Arabian Journal for Science and Engineering (2023) 48:9849–9857
https://doi.org/10.1007/s13369-022-07421-0

RESEARCH ARTICLE-COMPUTER ENGINEERING AND COMPUTER SCIENCE

Intrusion Detection System Based on Pattern Recognition

Mohamed M. Abdeldayem[1,2]

Received: 24 April 2022 / Accepted: 24 October 2022 / Published online: 7 November 2022

Abstract
Artificial intelligence has been developed to be able to solve difficult problems that involve huge amounts of data and that require rapid decision-making in most branches of science and business. Machine learning is one of the most prominent areas of artificial intelligence, which has been used heavily in the last two decades in the field of network security, especially in Intrusion Detection Systems (IDS). Pattern recognition is a machine learning method applied in medical applications, image processing, and video processing. In this article, two layers' IDS is proposed. The first layer classifies the network connection according to the used service. Then, a minimum number of features that optimize the detection accuracy of malicious activities on that service are identified. Using those features, the second layer classifies each network connection as an attack or normal activity based on the pattern recognition method. In the training phase, two multivariate normal statistical models are created: the normal behavior model and the attack behavior model. In the testing and running phases, a maximum likelihood estimation function is used to classify a network connection into attack or normal activity using the two multivariate normal statistical models. The experimental results prove that the proposed IDS has superiority over related IDSs for network intrusion detection. Using only four features, it successfully achieves DR of 97.5%, 0.001 FAR, MCC 95.7%, and 99.8% overall accuracy.

Keywords Intrusion detection system (IDS) · Pattern recognition · Machine learning techniques · Network security

1 Introduction

In the era of the communication and information revolution, the use of the Internet has grown with a steady increase in various areas of life. In addition, the Internet of Things (IoT) and the precautionary precautions as a result of the COVID-19 pandemic have led to tremendous use of the Internet in various fields such as distance education, government transactions, banking transactions, entertainment, and electronic commerce. With the growth of Internet use, the risks increase and the need for security becomes urgent to build trust and protect the network assets. Therefore, cybersecurity always strives to protect these assets and to provide availability, confidentiality, authentication, and integrity of information. An IDS is a cybersecurity system that captures each packet that passes through the network. It provides a thorough inspection of the packet and then determines whether or not it exhibits abnormal behavior according to network packet features. IDSs can be categorized based on the detection process into misuse or anomalies. In misuse-based IDS, a database of known malicious behaviors is created, and each new packet is compared against this database to be identified as an attack or not. In the case of anomaly IDS, a model of normal behavior is constructed, and then, each new packet is examined to determine if it belongs to the model of normal behavior or not, and as a result, it can be identified as a normal or harmful activity. As Machine Learning (ML) has evolved, researchers provided training model datasets to detect anomalous malignant behaviors. ML can be categorized into three types of learning: supervised, semi-supervised, and unsupervised. The training datasets are developed by experts; therefore, supervised learning is the most accurate. Pattern recognition is one of the ML methods used in many applications. In this paper, a pattern recognition service-based IDS is proposed. It can be classified as supervised ML. Many related works are introduced in the next section. Section 3 explains the proposed IDS architecture, training phase, and testing phase. Section

✉ Mohamed M. Abdeldayem
mdayem@gmail.com

[1] Computers Science and Engineering Department, College of Applied Studies and Community Service, King Saud University, Riyadh, Saudi Arabia

[2] Information Technology Department, Faculty of Computers and Artificial Intelligence, Cairo University, Cairo, Egypt

4 presents IDS performance metrics; however, Sect. 5 discusses the experimental results based on these performance metrics. Section 6 concludes and summarizes the paper. In addition, future works are illustrated.

2 Related IDSs Works

Some of the related IDSs will be introduced in this section. Some MLs are designed based on the decision tree hierarchy [1–4], which is a simple supervised ML technique. Each leaf on the decision tree includes some rules; according to these rules, the data are classified down from the root to the final decision tree leaf until ranked. Decision trees have been used with success in several classification problems recently. For instance, a decision tree algorithm detects malicious domain in domain name systems [4]. Cyber-attacks are predicted based on a deep neural network [5]; in addition, many datasets are evaluated to conduct the benchmark. Self-Taught Learning method (STL) [6] proposed a framework for selecting features. Filter and wrapper features selection methodologies are used by J48 and Naïve Bayes ML classifiers to detect network anomalies [7]. In [8], an IDS using Bayesian network algorithms is proposed, and Cyber-attacks are sorted into stages; therefore, earlier stages of the attack can be detected. In addition, the behavior of the attack can be predicted, which helps in cyber-attack defense. The proposed IDS in [9] uses convolutional and recurrent neural network to find and classify attacks. The NSL-KDD data set is used to train and evaluate them. In [10], a proposed IDS based on convolutional neural network classifies network traffic into normal or malicious packets. It was trained and evaluated using CICIDS2017. The back-propagation neural network [11,12] is a supervised learning algorithm that propagates the error signal back to update the weight while reducing the error. It is popular in decision-making as it is easy to implement, as well as being accurate in forecasting and classification problems. During the training phase, the actual and target outputs are compared to update the weights. The Support Vector Machine (SVM) classifier sets the support vectors for feature separation in the hyperplane [13,14], and SVM is a supervised ML algorithm. In various applications, SVMs have produced successful results that include image analysis, biometrics and bioinformatics [13–15]. In [15], the SVM includes two phases: training and testing, in addition to radial basis function, which is the preferred kernel. The multi-class classification yields multiple binary classification cases based on two strategies: OvR-HMC and OvA-HMC. OvR-HMC in each layer, the class with the largest number, is identified as 1 and the remaining classes as 0. The previously classified samples were then removed. OvA-HMC specifies class and labels as 1 for each layer, with the other labeled as 0. Previously classified as abnormal, these samples were preserved. Moreover, other types of malicious behaviors were preserved by OvA-HMC, which makes the distribution learned from applying the techniques on the C-NSLKDD standard dataset similar to the learned distribution from applying the techniques on a real dataset. Consequently, OvA-HMC exceeds OvR-HMC and similar IDS techniques. An IDS is proposed to detect malicious behavior on HTTP network service [16]. First, it identifies the connection features that best classify HTTP packets, and then, it classifies network traffic into normal or attack behavior based on the Naive Bayes equation. It satisfies good IDS performance metrics with a small number of features compared to other IDSs. That is because it limits classification to a single network service rather than working on all network services; therefore, it prevents any feature contradiction that degrades attack detection on each service.

