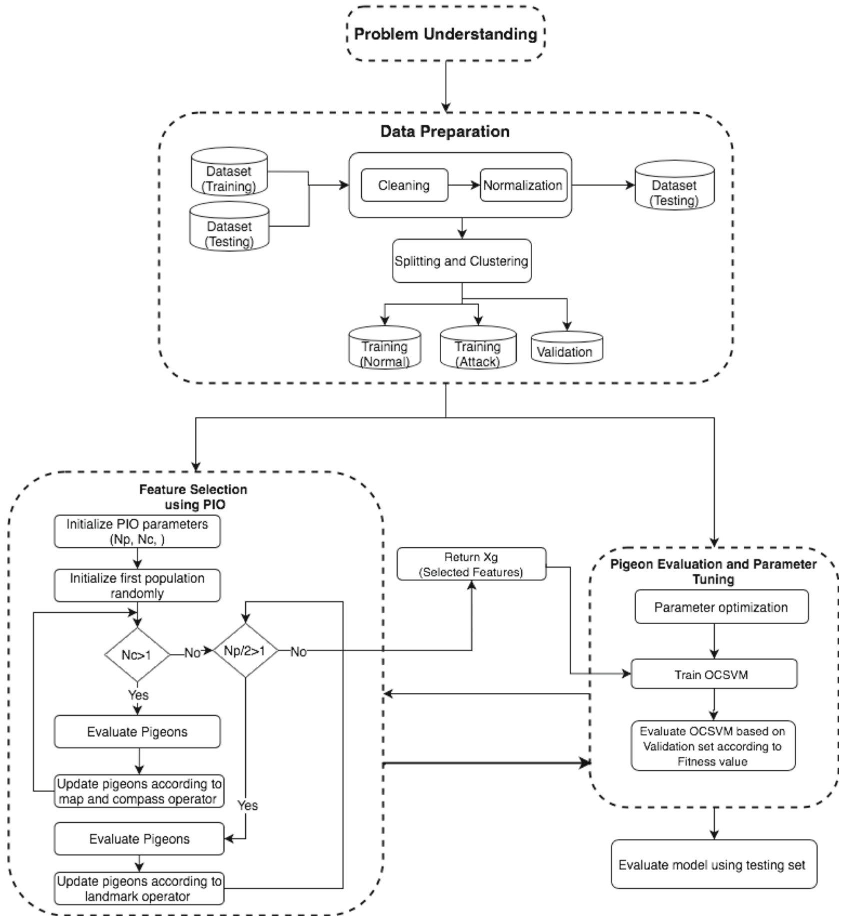


Fig. 6 Methodology design for the model development

4 Run complexity analysis

In this section, we evaluate the proposed NIDS in terms of run time complexity. The run complexity analysis of the proposed NIDS is divided into two main parts, the training part, and the testing part. The training part includes clustering the data, feature selection using PIO and parameter tuning, and model training process. The overall complexity of the proposed NIDS is illustrated by (7).

$$\textit{Overall complexity} = O(\textit{Training}) + O(\textit{Prediction}) \tag{7}$$

4.1 Training complexity analysis

The complexity analysis of system training is composed of data clustering and feature selection using PIO. The feature selection process involves tuning the ν parameter, training the model using OCSVM, and validation process. (8) illustrates the complexity of model training in general.

$$\textit{Complexity of training} = O(\textit{Clustering}) + O(\textit{Feature Selection}) \tag{8}$$

- Clustering Complexity

  The k-means clustering algorithm complexity is $O(kITn)$ where k is the number of clusters, the I is the number of instances in the dataset, T is the time required for calculating the distance between two data points and n is the number of iteration required by the k-means to converge.
- Feature Selection Complexity

  The run time complexity of PIO depends on the main parts of the algorithms; the initialization phase of the population, the map and compass operator, and the landmark operator. The map and compass operator and the landmark operator are used to generate a new solution. The fitness function is used to evaluate the pigeon which includes the training and the predicting time that depends on the classifier used. Finally, it depends on the population size, the number of iterations, and the dimension of the population (number of features) as in (9).

$$\begin{aligned}&O(\textit{Feature Selection with PIO})\\&= O(\textit{Initialization phase})\\&+O(runtime\ of\ map\ and\ compass)\\&+O(runtime\ of\ landmark\ operator)\\&+O(fitness\ function)\end{aligned} \tag{9}$$

