AUT Journal of Modeling and Simulation

AUT J. Model. Simul., 51(2) (2019) 211-220
DOI: 10.22060/miscj.2019.16555.5161

Intrusion Detection in IOT based Networks Using Double Discriminant Analysis

M. Imani

Faculty of Electrical and Computer Engineering, Tarbiat Modares University, Tehran, Iran

ABSTRACT: Intrusion detection is one of the main challenges in wireless systems especially in Internet of things (IOT) based networks. There are various attack types such as probe, denial of service, remote to local and user to root. In addition to the known attacks and malicious behaviors, there are various unknown attacks which some of them have similar behaviors with respect to each other or mimic the normal behavior. So, classification of connections in IOT based networks is a hard and challenging task. In this paper, an intrusion detection framework is proposed for classification of various attacks and separation of them from the normal connections. The double discriminant embedding (DDE) method is used to transform the original feature space of data. This transform is implemented in two steps. In the first step, the difference between the features is maximized; and in the second one, the difference between classes is increased. The extracted features not only have less overlapping with respect to each other and contain less redundant information but also they provide more separation between different classes. The extracted features are fed to the support vector machine (SVM) with polynomial kernel for classification. The experiments on NSL-KDD dataset have shown improvement of the SVM classifier when the DDE features are used.

Review History:
Received: 2019-06-14
Revised: 2019-09-19
Accepted: 2019-09-19
Available Online: 2019-12-01

Keywords:
Intrusion detection
support vector regression
double discriminant embedding
Internet of things

1. Introduction

1.1. Problem and related works

Internet of things (IOT) technologies have been developed in various industry sectors such as smart cities, social domains, healthcare and smart energy systems [1]. Providing security of these networks has a great importance. To manage the security intrusion, there are various mechanisms where one of the main mechanisms is intrusion detection. This approach can be studied from two main views [2]-[4]: supervised and unsupervised. In the supervised view, the pre-defined attack patterns or malicious activities are available as signatures of attack (intrusion class). In the unsupervised view, there is not any known signature or pattern of attack classes where these methods are called as anomaly detection. Each of the supervised intrusion detection methods or the unsupervised ones (anomaly detection) has itself advantages and disadvantages. The supervised intrusion detection methods outperform the anomaly detection approaches in identification of known attacks. But, their performance is decreased in dealing with unknown patterns. In contrast, the anomaly detection methods have weaker performance in detection of known patterns. They have better detection performance when dealing with anomalous patterns. Selection of supervised or unsupervised intrusion detection methods is corresponding to availability of training samples, type and severity of intrusion and security level of the network. In some cases, both methods can be used to provide the integrated advantages.

Different machine learning methods have been assessed for intrusion detection in [5]. Then, a method combining several classifiers has been designed for detection of one attack type in the KDD CUP 1999 dataset. Among various intrusion detection methods, some of them are different versions of decision trees. For example, a bagging boosting based on C5 decision trees has been introduced in [6] that was winner of KDD CUP 1999. Another decision tree based method has been utilized for composing the optimal decision forest in [7].

Decision tree is blended with genetic algorithm, as an evolutionary technique, for generation of detection rules [8]. The rules should be generated such that not only provide accurate decision about detection of all attacks but also be linguistically interpretable for human administrator. But, decision trees have two main challenges. First is the size of tree and second is discretization of continous features. To deal with the first problem, Dendron is used for reduction of detection rules. To handle the second problem, the equal-frequency discretization method is used. An unbiased and accurate decision tree is also provided by utilizing the genetic algorithm.

The random forest algorithm is utilized for detection of known patterns by using training samples and unknown anomalies through the outlier detection mechanisms [9].

*Corresponding author's email: maryam.imani@modares.ac.ir

At first, the intrusion patterns are automatically built over training samples by using the random forests. The intrusions are detected by matching the activities against the constructed patterns. Then, the novel and unknown patterns are detected by using the outlier detection approach through the random forests.