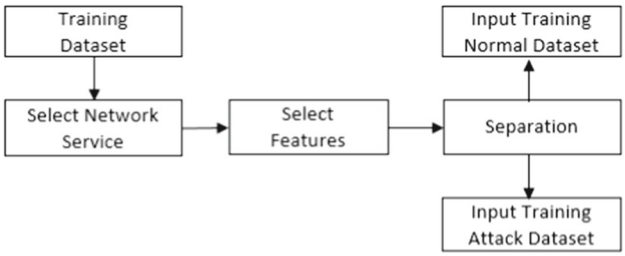

Fig. 1 Training dataset service classifier

3 The Proposed IDS

The proposed IDS consists of two classifiers: a service-based classifier and an intrusion detection classifier. In Subsect. (3.1), the IDS architecture is illustrated. The training phase is explained in Subsect. (3.2), while Subsect. (3.2) describes the testing phase.

3.1 The IDS Architecture

The IDS includes two classifications levels. The first level classifies the dataset based on the network services, such that each subset includes only data that are belonging to a specified network service (HTP, FTP, SMTP, etc.). Then, in the select features process, the target features of the service's subset are extracted. As illustrated in Fig. 1, the training data are classified based on network services, and the service training dataset is extracted based on a target features. Finally, in the separation process, two datasets are produced: Input Training Normal Dataset to model normal behavior and the Input Training Attack Dataset to model attack behavior.

The Input Testing dataset for each network service is obtained from the Testing dataset, and it includes only the

Fig. 2 Testing dataset service classifier

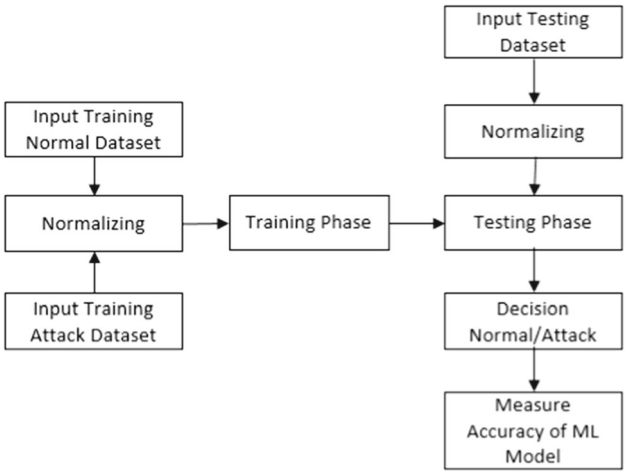

Fig. 3 Machine learning decision model

features that are selected in the Input Training dataset figure 2.

In the second classification level, Fig. 3 is based on supervised ML. Equation (1) is used to normalize the Input Training Normal Dataset and the Input Training Attack Dataset. Please note that in Fig. 3 Machine Learning Decision Model, the normalization process is implemented twice to scale each feature within the range of values [0,1]. The normalization process is used in the training phase to normalize the training dataset and during the testing phase to normalize the testing dataset.

$$\forall \boldsymbol{f}_j \in \{\boldsymbol{F}\}_k, \boldsymbol{f}'_{j,i} = \frac{\boldsymbol{f}_i}{\max\left(\boldsymbol{fn}_{\boldsymbol{j},\boldsymbol{i}}, \boldsymbol{fa}_{\boldsymbol{j},\boldsymbol{i}}\right)}, \quad \boldsymbol{f}'_{j,i} \in [0,1] \tag{1}$$

where

$\{\boldsymbol{F}\}_k$: the set of k features

$\boldsymbol{f}'_{j,i}$: value of ith feature of $\{F\}_k$ for j connection

$\boldsymbol{fn}_{\boldsymbol{j},\boldsymbol{i}}$: value of ith feature of normal dataset for j connection

$\boldsymbol{fa}_{j,i}$: value of ith feature of attack dataset for j connection

$\boldsymbol{f}'_{j,i}$:normalized value of ith feature for j connection

Based on the pattern recognition method, normal behavior and attack behavior models are created in the training phase. The purpose of the normal behavior model is to determine the probability that the connection is a normal connection. In contrast, the purpose of the attack behavior model is to calculate the probability that it is an attack. In the testing phase, the input testing dataset is normalized using equation (1); then, the behavior of each connection is compared with each of normal model and attack models; then, it is classified as attack or normal behavior based on the model that it belongs to. Finally, the accuracy and benchmark matrices are determined by comparing the testing decisions with the actual states of the testing connections.