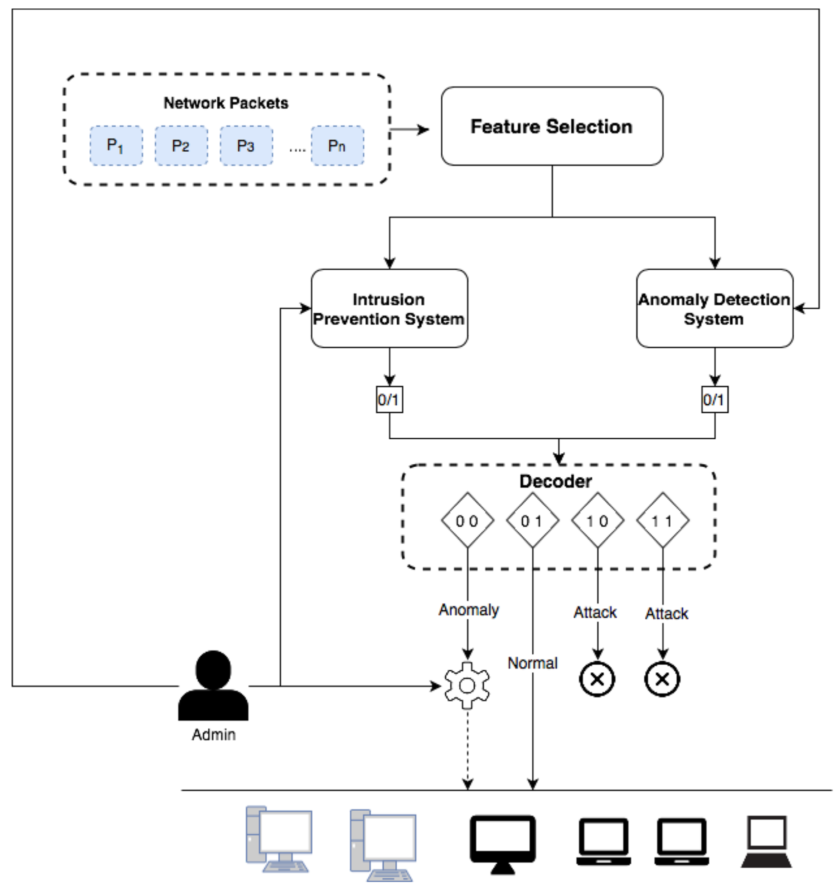


Fig. 7 The proposed network IDS design

The complexity of the map and compass operator is O (FN_p) where N_p is the number of pigeons in the population. In the map and compass operator, every feature will be updated. The complexity of the landmark operator is O $(FN_p \log_2 N_p)$ since after each iteration, the number of pigeons will be decreased to half. The complexity of the initialization phase is O(FN_p). The complexity of the fitness function will depend on training, parameter tuning, and the predicting time of the OCSVM. The training time will be repeated 4 times, which equals to the number of ν in νs array. Then, the time complexity of fitness function for each pigeon becomes O$(4 * (k^2 FN_p + k^3) + FN_p)$. The overall run time complexity is presented in (10).

$$
\begin{aligned}
&O(Feature\,selection\,with\,PIO)\\
&= O(FN_p) + O(FN_p * t_1)\\
&+O((FN_p \log_2 N_p) * t_2) + O(k^2 FN_p + k^3 + FN_p)\\
&= O(k^2 FN_p + k^3 + FN_p) \qquad (10)
\end{aligned}
$$

The feature selection complexity is O$(I^2 FN_p)$, where the I is the number of instances in the dataset and F is the number of features (attributes) and Np is the population size. But the proposed system will work on k instances only, where $k << I$.

Table 5 Possible combinations of flag bits

System Type		Decision
Anomaly Detection	Intrusion Prevention	
0	0	Anomaly
0	1	Attack
1	0	Normal
1	1	Attack

4.2 Testing complexity analysis

The complexity analysis of the testing phase is equal to the prediction time of both subsystems. The prediction time is determined by the prediction time of OCSVM which is 2* O$(n_{sv}F)$. Where the n_{sv} is the number of support vectors and F is the number of features. Also, the voting process requires O(1) operation. Since the two subsystems are predicting the packet stream in parallel, then the overall run time complexity of the testing phase is O$(n_{sv}F)$.

5 Experiments and results

This section introduces the performance metrics used to evaluate the proposed Network-IDS and discusses the conducted experiments.

5.1 Performance metrics

There are several evaluation metrics that have been proposed to evaluate an IDS. Evaluation metrics can be classified into two major categories. The first one is used for evaluating the efficiency of an IDS. This category evaluates the system in terms of the resources needed (e.g. CPU, Memory, ...) to allocate the system. On the other hand, the second category is used for evaluating the effectiveness of the system. The effectiveness of IDS deals with the system ability to distinguish between intrusion or benign traffic [55]. Most of the researchers did not take into account the efficiency of the system; they focused on evaluating the accuracy and the effectiveness of their proposed IDS systems in terms of false alarms, and detection rate. In this section, we define the set of performance metrics widely used to evaluate IDS. All selected metrics calculations are based on the confusion matrix output. Figure 8 illustrates the output of the confusion matrix.