An evolutionary soft computing (ESC) intrusion detection system is proposed in [10]. At first, several neuro-fuzzy classifiers are used for activities classification. The classifiers outputs are given to a fuzzy inference system to make the final decision. A genetic algorithm is also employed to optimize the structure of fuzzy sets. Another soft computing based intrusion detection method is proposed in [11]. It is a rule based discovering system using the rough set theory. It exploits applicable association rules, rule selection and data reduction to improve detection accuracy in the developed intrusion detection system. Rough set as a powerful tool for dealing to uncertain and vague knowledge is used for large data set reduction that results in more efficient data mining for intrusion detection.

The deep belief network (DBN) has been used for intrusion detection in [12]. DBN has a multi-layer structure with some advantages such as pre-training and fine-tuning learning approach. DBN is able to extract deep features from the training sets. So, DBN has not difficulties of the traditional neural networks such as needing to a large labeled samples and easy to fall into local extremes. To optimum the number of hidden layers and also the number of neurons in the hidden layers, the genetic algorithm is applied.

In [13], an intrusion detection algorithm is proposed that is based on IOT feature extraction and deep migration learning model. The migration learning is appropriate to deal with inconsistent distribution of target data and source data. The introduced method in [13] tries to replace the model generation and parameter system with an automation module. In other words, it has proposed a self-learning deep migration learning method that is able to fit deep neural networks. The experiments have shown high detection rate and low false positive rate of the deep learning based method with respect to some other competitors.

A semi-supervised intrusion detection method has been introduced in [14] that combines the fuzzy c-mean clustering with active learning support vector machine (SVM). The fuzzy c-mean clustering allows each sample to belong to some clusters with assigning a membership degree related to each cluster. The traditional SVM is supervised where a sufficient labeled data is required for its training. But, due to the high cost of acquiring labeled data, the active learning is used in [14]. Active learning as a semi-supervised approach uses a small set of labeled samples beside a large set of unlabeled ones.

Some machine learning algorithms were compared for intrusion detection application in [15]. The assessed methods were decision tree, logistic regression, artificial neural network, random forest and SVM. The superior performance of the random forest classifier with respect to its competitors was reported. A survey on machine learning methods used in the intrusion detection systems is given in [16]. According to its conclusions, although several intelligent techniques can achieve better recognition rate but they yet have problems in false positive rate. Some other methods stabilized the false positive rate at the price of high computations and increasing the running time. A comprehensive investigation is also provided in [17].

The SVM classifier is used for intrusion detection in several works such as [18]-[20]. In [21], data at first is processed with a hierarchical clustering algorithm to provide abstracted and fewer training samples. In addition, a feature selection method is used to remove unimportant features from the training instances. Then, the selected features are given to the SVM classifier. Two-layer dimension reduction and two-tier classification (TDTC) is proposed for intrusion detection in [22]. TDTC uses two feature reduction approaches and two classifiers. Both of principal component analysis (PCA) and linear discriminant analysis (LDA) [23] are used for feature extraction. Then, the Bayes classifier and an extended version of the nearest neighbor classifier are used for attacks classification. The main advantage of TDTC algorithm is detection of low frequency attacks such as R2L and U2R. Although the detection accuracy in high frequency attacks such as probe and Dos and also in normal class is a bit decreased, but TDTC has significant success in detection of anomalous hard-to-detect intrusions.

1.2.Motivation and novelties

The PCA transform is a popular feature reduction method. But, it does not consider the separation among classes. So, it may have not desired performance in classification applications. In contrast, methods such as LDA and different versions of it are appropriate choices for feature reduction in classification applications by maximizing the between-class scatters and minimizing the within-class scatters. But, they have two main difficulties. First, they can extract maximum $c-1$ features where c indicates the number of classes. Second, due to calculation of the scatter matrices, they have good efficiency just when a sufficient number of training samples is available, and they fail when training set is small. To deal with these difficulties, the double discriminant embedding (DDE) method has proposed in [24] for feature reduction of hyperspectral images. DDE not only maximizes the differences between classes, which simplifies distinguishing between classes but also extracts the most informative features with minimum redundancy. DDE is used for extraction of features from IOT connections, in the intrusion detection application, for the first time in this work,. The features extracted by DDE are given to the SVM classifier for intrusion detection and attacks classification. The experiments on a popular dataset have shown the good performance of the proposed framework. Some new contributions of the proposed framework are represented as follows:

1-The DDE method increases the difference between various features such as length of connection, type of protocol, destination service and status of connection. The

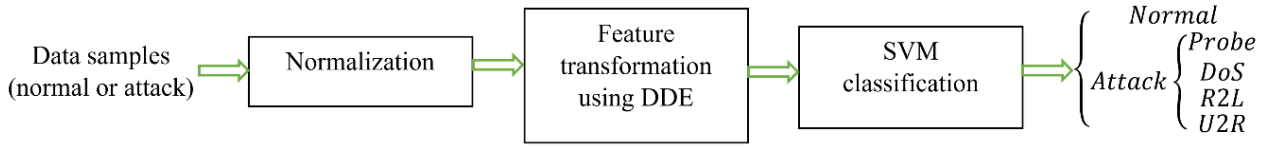

Fig. 1. Proposed framework for intrusion detection.

result is decrease of redundancy among extracted features in the projected feature space that yields lower false alarm rate.

2-By using the DDE transformation, the distance between normal class and attack classes containing different types of intrusions such as probe, denial of service, remote to local and user to root is increased. So, a good separation among normal and intrusion connections is provided.

2.Intrusion detection framework

There are various attacks or malicious behaviors in the IOT based networks. To increase the differences between normal and attack behaviors, a feature transformation method is proposed in this work. The proposed intrusion detection system is shown in Fig. 1. There are three main steps for intrusion detection in this framework, which are explained with more details in the following. At first, data samples are normalized. Then, the DDE features are extracted and then, the extracted features are given to the SVM module for classification. The result is assigning a label of normal or one type of known attacks to each connection sample.

2.1. Normalization

To improve the efficiency of feature transformation and classification modules, each of n_f features is normalized along with all N samples according to:

$$\boldsymbol{x}_i = \frac{\tilde{\boldsymbol{x}}_i - \min\left(\tilde{\boldsymbol{x}}_i\right)}{\max\left(\tilde{\boldsymbol{x}}_i\right) - \min\left(\tilde{\boldsymbol{x}}_i\right)} \times 99 + 1; \quad i = 1,2,\ldots,n_f \tag{1}$$

where $\tilde{\boldsymbol{x}}_i$ is the vector containing i th feature of N available samples and $\boldsymbol{x}_i$ is the normalized version of it that has positive values in [1,100]. This feature range is also selected in [22].

2.2. Feature transformation using DDE

DDE is chosen for feature extraction in this step. DDE has some advantages:

1) It increases the distances among n_f features. The result is features that contain the minimum overlapping and redundant information. This characterization allows a feature reduction with more reliability because the most informative features are chosen and the redundant ones are discarded.

2) DDE increases the differences among various classes. So, the difference between normal and attack classes is maximized. Therefore, they can better separated, i.e., intrusions are detected with more accuracy.

3) In contrast to feature transformation methods such as LDA that can extract maximum $c-1$ features, where c denotes the number of classes, DDE can extract any number of desired features.

4) DDE just uses the first order statistics (mean vectors). It does not need to estimate the second order statistics such as covariance matrix. So, it has good efficiency when a limited number of training samples is available.

Suppose the mean matrix of c classes with n_f features is given by [24]:

$$\begin{bmatrix} m_{11}\, m_{12}\, m_{13} \cdots m_{1c} \\ m_{21}\, m_{22}\, m_{23} \cdots m_{2c} \\ m_{31}\, m_{32}\, m_{33} \cdots m_{3c} \\ \vdots \quad \vdots \quad \vdots \quad \ldots \quad \vdots \\ m_{n_f 1}\, m_{n_f 2}\, m_{n_f 3} \cdots m_{n_f c} \end{bmatrix} \tag{2}$$

where $m_{ij}\left(i = 1,2,\ldots,n_f; j = 1,2,\ldots,c\right)$ represents the mean of class j in i th dimension. Corresponding to each row of the above matrix, a vector $\boldsymbol{h}_i$ can be considered:

$$\boldsymbol{h}_i = \begin{bmatrix} m_{i1} & m_{i2} & \cdots & m_{ic} \end{bmatrix}^T, \quad i = 1,2,\ldots,n_f \tag{3}$$