3.2 Training Phase

In the training phase, the normal behavior of the service network connection is modeled using the Input Training Normal dataset. A multivariate normal distribution is identified and its covariance and mean vectors are determined to model the normal behavior. In addition, the Input Training Attack dataset is used to identify a multivariate normal distribution and to determine its covariance and mean vectors. This distribution models the attack behavior against the target network service. In the same manner as [17], let us define the normal and attack datasets that input into the training phase after normalization process as follows: Each of the normalized datasets (normal and attack) is within the real metric space RN. Each of them can be defined as:

$$\{f'\}_k = \left\{f'_1, f'_2, f'_3, \ldots, f'_j, \ldots, f'_{k-1}, f'_k\right\} \in R^N$$

$$f'_i = \left[f'_{i,1}, f'_{i,2}, f'_{i,3}, \ldots, f'_{i,N-1}, f'_{i,N}\right]^T; \forall i \in \{1, 2, 3, \ldots, k\}$$

where N is the number of connections and

k is the number of features.

Each of multivariate normal and attack datasets is presented as the following matrix (Eq. 2):

$$\{f'\}_k = \begin{bmatrix} \boldsymbol{f'_{1,1}} & \boldsymbol{f'_{2,1}} & \cdots & \boldsymbol{f'_{k,1}} \\ \boldsymbol{f'_{1,1}} & \boldsymbol{f'_{2,2}} & \cdots & \boldsymbol{f'_{k,2}} \\ \vdots & \vdots & \vdots\vdots & \vdots \\ \boldsymbol{f'_{1,N}} & \boldsymbol{f'_{2,N}} & \cdots & \boldsymbol{f'_{k,\mathrm{N}}} \end{bmatrix} \tag{2}$$

The multivariate normal probability density function is illustrated in Eq. 3 as follows [18]:

$$f(x) = \frac{1}{(2\pi)^{p/2} \mid \Sigma \mid^{1/2}} e^{-\frac{1}{2}(x-\mu)'} \Sigma^{-1}(x-\mu) \tag{3}$$

where $-\infty < x_j < \infty$, j=1,2, …, p

p is the number of variable, and the data vector is [x1, x2,xp];

Σ is the covariance p× p matrix.

To apply Eq. 3 to the proposed IDS, x will be replaced by the values of the features $\{f'\}_k$ and p will be replaced by k as shown in Eq. 4:

$$f\left(f'\right) = \frac{1}{(2\pi)^{k/2} \mid \Sigma \mid^{1/2}} e^{-\frac{1}{2}(f'-\mu)'\Sigma^{-1}\left(f'-\mu\right)} \tag{4}$$

where-$\infty < x_j < \infty, j = 1, 2, \ldots, k$

The mean vector of the sample, In addition, the sample covariance matrix can be used in Eq. (3) [18]. Therefore, both of them are used to find the normal distribution model. In the same way, the attack distribution model can be found using the multivariate attack dataset sample mean vector and covariance matrix. Let the mean vector and the covariance matrix for the multivariate normal dataset be $\overline{fn}$ (Eq. 5) and Sn (Eq. 6) in order, and the mean vector and the covariance matrix for the multivariate attack dataset be $\overline{fa}$ (Eq. 9), and Sa (Eq. 10) in order.

$$\overline{fn} = \frac{1}{N}\sum_{i=1}^{N} fn_i \tag{5}$$

$$Sn = \frac{1}{N-1}\sum_{i=1}^{N}\left(fn_i - \overline{fn}\right)\left(fn_i - \overline{fn}\right)' \tag{6}$$

The variances of the multivariate normal dataset on the main diagonal of Sn are sn_j^2 (Eq.7),

$$sn_j^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(fn_{ij} - \overline{fn}_j\right)^2 \tag{7}$$

The covariance of the multivariate normal dataset are sn_{jk} (eq.8),

$$sn_{jk} = \mid \frac{1}{N-1}\sum_{i=1}^{N}\left(fn_{ij} - \overline{fn}_j\right)\left(fn_{ik} - \overline{fn}_k\right) \tag{8}$$

$$\overline{fa} = \frac{1}{N}\sum_{i=1}^{N} fa_i \tag{9}$$

$$Sa = \frac{1}{N-1}\sum_{i=1}^{N}\left(fa_i - \overline{fa}\right)\left(fa_i - \overline{fa}\right)' \tag{10}$$

The variances of the multivariate attack dataset on the main diagonal of Sn are sn_j^2 (Eq. 11),

$$sn_j^2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(fn_{ij} - \overline{fn}_j\right)^2 \tag{11}$$

The covariance of the multivariate attack dataset are sa_{jk} (Eq. 12),

$$sa_{jk} = \frac{1}{N-1}\sum_{i=1}^{N}\left(fa_{ij} - \overline{fa}_j\right)\Big)\left(fa_{ik} - \overline{fa}_k\right) \tag{12}$$