Performance metrics definition and formulas according to the confusion matrix as following [55]:

- **Sensitivity** (True Positive Rate (TPR), Detection Rate or Recall): Measures the proportion of actual attacks that are correctly identified as in (11).

$$TPR = \frac{TP}{TP + FN} \tag{11}$$

- **Accuracy**: Measures the proportion of correct classified classes to the total number of classifications as in (12).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{12}$$

	Predicted No	Predicted Yes
Actual No	TN	FP
Actual Yes	FN	TP

Fig. 8 Confusion matrix output

- **False Positive Rate** (FPR or False Alarms): Measures the proportion of normal that are identified as attacks as in (13).

$$FPR = \frac{FP}{TN + FP} \tag{13}$$

- **F-score** (F-measure): Measures the accuracy of the model by considering both precision and recall as in (14).

$$F - Score = \frac{2 * TP}{2 * TP + FP + FN} \tag{14}$$

5.2 Results

In this section, the proposed NIDS will be evaluated in terms of detection rate, false alarms, accuracy and g-mean and compared with the state-of-the-art related works using three popular datasets mentioned previously (KDDCUP-99, NSL-KDD, and UNSW-NB15).

As mentioned earlier, the proposed NIDS is composed of two subsystems, both subsystems used the OCSVM and PIO feature selection to train the desired model. In this section, the results of each subsystem will be evaluated independently, then it will be compared with the full system to project out the benefit of fusion the votes of both subsystems.

Note that the proposed NIDS used semi-supervised machine learning techniques to train the proposed models such as OCSVM. To be fair in comparison with related works, only anomaly detection techniques (semi-supervised) were chosen for comparison purposes such as (Isolation Forest (iForest), OCSVM, auto-encoder, Gaussian mixture model, and K-nearest neighbor (KNN)).

The results discussion will be divided into three phases, according to the datasets used to evaluate the proposed system. The first evaluation will be using the popular KDDCUP-99 dataset. Table 6 illustrates the parameters' setting for the OCSVM and the PIO for training both subsystems: intrusion prevention system, and anomaly detection system. As mentioned previously that the intrusion prevention system trained using only attack records, while the anomaly detection system trained over normal data.

Table 7 illustrates the set of features produced by the PIO feature selection algorithm for both subsystems using the three selected datasets. Note that the feature index is started from zero, to map the feature number with feature name mentioned in Tables 2 and 4, it should be incremented by 1.

Figure 9 illustrates the training and testing time for the three examined datasets using the proposed system. As the figure shows that the training time is much less than the testing time, since the model is trained over a

Table 6 Parameter settings for OCSVM and PIO

Parameter	Value
OCSVM Parameters	
ν	[0.1, 0.01, 0.001, 0.074]
γ	scale
kernel	RBF
ν	[0.1, 0.01, 0.001, 0.074]
PIO Parameters	
Map and Compass Factor (R)	0.09
Population size (Np)	128
Number of Iterations	100

small representative dataset that have 1200 instances only, while the testing set contain 22544, 311020 and 82322 for NSL-KDD, KDDCUP99 and UNSW-NB15 respectively.

5.3 KDDCUP-99 results

The evaluation of the proposed NIDS using the KDDCUP-99 dataset goes through two phases: the first phase is illustrated in Table 8 were each subsystem is evaluated in terms of DR, false alarms, accuracy, g-means and f-measure. This evaluation illustrates how the second subsystem (training the OCSVM over attack data) affects the results of the main system by decreasing the false alarms rate. As the results in Table 8, it shows that the DR for the intrusion prevention system was 93%, and 99.8% for the anomaly detection system, while the fusion of both subsystems was 99.8%. Despite the high DR of the anomaly detection system, the false alarms rate was 3.4% compared to the false alarms rate of the intrusion prevention system which was 0.01%.

Table 9 illustrates the results of the proposed IDS compared with other proposed IDSs from the literature using KDDCUP-99 in terms of DR, FPR, accuracy, g-means, and f-measure. The "-" sign indicates that the value for the corresponding measure was not reported. The IDSs compared with the proposed system were based on several machine learning techniques such as Isolation Forest (iForest), v-SVC, hierarchical clustering and SVM, CNN-LSTM, autoencoder and ANN with an evolutionary algorithm. The results show that the proposed IDS outperformed all the examined IDS with the highest DR 99.8% and 0.2% false alarms rate. The SDAEs proposed in [64] came in second place with 99.9%, and 3% in terms of DR and FPR respectively. The system proposed by [16] came in the third place with 96.85% in terms of accuracy, while the system proposed in [22] achieved the best FPR compared to all other examined IDS with 0.01% but suffers from low DR.