Maximizing the differences between features can be equal to maximizing differences between $\boldsymbol{h}_i\left(i = 1,2,\ldots,n_f\right)$ vectors. To this transformation, the projection matrix $\boldsymbol{A}_1$ is used:

$$\left(\boldsymbol{g}_i\right)_{c\times 1} = \left(\boldsymbol{A}_1\right)_{c\times c}\left(\boldsymbol{h}_i\right)_{c\times 1}, \quad i = 1,2,\ldots,n_f \tag{4}$$

The entries of matrix $\boldsymbol{A}_1$ can be obtained by defining the cost function ψ_1 and maximizing it:

$$\psi_1 = \sum_{j=1}^{n_f}\sum_{i=1}^{n_f} \left\|\boldsymbol{g}_i - \boldsymbol{g}_j\right\|^2 (w_1)_{ij} \tag{5}$$

where

$$(w_1)_{ij} = \left(\left\|\boldsymbol{h}_i - \boldsymbol{h}_j\right\|^2\right)^{-1} \quad ; i = 1,2,\ldots,n_f; j = 1,2,\ldots,n_f \tag{6}$$

To obtain the matrix form of the above optimization function, i.e.,

$$\boldsymbol{G}_{c\times n_f} = \left(\boldsymbol{A}_1\right)_{c\times c}\boldsymbol{H}_{c\times n_f} \tag{7}$$

the following matrices are defined:

$$\boldsymbol{H}=\left[\boldsymbol{h}_1,\boldsymbol{h}_2,\ldots,\boldsymbol{h}_{n_f}\right] \tag{8}$$

$$\boldsymbol{G}=\left[\boldsymbol{g}_1,\boldsymbol{g}_2,\ldots,\boldsymbol{g}_{n_f}\right] \tag{9}$$

The optimization function in (5) is rewritten as follows:

$$\begin{aligned}\psi_1 &= tr\left(\boldsymbol{G}\left(\boldsymbol{D}_1-\boldsymbol{W}_1\right)\boldsymbol{G}^T\right)\\ &= tr\left(\boldsymbol{A}_1\boldsymbol{H}\boldsymbol{L}_1\boldsymbol{H}^T\boldsymbol{A}_1^T\right)\end{aligned} \tag{10}$$

where $\boldsymbol{L}_1=\boldsymbol{D}_1-\boldsymbol{W}_1$ and $\boldsymbol{D}_1$ is a diagonal matrix with $(\boldsymbol{D}_1)_{ii}=\sum_{j=1}^{n_c}(w_1)_{ij}$. c eigenvalues of $\boldsymbol{H}\boldsymbol{L}_1\boldsymbol{H}^T$ are computed and sorted descending. The associated eigenvectors of $\boldsymbol{H}\boldsymbol{L}_1\boldsymbol{H}^T$ make the projection matrix $\boldsymbol{A}_1$. After applying the first projection on data, the mean matrix in (2) is transformed to:

$$\begin{bmatrix} \acute{m}_{11} & \acute{m}_{12} & \acute{m}_{13} & \cdots & \acute{m}_{1c}\\ \acute{m}_{21} & \acute{m}_{22} & \acute{m}_{23} & \cdots & \acute{m}_{2c}\\ \acute{m}_{31} & \acute{m}_{32} & \acute{m}_{33} & \cdots & \acute{m}_{3c}\\ \vdots & \vdots & \vdots & \cdots & \vdots\\ \acute{m}_{n_f1} & \acute{m}_{nf2} & \acute{m}_{nf3} & \cdots & \acute{m}_{nfc}\end{bmatrix} \tag{11}$$

where by considering $\acute{\boldsymbol{m}}_i=[\acute{m}_{1i}\ \ \acute{m}_{2i}\ \ \cdots\ \ \acute{m}_{n_fi}]^T,\ i=1,2,\ldots,c$, the distances among $\acute{\boldsymbol{m}}_i;i=1,2,\ldots,c$ are maximized to provide the projection matrix $\boldsymbol{A}_2$:

$$(\boldsymbol{r}_i)_{n_f\times 1}=(\boldsymbol{A}_2)_{n_f\times n_f}(\acute{\boldsymbol{m}}_i)_{n_f\times 1} \tag{12}$$