3.3 Testing Phase

During testing phase, each connection in the normalized testing dataset is classified as normal or attack behavior. For each connection, the probability of it being a normal behavior or attack behavior is determined by calculating Maximum Likelihood Estimation (MLE) [19–23]. In order to find the most accurate unknown probability model parameters [23], MLE methods have advantages over alternate methods: they are fixed under the parameter conversion, they are simpler, they are functions of sufficient statistics, and in addition, they satisfy good convergence properties and are asymptotically unbiased with a large training dataset [22,23]. Let D be a training data set belonging to the probability density function $p(D \mid \theta)$. It is used to calculate the unknown vector θ. Assume D includes independent n samples (x1, x2,...., xn), the likelihood function can be estimated using Eq. (13) [23].

$$p(D \mid \theta) = \prod_{k=1}^{n} p\left(X_k \mid \theta\right) \tag{13}$$

The MLE of θ that maximize the likelihood function $p(D \mid \theta)$ is $\widehat{\theta}$. Because our model is based on multivariate normal distribution, the unknown parameter θ includes the distribution's mean vector μ and the covariance matrix Σ. To determine Σ we find $\widehat{\mu}$ and $\widehat{\Sigma}$, $\widehat{\mu}$ the MLE of μ and $\widehat{\Sigma}$ is MLE of Σ . Both of $\widehat{\mu}$ and $\widehat{\Sigma}$ will be estimated from the training dataset D (Eqs. 14 and 15) [6].

$$\widehat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k \tag{14}$$

$$\widehat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}\left(x_k - \widehat{\mu}\right)\left(x_k - \widehat{\mu}\right)^t \tag{15}$$

3.4 MLE for the Proposed Model

Both $\widehat{\mu}$ and $\widehat{\Sigma}$ will be estimated from the proposed training dataset; however, the proposed model includes the normal training dataset f' and the attack training dataset fa ; thus, $\hat{\theta}_1$ includes $\hat{\mu}_1$ and $\widehat{\sum}_1$ that will be estimated for normal behavior, and $\hat{\theta}_2$ includes $\hat{\mu}_2$ and $\widehat{\sum}_2$ for attack behavior. Let the normalized test dataset be F , the Eqs. 13, 14 and 15 will

be rewritten to be applied in the proposed model for each connection F_t in F as follows: For normal behavior:

$$\hat{\mu}_1 = \frac{1}{N}\sum_{i=1}^{N} f_i' = \overline{fn} \tag{16}$$

$$\widehat{\sum}_1 = \frac{1}{N-1}\sum_{i=1}^{N}\left(fn_{ij} - \overline{fn}_j\right)\left(fn_{ik} - \overline{fn}_k\right) = sn_{jk} \tag{17}$$

$$p_1\left(F_t \mid \hat{\theta}_1\right) = \prod_{i=1}^{k} p\left(F_i \mid \hat{\theta}_1\right) \tag{18}$$

For attack behavior:

$$\hat{\mu}_2 = \frac{1}{N}\sum_{i=1}^{N} fa_i = \overline{fa} \tag{19}$$

$$\widehat{\Sigma}_2 = \frac{1}{N-1}\sum_{i=1}^{N}\left(fa_{ij} - \overline{fa}_j\right)\left(fa_{ik} - \overline{fa}_k\right) = sa_{jk} \tag{20}$$

$$p_2\left(F_t \mid \hat{\theta}_2\right) = \prod_{i=1}^{k} p\left(F_i \mid \hat{\theta}_2\right) \tag{21}$$

$$p_1\left(F_t \mid \hat{\theta}_1\right) > p_2\left(F_t \mid \hat{\theta}_2\right) = \{\text{ yes,} \quad \text{Normal}$$
$$\text{Otherwise,} \quad \text{Attack} \tag{22}$$

(MLE)N is determined using the feature vector of the connection and the multivariate normal distribution of normal behavior (Eqs. 16 and 18). In addition, (MLE)A is determined using the feature vector of the connection and the multivariate normal distribution of attack behavior (Eqs. 19 and 21); then, the connection is classified as normal behavior or attack behavior based on Eq. 22.

4 Performance Metrics

After classifying all connections in the testing dataset, the performance metrics for the proposed IDS are determined. These include [24–30]: True Positive (TP), False Positive (FP), True Negative (TN), False Negative (FN), Attack Detection Rate (ADR), Accuracy (Acc), Mathew's correlation coefficient (Mcc), and False Alarm Rate (FAR). TP is the number of attacks that the IDS detected as attacks; FP is the number of attacks that the IDS did not detected as attacks; it classified them as normal instances; TN is the number of normal instances that the IDS detected as normal instances; FN is the number of normal instances that the IDS didn't detected as normal; it classified them as attacks. ADR is the ratio of the number of detected attacks to all true attacks. FAR is the ratio of the number of normal instances detected as attacks to the total number of true normal instances. Acc is the total number of true attacks and true normal instances detected by the system (TP and TN) across all data set samples. Mcc can be used as a measure of the quality of binary classification [30]. While the prediction is perfect if MCC = +1, MCC =-1 identifies the worst possible prediction. Table 1 shows how the connection can be classified as: TN, FN, TP or FP [10].