Table 7 The set of features for both subsystems for KDDCUP-99, NSL-KDD, and UNSW-NB15

Dataset	Attacks	Normal
KDDCUP-99	[0, 3, 4, 6, 12, 14, 15, 16, 17, 19, 21, 25, 27, 30, 32, 33, 37]	[1, 4, 9, 11, 13, 18, 20, 21, 25, 31, 32, 35, 36, 37, 39]
NSL-KDD	[9, 12, 20, 24, 25, 27]	[1, 7, 11, 16, 18, 23, 25, 28, 30, 31, 32, 35, 36, 38]
UNSW-NB15	[0, 1, 2, 3, 9, 15, 16, 17, 23, 30, 31, 32, 33, 34, 35, 36, 39]	[1, 2, 3, 5, 6, 7, 8, 10, 14, 16, 17, 18, 19, 23, 25, 26, 27, 28, 29, 31, 32, 35, 36, 39, 40]

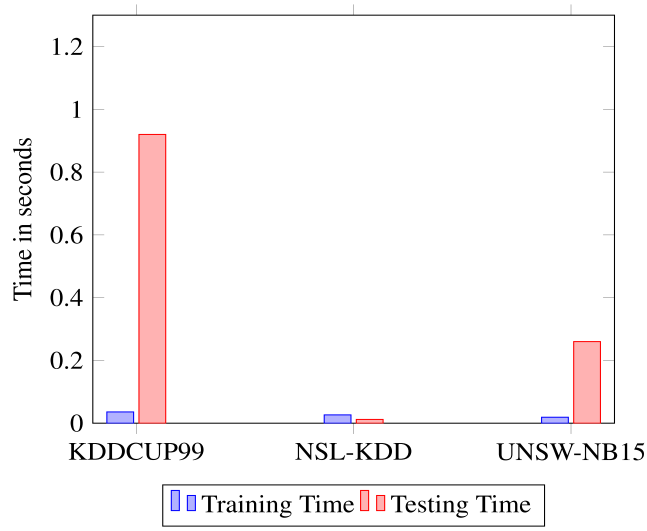


Fig. 9 Training and Testing time for the three datasets (KDDCUP99, NSL-KDD and UNSW-NB15)

5.4 NSL-KDD results

The evaluation criteria of the proposed system using the NSL-KDD dataset are similar to the criteria used with KDDCUP-99. First, the proposed system evaluation illustrated in Table 10 were each subsystem is evaluated in terms of DR, false alarms, accuracy, g-means and f-measure. Both subsystems have high DR, but the anomaly detection subsystem which is based on training the OCSVM over the normal records only suffers from high false alarms with a 30% rate. The overall system which is based on the fusion of both subsystems reduces the false alarms rate significantly.

Table 11 presents the comparison results between the proposed IDS with the state-of-the-art proposed systems using the NSL-KDD dataset in terms of DR, false alarms, accuracy, g-mean, and f-measure. The "-" sign indicates that the value for the corresponding measure was not reported. The IDS proposed by the researches were based on Self Organization Map (SOM), Artificial Neural Network (ANN), iForest, OCSVM, PSOGSARF, CNN-LSTM and Local Outlier Factor (LOF). The results show that the

Table 8 Evaluation of the proposed system using KDDCUP-99 dataset

Approach	DR	False Alarms	Accuracy	G-mean	F-measure
Intrusion Prevention System (OCSVM, Attack)	0.93	0.0001	0.941	0.964	0.963
Anomaly Detection System (OCSVM, Normal)	0.998	0.034	0.969	0.9817	0.909
Fusion of both subsystems (IDS)	0.998	0.002	0.997	0.998	0.992

Table 9 Comparison of several NIDSs using the KDDCUP-99 dataset

Reference	Approach	DR	FPR	Accuracy	G-mean	F-measure
Giacinto et al. [22]	$\nu - SVC$	0.9291	0.0001	–	–	–
Horng et al. [28]	hierarchal clustering & SVM	–	0.0073	0.957	–	–
Farahnakian [16]	iForest	–	–	0.9089	–	–
Farahnakian [16]	SDAEs	–	–	0.9685	–	–
Farahnakian and Heikkonen [17]	Autoencoder	0.9565	0.0035	0.9653	–	–
Zong et al. [66]	Gaussian Mixture	0.9442	–	–	–	0.9369
Benmessahel et al. [12]	ANN(FNN–LSO)	0.898	0.0221	–	–	–
Yao et al. [64]	CNN–LSTM	0.999	0.03	0.99	–	–
Proposed Approach	(OCSVM, PIO)	0.998	0.002	0.997	0.998	0.992

Table 10 Evaluation of the proposed system using the NSL-KDD dataset

Approach	DR	False Alarms	Accuracy	G-mean	F-measure
Intrusion Prevention System (OCSVM, Attack)	0.997	0.0005	0.997	0.998	–
Anomaly Detection System (OCSVM, Normal)	0.998	0.302	0.830	0.752	–
Fusion of both subsystems (IDS)	0.997	0.0001	0.998	0.998	0.994

Table 11 Comparison of several NIDSs using the NSL-KDD dataset

Reference	Approach	DR	FPR	Accuracy	G-mean	F-measure
Karami [29]	SOM	0.9153	0.007	0.994	–	0.9232
Benmessahel et al. [11]	MVO–ANN	0.9625	0.0003	0.9821	–	
Pérez et al. [43]	OCSVM	0.95	–	0.92	–	0.93
Pérez et al. [43]	iForest	0.35	–	0.62	–	0.51
Pérez et al. [43]	LOF	0.5	–	0.42	–	0.5
Benmessahel et al. [12]	ANN(FNN–LSO)	0.898	0.0221	–	–	0.9305
Boahen et al. [13]	PSOGSARF	–	–	0.985	–	0.986
Yao et al. [64]	CNN–LSTM	0.997	0.0034	0.997	–	–
Proposed Approach	(OCSVM, PIO)	0.997	0.0001	0.998	0.998	0.994

Table 12 Evaluation of the proposed system using the UNSW-NB15 dataset

Approach	DR	False Alarms	Accuracy	G-mean
Intrusion Prevention System (OCSVM, Attack)	0.985	0.0007	0.985	0.992
Anomaly Detection System (OCSVM, Normal)	0.999	0.478	0.529	0.722
Fusion of both subsystems (IDS)	0.999	0.0006	0.993	0.996

proposed IDS outperform all other examined systems with 99.8%, 99.7%, and 0.2% for DR, accuracy, and FPR respectively. The MVO-ANN system proposed by [64] came in second place with 99.7%, 99.7% and 0.34% for DR, accuracy and FPR, respectively. The system proposed by [11] came in the third place with 96.2%, 98.2% and 0.03% for DR, accuracy and FPR, respectively.The worst IDS was the one used the LOF to train the model, which achieved only 42% for accuracy.

5.5 UNSW-NB15

The last dataset used for evaluation is the UNSW-NB15. Table 12 illustrates the evaluation results of the proposed system using the UNSW-NB15 dataset. The results achieved by the intrusion prevention system which is based on training the OCSVM with PIO over the attack records only were 98.5%, and 0.07% for DR and false alarms, respectively. The achieved results were high, but the results achieved by the anomaly detection subsystem suffer from high false alarms with 47.8%. The results achieved by the overall system which consider the votes of both subsystems were 99.9%, 0.06 and 99.3% for DR, false alarms, and accuracy, respectively. The results of the overall system will be compared with the state-of-the-art related works in Table 13.

Table 13 presents the comparison results of the proposed NIDS with the proposed state-of-the-art works using the UNSW-NB15 in terms of DR, FPR, accuracy, g-mean and f-measure. The "-" sign indicates that the value for the corresponding measure was not reported. The NIDSs proposed by the researches were based on SOM, ANN, iForest, OCSVM, PSOGSARF, NB-SVM, and LOF. The proposed NIDS outperformed all examined IDS with 99.9%, 0.06 and 99.3% for DR, FPR and accuracy, respectively. The MVO-ANN system proposed in [11] came in second place with 99.65% and 0.004% for DR and FPR, respectively. While the system that used the SOM came in third place with 87.44%, 1.95% and 98.26% for DR, FPR and accuracy, respectively. The IDS that used the iForest to train the model achieved the worst results with 19%, 55% and 31% for DR, accuracy and f-measure, respectively.

6 Conclusion

In this paper, a lightweight intelligent IDS based on pigeon inspired optimizer and OCSVM is proposed. This IDS aims to reduce the number of false alarms while maintaining a high detection rate. It uses the binary feature selection algorithm based on PIO proposed in [6] to select the optimal subset of features for normal traffic and for attack traffic independently. Moreover, OCSVM parameters have been tuned during the feature selection process.