Table 1 Connections classifications

	Classified as normal	Classified as attack
Normal	TN	FP
Attack	FN	TP

$$ADR = \frac{TP}{TP + FN} \tag{23}$$

The ADR ratio is used to identify the IDS capability to detect attacks based on the features vector [24–30].

$$FAR = \frac{FP}{FP + TN} \tag{24}$$

FAR is the ratio of normal connections that are identified as attacks to all normal connections. If FAR is large, this indicates that the IDS is unreliable because it will generate too many false alarms [24–30]. The Acc represents ratio of accurate detected connections to all connections [27].

$$Acc = \frac{TP + TN}{TP + TN + FP + FN}$$
$$Mcc = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}} \tag{25}$$

5 Results

The Dell Inspiration 15, 5570 Series, 64-based PC, Core i7, 16 GB RAM, and 1 TB HD and 1 GB VGA are used to test the proposed IDS performance. Jupyter Notebook took 0.0588 seconds to train the data, while MATLAB Software took 2.4507 seconds. To evaluate IDS, an IDS dataset must be used in the training and testing phases. MIT Lincoln Labs published KDD-DARPA, which is the most famous standard IDS data set. This includes a wide range of intrusion connections. A KDD is used to test a large number of IDSs. Although it is old, it is still used to evaluate the capability of IDS. The KDD includes forty-one ordered features for each connection, and a table describing these ordered features is found in [31]. NSL-KDD was published as an enhanced version of

Table 2 NSL-KDD selected features

Index	Features
A	src-bytes, dst_bytes, hot, count, and srv_count
B	src-bytes, hot, count, and srv_count
C	src-bytes, dst_bytes, count, and srv_count
D	Duration, src-bytes, dst_bytes,hot,count, and srv _count
Bays_9f	Duration, src-bytes, dst_bytes,hot, count, and srv_count
Bays_8f	src-bytes, dst_bytes, count, and srv_count

Table 3 True/false alarms for each IDS

Index	TP	FN	TN	FP
A	307	4	6630	9
B	307	4	6626	13
C	306	5	6631	8
D	308	3	6630	9
Bays_9f	307	4	6631	8
Bays_8f	306	5	6630	9

Table 4 ADR, FAR, Acc, and Mcc

Index	ADR	FAR	Acc	Mcc
A	97.2%	0.1%	99.7%	95.0%
B	95.9%	0.2%	99.6%	93.1%
C	97.5%	0.1%	99.8%	95.3%
D	97.2%	0.1%	99.7%	95.1%
Bays_9f	97.5%	0.1%	99.8%	95.4%
Bays_8f	97.1%	0.1%	99.7%	95.6%

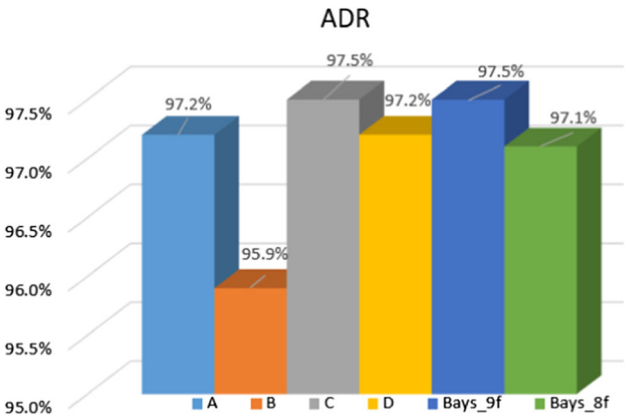

Fig. 4 ADR comparison

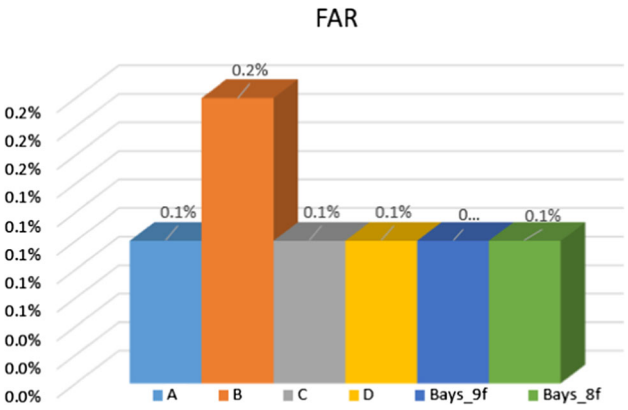

Fig. 5 FAR comparison

the KDD [32]. It eliminates some of the KDD's inconsistencies [32]. Therefore, I used NSL-KDD to test proposed IDS performance. This includes two level classifiers. The first level classifies the connection based on the three attribute features: Protocol type, service and flag (features numbers 2, 3 and 4 in NSL-KDD dataset); in addition, the second level classifies the connection based on some other features. In the second level, different combinations of features (A-D) are selected as shown in Table 2 In addition, it includes the group of features that are used in Bays_9f IDSs. MATLAB software [33] is used to test the performance of the proposed IDS, and Tables 3 and 4 illustrate the performance results for each combination of features compared to the results for Bayes_9F and Bayes_8f.