The proposed NIDS consists of two subsystems: intrusion prevention system which uses the OCSVM and PIO for training a model on only attack records, while the second subsystem uses the same techniques for training a model over normal records. The NIDS is a fusion of both subsystems to judge each packet passes the network.

Table 13 Comparison of several NIDSs using the UNSW-NB15 dataset

Reference	Approach	DR	FPR	Accuracy	G–mean	F–measure
Karami [29]	SOM	0.8744	0.0195	0.9826	–	0.8917
Benmessahel et al. [11]	MVO–ANN	0.9965	0.00004	0.9961	–	–
Pérez et al. [43]	iForest	0.19	–	0.55	–	0.31
Pérez et al. [43]	LOF	0.72	–	0.83	–	0.82
Pérez et al. [43]	OCSVM	0.64	–	0.79	–	0.77
Benmessahel et al. [12]	ANN (FNN–LSO)	0.993	0.094	–	–	0.959
Ren et al. [50]	DO–IDS	0.487	0.124	0.865	–	0.488
Boahen et al. [13]	PSOGSARF	–	–	0.989	–	0.988
Gu and Lu [23]	NB–SVM	0.952	0.087	0.952	–	–
Proposed Approach	(OCSVM, PIO)	0.999	0.0006	0.993	0.996	0.992

The evaluation process for the proposed system is divided into two rounds, the first round evaluates each subsystem independently then compared it with the total system. The second round evaluates the system by comparing it with the state-of-the-art related works in terms of DR, FPR, accuracy, f-measure, and g-mean. All experiments used three famous IDS datasets (KDDCUP-99, NSL-KDD, and UNSW-NB15) for evaluation. The proposed system outperforms the state-of-the-art works using the three mentioned datasets.