As shown in Table 3, there are a little difference in the results among all techniques. However, proposal D has the greatest TP alarms and the lowest FN alarms, and proposals C and Bayes_9F have the largest TN and the lowest FP. Table4illustrates the comparison of ADR, FAR, Acc, and Mcc for all proposed IDSs. Please keep in mind that all measurements are in percentages. As the ADR increases, the proposal C and Bays_9F have the same ADR (97.5 %), which is the largest value, while the proposal C has a slightly lower ADR value (97.2%), as shown in Fig. 4. The FAR for all IDSs has the same value (0.1%) except the proposal B has (0.2%) value (Fig. 5).

In Fig. 6, the largest value of Acc is 99.8% which is for the proposal C and Bayes_9F. The Mcc value for Bayes_8F is the highest, but it is similar to the Mcc values for Bayes_9F and proposal C, as shown in Fig. 7. We can conclude that the performance of proposal C and Bayes_9F is similar; however, the proposed IDS C uses lower number of features that can enhance the speed of implementations and detection rate. In addition, it is based on machine learning technique that accelerate the IDS, while Bayes_9F based on conditional probability which is more complex and may reduce the IDS detection rate.

The proposals A and C are coded using PyTorch [34] and other python library for machine learning, such as pandas [35] and tensorflow [36]. The IDS python code is implemented using Jupyter under Anaconda Navigator [37]. The experimental results compared with MATLAB implementation are presented in Tables 5, 6, and Fig. 8. As shown in Tables 5 and 6, and Fig. 8, there is no significant dif-

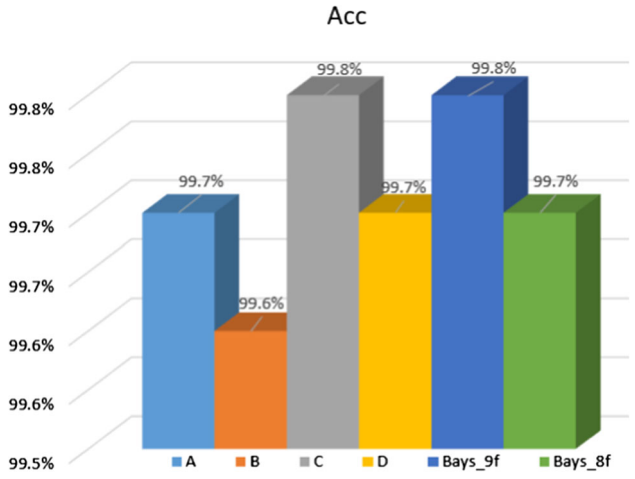

Fig. 6 Acc comparison

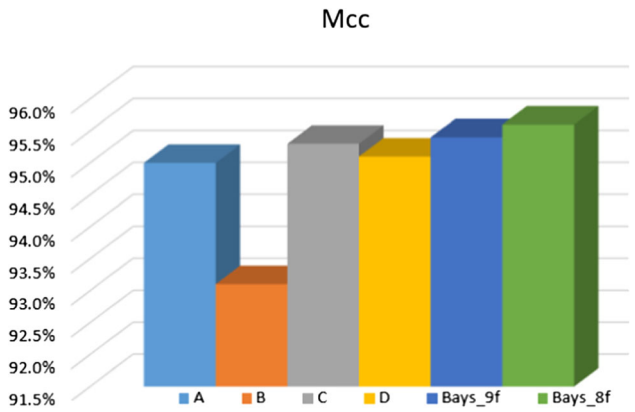

Fig. 7 Mcc comparison

Table 5 True/false experimental results

Index	TP	FN	TN	FP
A	307	4	6630	9
Python-A	409	28	6663	10
C	306	5	6631	8
Python-C	410	27	6665	8

ference between the two implementations. However, the MATLAB implementation has a little superiority over the Python implementation in terms of accurate results, but the Python implementation has a higher speed.

In [11], the performance of 10 IDS ML algorithms is measured using NSL-KDD data set. Table 7, and Figs. 9, 10 and 11 illustrate the performance of the proposed IDS compared to the 10 IDS algorithms. The proposed IDS has superiority over the other IDSs in terms of ACC, FAR, and DR. It achieves an ACC of 99.8%,a FAR of 0.001, a DR of 97.5%.

Table 6 Performance measurements

Index	ADR	FAR	Acc	Mcc
A	97.2%	0.1%	99.7%	95.0%
Python-A	93.6%	0.1%	99.5%	95.3%
C	97.5%	0.1%	99.8%	95.3%
Python-C	93.9%	0.2%	99.6%	95.7%

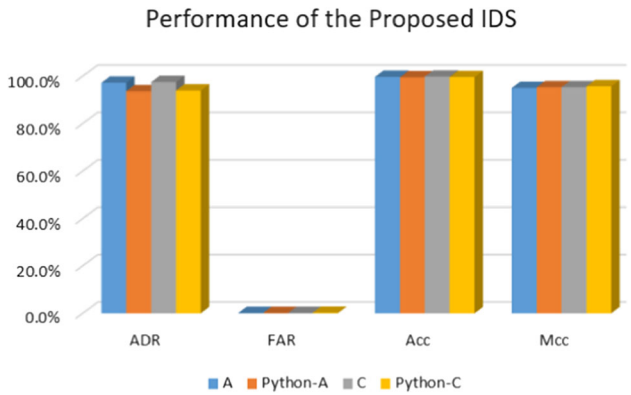

Fig. 8 Comparison MATLAB and Python implementation

Table 7 Performance measurements

IDSs	ACC (%)	FAR	DR (%)
Proposed	99.8	0.001	97.5
BAT-MLP	87.7	0.188	92.5
BBO-MLP	87.9	0.146	89.8
CS-MPL	86.82	0.192	91.4
DA-MLP	88.3	0.214	95.6
DE-MLP	84.8	0.213	89.5
EBAT-MLP	97.5	.022	97.2
GA-MLP	88.57	0.182	93.7
GSA-MLP	84.4	0.310	96.1
PSO-MLP	83.9	0.290	93.7

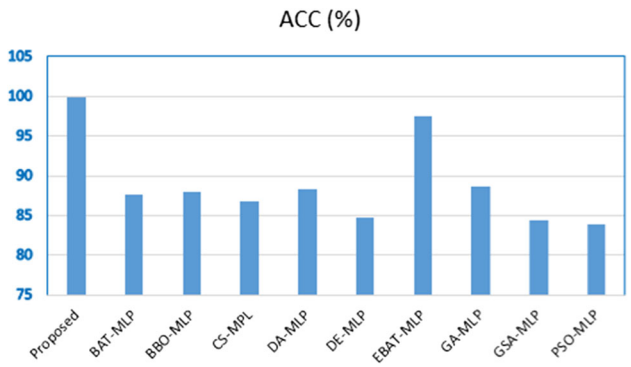

Fig. 9 Comparison of IDSs ACC

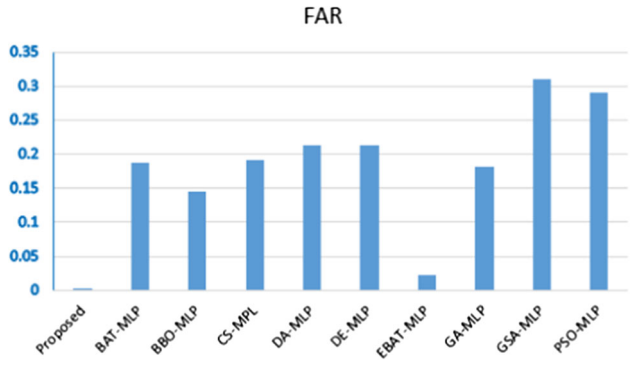

Fig. 10 Comparison of IDSs FAR

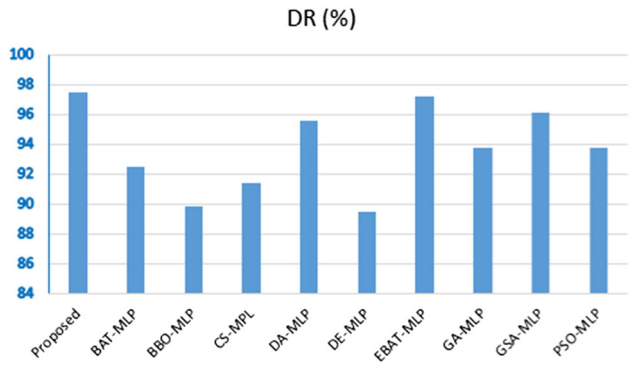

Fig. 11 Comparison of IDSs DR

6 Conclusion

Nowadays, machine learning is used in several applications. Pattern recognition is a machine learning method that is used in applications that require rapid decision-making. In this paper, a pattern recognition-based IDS is proposed, its structure and implementation are described, and its performance is tested on the NSL-KDD dataset. The proposed IDS consists of two levels of classification. The first identifies the required service for the network packet and accordingly selects the set of features that best classifies the packet. The second classifies the packet as normal or attacking based on the maximum likelihood estimation function that is applied to the selected features. The experimental results illustrated that the proposed intrusion detection system is superior to other intrusion detection systems in the matters of accuracy, DR, and FAR. Using only 4 features, it classifies the network packets with 99.8% accuracy, 97.5% DR, and 0.001 FAR. For future work, deep learning, the most recent machine learning techniques will be evaluated in network IDS. Other classifiers will be applied to the proposed IDS to achieve better detection rate; in addition, the proposed IDS will be applied on other network services and will be installed in a real-life network.