References

1. Abdiansah A, Wardoyo R (2015) Time complexity analysis of support vector machines (svm) in libsvm. Int J Comput Appl 128:28–34
2. Aggarwal A, Sahay T, Bansal A, Chandra M (2015) Grid search analysis of nu-svc for text-dependent speaker-identification. In: 2015 Annual IEEE india conference (INDICON). IEEE, pp 1–5
3. Al-Azzam S, Sharieh A, Al-Sharaeh S, Azzam N (2020) A data estimation for failing nodes using fuzzy logic with integrated microcontroller in wireless sensor networks. Int J Electric Comput Eng (2088-8708) 10
4. Al-Yaseen WL, Othman ZA, Nazri MZA (2017) Multi-level hybrid support vector machine and extreme learning machine based on modified k-means for intrusion detection system. Expert Syst Appl 67:296–303
5. Alazzam H, Alsmady A, Shorman AA (2019) Supervised detection of iot botnet attacks. In: Proceedings of the second international conference on data science, E-Learning and information systems, pp 1–6
6. Alazzam H, Sharieh A, Sabri KE (2020) A feature selection algorithm for intrusion detection system based on pigeon inspired optimizer. Expert Syst Appl 148:113249
7. Albdour L, Manaseer S, Sharieh A (2020) Iot crawler with behavior analyzer at fog layer for detecting malicious nodes. Int J Commun Netw Inform Secur 12:83–94
8. Amaral AA, de Souza Mendes L, Zarpelão BB, Junior MLP (2017) Deep ip flow inspection to detect beyond network anomalies. Comput Commun 98:80–96
9. Aslahi-Shahri B, Rahmani R, Chizari M, Maralani A, Eslami M, Golkar MJ, Ebrahimi A (2016) A hybrid method consisting of ga and svm for intrusion detection system. Neural comput Appl 27:1669–1676
10. Bahrololum M, Salahi E, Khaleghi M (2009) Anomaly intrusion detection design using hybrid of unsupervised and supervised neural network. Int J Comput Netw Commun (IJCNC) 1:26–33
11. Benmessahel I, Xie K, Chellal M (2018) A new evolutionary neural networks based on intrusion detection systems using multiverse optimization. Appl Intell 48:2315–2327
12. Benmessahel I, Xie K, Chellal M, Semong T (2019) A new evolutionary neural networks based on intrusion detection systems using locust swarm optimization. Evol Intel 12:131–146
13. Boahen EK, Bouya-Moko BE, Wang C (2021) Network anomaly detection in a controlled environment based on an enhanced psogsarfc. Comput Secur 104:102225
14. Callegari C, Giordano S, Pagano M, Pepe T (2011) Combining sketches and wavelet analysis for multi time-scale network anomaly detection. Comput Secur 30:692–704
15. David J, Thomas C (2015) Ddos attack detection using fast entropy approach on flow-based network traffic. Procedia Comput Sci 50:30–36
16. Farahnakian F (2018) Anomaly-based intrusion detection using deep neural networks. Int J Digit Content Technol Appl 12:70–18
17. Farahnakian F, Heikkonen J (2018) A deep auto-encoder based approach for intrusion detection system. In: 2018 20th international conference on advanced communication technology (ICACT). IEEE, pp 178–183
18. Fourie C, Van Niekerk A, Mucina L (2011) Optimising a one-class svm for geographic object-based novelty detection. In: Proceedings of the first AfricaGeo conference, pp 1–25
19. Gao W, Morris TH (2014) On cyber attacks and signature based intrusion detection for modbus based industrial control systems. J Digit Forens Secur Law 9:3
20. Ghafoori Z, Rajasegarar S, Erfani SM, Karunasekera S, Leckie CA (2016) Unsupervised parameter estimation for one-class support vector machines. In: Pacific-Asia conference on knowledge discovery and data mining. Springer, pp 183–195
21. Ghanem WAH, Jantan A, Ghaleb SAA, Nasser AB (2020) An efficient intrusion detection model based on hybridization of artificial bee colony and dragonfly algorithms for training multilayer perceptrons, vol 8, pp 130452–130475
22. Giacinto G, Perdisci R, Del Rio M, Roli F (2008) Intrusion detection in computer networks by a modular ensemble of one-class classifiers. Inform Fusion 9:69–82
23. Gu J, Lu S (2021) An effective intrusion detection approach using svm with naïve bayes feature embedding. Comput Secur 103:102158
24. Hamamoto AH, Carvalho LF, Sampaio LDH, Abrão T., Proença ML Jr (2018) Network anomaly detection system using genetic algorithm and fuzzy logic. Expert Syst Appl 92:390–402
25. Hamdi M, Boudriga N (2007) Detecting denial-of-service attacks using the wavelet transform. Comput Commun 30:3203–3213
26. Helser S, Hwang MI (2021) Identity theft: a review of critical issues. Int J Cyber Res Educ (IJCRE) 3:65–77
27. Holm H (2014) Signature based intrusion detection for zero-day attacks:(not) a closed chapter? In: 2014 47th Hawaii international conference on system sciences. IEEE, pp 4895–4904
28. Horng S-J, Su M-Y, Chen Y-H, Kao T-W, Chen R-J, Lai J-L, Perkasa CD (2011) A novel intrusion detection system based on hierarchical clustering and support vector machines. Expert Syst Appl 38:306–313
29. Karami A (2018) An anomaly-based intrusion detection system in presence of benign outliers with visualization capabilities. Expert Syst Appl 108:36–60
30. Liao H-J, Lin C-HR, Lin Y-C, Tung K-Y (2013) Intrusion detection system: A comprehensive review. J Netw Comput Appl 36:16–24
31. Likas A, Vlassis N, Verbeek JJ (2003) The global k-means clustering algorithm. Pattern Recognit 36:451–461
32. Lin S-W, Ying K-C, Lee C-Y, Lee Z-J (2012) An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. Appl Soft Comput 12:3285–3290
33. Mantovani RG, Rossi AL, Vanschoren J, Bischl B, De Carvalho AC (2015) Effectiveness of random search in svm hyper-parameter tuning. In: 2015 international joint conference on neural networks (IJCNN). Ieee, pp 1–8
34. Manzoor MA, Morgan Y (2017) Network intrusion detection system using apache storm. Probe 4107:4166
35. Meidan Y, Bohadana M, Mathov Y, Mirsky Y, Shabtai A, Breitenbacher D, Elovici Y (2018) N-baiot—network-based detection of iot botnet attacks using deep autoencoders. IEEE Pervasive Comput 17:12–22

36. Meng W, Li W, Kwok L-F (2015) Design of intelligent knn-based alarm filter using knowledge-based alert verification in intrusion detection. Secur Commun Netw 8:3883–3895
37. Modi C, Patel D, Borisaniya B, Patel H, Patel A, Rajarajan M (2013) A survey of intrusion detection techniques in cloud. J Netw Comput Appl 36:42–57
38. Moustafa N, Slay J (2015) Unsw-nb15: A comprehensive data set for network intrusion detection systems (unsw-nb15 network data set). In: 2015 military communications and information systems conference (MilCIS). IEEE, pp 1–6
39. Muniyandi AP, Rajeswari R, Rajaram R (2012) Network anomaly detection by cascading k-means clustering and c4. 5 decision tree algorithm. Procedia Eng 30:174–182
40. Niaksu O (2015) Crisp data mining methodology extension for medical domain. Baltic J Modern Comput 3:92
41. Ozkan H, Ozkan F, Kozat SS (2015) Online anomaly detection under markov statistics with controllable type-i error. IEEE Trans Signal Process 64:1435–1445
42. Peddabachigari S, Abraham A, Grosan C, Thomas J (2007) Modeling intrusion detection system using hybrid intelligent systems. J Netw Comput Appl 30:114–132
43. Pérez D, Alonso S, Morán A, Prada MA, Fuertes JJ, Domínguez M (2019) Comparison of network intrusion detection performance using feature representation. In: International conference on engineering applications of neural networks. Springer, pp 463–475
44. Prasad R, Rohokale V (2020) Artificial intelligence and machine learning in cyber security. In: Cyber Security: The lifeline of information and communication technology. Springer, pp 231–247
45. Qatawneh M, Almobaideen W, AbuAlghanam O (2020) Challenges of blockchain technology in context internet of things: A survey. Int J Comput Appl 975:8887
46. Faris H, Castillo P, Merelo Guervós J, Al-Madi N (2018) The influence of input data standardization methods on the prediction accuracy of genetic programming generated classifiers. In: The 10th international joint conference on computational intelligence, pp 79–85. https://doi.org/10.5220/0006959000790085
47. Rahm E, Do HH (2000) Data cleaning: Problems and current approaches. IEEE Data Eng Bull 23:3–13
48. Rajakumari SB, Nalini C (2014) An efficient data mining dataset preparation using aggregation in relational database. Indian J Sci Technol 7:44
49. Ravale U, Marathe N, Padiya P (2015) Feature selection based hybrid anomaly intrusion detection system using k means and rbf kernel function. Procedia Comput Sci 45:428–435
50. Ren J, Guo J, Qian W, Yuan H, Hao X, Jingjing H (2019) Building an effective intrusion detection system by using hybrid data optimization based on machine learning algorithms. Secur Commun Netw
51. Revathi S, Malathi A (2013) A detailed analysis on nsl-kdd dataset using various machine learning techniques for intrusion detection. Int J Eng Res Technol (IJERT) 2:1848–1853
52. Roesch M et al (1999) Snort: Lightweight intrusion detection for networks. In: Lisa, vol 99, pp 229–238
53. Sanjaya SKSSS, Jena K (2014) A detail analysis on intrusion detection datasets. In: 2014 IEEE International Advance Computing Conference (IACC)
54. Scott SL (2004) A bayesian paradigm for designing intrusion detection systems. Comput Stat Data Anal 45:69–83
55. Shewale VR, Patil HD (2016) Performance evaluation of attack detection algorithms using improved hybrid ids with online captured data. Int J Comput Appl
56. Siddique K, Akhtar Z, Khan MA, Jung Y-H, Kim Y (2018) Developing an intrusion detection framework for high-speed big data networks: a comprehensive approach. KSII Trans Int Inform Syst 12
57. Tavallaee M, Bagheri E, Lu W, Ghorbani AA (2009) A detailed analysis of the kdd cup 99 data set. In: 2009 IEEE symposium on computational intelligence for security and defense applications. IEEE, pp 1–6
58. Thakral A, Rakesh N, Gupta A (2012) Area prone to cyber attacks. CSI Communications
59. Truong TC, Zelinka I, Plucar J, Čandík M, Šulc V (2020) Artificial intelligence and cybersecurity: Past, presence, and future. In: Artificial intelligence and evolutionary computations in engineering systems. Springer, pp 351–363
60. Van Der Maaten L, Postma E, Van den Herik J (2009) Dimensionality reduction: a comparative. J Mach Learn Res 10:13
61. Von Solms R, Van Niekerk J (2013) From information security to cyber security. Comput Secur 38:97–102
62. Wang G, Hao J, Ma J, Huang L (2010) A new approach to intrusion detection using artificial neural networks and fuzzy clustering. Exp Syst Appl 37:6225–6232
63. Wu W-J, Lin S-W, Moon WK (2012) Combining support vector machine with genetic algorithm to classify ultrasound breast tumor images. Comput Med Imaging Graph 36:627–633
64. Yao R, Wang N, Liu Z, Chen P, Sheng X (2021) Intrusion detection system in the advanced metering infrastructure: a cross-layer feature-fusion cnn-lstm-based approach. Sensors 21:626
65. Zhang Y, Lee W, Huang Y-A (2003) Intrusion detection techniques for mobile wireless networks. Wirel Netw 9:545–556
66. Zong B, Song Q, Min MR, Cheng W, Lumezanu C, Cho D, Chen H (2018) Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: ICLR