References

1. Yee Jian, C., Shih Yin, O., Kok-Seng, W., and Ying Han, P.: Decision tree with sensitive pruning in network-based intrusion detection system. In: *Computational Science and Technology*, pp. 1–10. Springer (2020)
2. Tony, T., Athira, V. P., Sabu, E.: Machine learning approaches in cyber security analytics. Springer (2020)
3. Chandrashekhar, A., Ashok Kumar, M., Vijay Kumar, J.: Evolutionary decision tree-based intrusion detection system. In: Proceedings of the third international conference on microelectronics, computing and communication systems , pp. 271–282. Springer (2019)
4. Ilham, R., Parman, S., and Muhammad, A. N.: Comparative analysis of k-nearest neighbor and decision tree in detecting distributed denial of service. In: 2020 8th international conference on information and communication technology (ICoICT)
5. Ravi, V.; Mamoun, A.; Soman, K.P.; Prabaharan, P.; Al-Nemrat, A.; Sitalakshmi, V.: Deep learning approach for intelligent intrusion detection system. IEEE Access **7**, 41525–41550 (2019)
6. Javaid, Ahmad; Niyaz, Quamar; Sun, Weiqing; Alam, Mansoor: A deep learning approach for network intrusion detection system. Eai Endors. Trans. Secur. Saf. **3**(9), e2 (2016)
7. Hebatallah, M. A., Mohamed, F., and Ayman, A.-H.: A framework for efficient network anomaly intrusion detection with features selection. In: 2018 9th international conference on information and communication systems (ICICS), pp. 157–162. IEEE (2018)
8. Pivarníková, M.; Sokol, P.; Bajtoš, T.: Early-stage detection of cyber attacks. Information **11**(12), 560 (2020)
9. Sara, A.-E., Aisha, A.-M., Felwa, A.-S.: Using deep learning techniques for network intrusion detection. In: *2020 IEEE international conference on informatics, IoT, and enabling technologies (ICIoT)*, pp. 171–176. IEEE (2020)
10. Samson, H.; Saleh, A.J.; Khalil, D.; Mohammad, M.: A novel intrusion detection model for detecting known and innovative cyberattacks using convolutional neural network. IEEE Open J. Comput. Soc. **2**, 14–25 (2021)
11. Waheed, A.H.M.G.; Aman, J.: A new approach for intrusion detection system based on training multilayer perceptron by using enhanced bat algorithm. Neural Comput. Appl. **32**(15), 11665–11698 (2020)
12. AHMG, Waheed; Aman, J.: An enhanced bat algorithm with mutation operator for numerical optimization problems. Neural Comput. Appl. **31**(1), 617–651 (2019)
13. Yunqian, M., Guodong, G.: Support vector machines applications, vol. 649. Springer (2014)
14. Ankita, S., Bhaswati, S., Siddharth, S. R., Manjusha, P.: Analysis of breast cancer dataset using big data algorithms for accuracy of diseases prediction. In: *International conference on computer networks and inventive communication technologies*, pp. 271–277. Springer (2019)
15. Hsiu-Min, C., Hui-Ying, H., Fanpyn, L., Chung-Hsien, T.: Classification of intrusion detection system based on machine learning. In: International cognitive cities conference, pp. 492–498. Springer (2019)
16. Abd-Eldayem, Mohamed M.: A proposed http service based ids. Egypt. Inf. J. **15**, 13–24 (2014)
17. Plamen A. P., Xiaowei, G.: Empirical approach to machine learning. Springer (2019)
18. Douglas, C.: Montgomery. *Introduction to statistical quality control*. Wiley (2020)
19. Ulisses, B.-N.: Fundamentals of Pattern Recognition and Machine Learning. Springer, Cham (2020)
20. Arcangelo, D.; Cosimo, D.; Wheeler, D.: Handbook of Image Processing and Computer Vision. Springer, Cham (2020)

21. Tsai, Ming-Tien.: On the maximum likelihood estimation of a covariance matrix. Math. Methods Stat. **27**(1), 71–82 (2018)
22. Ameet, J.V.: Machine Learning and Artificial Intelligence. Springer, Cham (2020)
23. Richard, R.J.: Mathematical Statistics: An Introduction to Likelihood Based Inference. Wiley, London (2018)
24. Zeeshan, A., Adnan, S. K., Cheah, W. S., Johari, A., and Farhan, A.: Network intrusion detection system: a systematic study of machine learning and deep learning approaches. Trans. Emerging Telecommun. Technol., 32(1):e4150 (2021)
25. Jecheva, V.; Nikolova, Evgeniya: Classification trees as a technique for creating anomaly-based intrusion detection systems. Serdica J. Comput. **3**(4), 335–358 (2009)
26. Natesan, P.; Balasubramanie, P.; Gowrison, G.: Improving attack detection rate in network intrusion detection using adaboost algorithm with multiple weak classifiers. J. Inf. Comput. Sci. **9**(8), 2239–2251 (2012)
27. Wang, M.; Zheng, K.; Yang, Y.; Wang, Xiujuan: An explainable machine learning framework for intrusion detection systems. IEEE Access **8**, 73127–73141 (2020)
28. Adriana-Cristina, E., Valentin, S.: An improved bat algorithm driven by support vector machines for intrusion detection. In: *Computational intelligence in security for information systems conference*, pp. 41–51. Springer (2015)
29. Mehdi, H.; Amir Masoud, R.; Bay, V.; Moazam, B.; Mohammad, M.; Mehran, Z.: Improving security using svm-based anomaly detection: issues and challenges. Soft Comput. **25**(4), 3195–3223 (2021)
30. Hanan, H.; David, B.; Ethan, B.; Amar Kumar, S.; Christos, T.; Robert, A.; Xavier, B.: A taxonomy of network threats and the effect of current datasets on intrusion detection systems. IEEE Access **8**, 104650–104675 (2020)
31. KDD Cup. Data (1999)http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (2018)
32. The nsl-kdd data set, https://www.unb.ca/cic/datasets/nsl.html. accessed 22 sep. (2022)
33. Matlab,https://www.mathworks.com/products/matlab.html.accessed 22 sep.(2022)
34. Pytorch, https://pytorch.org/. accessed 22 sep. (2022)
35. pandas, https://pandas.pydata.org/. accessed 22 sep. (2022)
36. Tensorflow, https://www.tensorflow.org/. accessed 22 sep. (2022)
37. Anaconda navigator, https://www.anaconda.com/products/individual. accessed 22 sep. (2022)