By defining the cost function ψ_2 and maximizing it, the entries of $\boldsymbol{A}_2$ are computed:

$$\psi_2=\sum_{j=1}^{c}\sum_{i=1}^{c}\left\|\boldsymbol{r}_i-\boldsymbol{r}_j\right\|^2(w_2)_{ij} \tag{13}$$

where $(w_2)_{ij}\,(i=1,2,\ldots,c;\ j=1,2,\ldots,c)$ are the entries of matrix $\boldsymbol{W}_2$:

$$(w_2)_{ij}=\left(\left\|\acute{\boldsymbol{m}}_i-\acute{\boldsymbol{m}}_j\right\|^2\right)^{-1};i=1,2,\ldots,c;\ j=1,2,\ldots,c \tag{14}$$

To obtain the matrix form of the above optimization problems, i.e.,

$$\boldsymbol{R}_{n_f\times c}=\left(\boldsymbol{A}_2\right)_{n_f\times n_f}\acute{\boldsymbol{M}}_{n_f\times c} \tag{15}$$

the following matrices are defined:

$$\acute{\boldsymbol{M}}=[\acute{\boldsymbol{m}}_1,\acute{\boldsymbol{m}}_2,\ldots,\acute{\boldsymbol{m}}_c] \tag{16}$$

$$\boldsymbol{R}=\left[\boldsymbol{r}_1,\boldsymbol{r}_2,\ldots,\boldsymbol{r}_c\right] \tag{17}$$

The function ψ_2 is rewritten:

$$\begin{aligned}\psi_2 &= tr\left(\boldsymbol{R}\left(\boldsymbol{D}_2-\boldsymbol{W}_2\right)\boldsymbol{R}^T\right)\\ &= tr\left(\boldsymbol{A}_2\acute{\boldsymbol{M}}\boldsymbol{L}_2\acute{\boldsymbol{M}}^T\boldsymbol{A}_2{}^T\right)\end{aligned} \tag{18}$$

where $\boldsymbol{L}_2=\boldsymbol{D}_2-\boldsymbol{W}_2$ and $\boldsymbol{D}_2$ denotes the diagonal matrix with $(\boldsymbol{D}_2)_{ii}=\sum_{j=1}^{c}(w_2)_{ij}$. For extraction of $n_{f,new}$ features from n_f original ones, $n_{f,new}$ eigenvectors of $\acute{\boldsymbol{M}}\boldsymbol{L}_2\acute{\boldsymbol{M}}^T$, corresponding to the $n_{f,new}$ largest eigenvalues compose the projection matrix $\boldsymbol{A}_2$.

2.3.Classification using SVM

The SVM classifier provides a hyperplane with the following discriminant function for separating each pair of classes [25]-[26]:

$$f\left(\boldsymbol{x}_i\right)=\boldsymbol{w}.\boldsymbol{x}_i+b \tag{19}$$

where $(\boldsymbol{x}_i)_{n_f\times 1}\,(i=1,\ldots,N)$ is i th training sample and $\boldsymbol{w}_{n_f\times 1}$ is the weight vector that is normal to hyperplane and b denotes the bias term. To separate each pair of the given classes, we should have:

$$y_i\left(\boldsymbol{w}.\boldsymbol{x}_i+b\right)\geq 1,\ i=1,\ldots,N \tag{20}$$

where y_i is the class label of i th training sample. The hyperplane, which maximizes margin among classes, is found where the margin is equal to $\frac{1}{\|\mathbf{w}\|}$. To this end, the following optimization problem should be solved [27]:

$$\begin{aligned}&\min\frac{1}{2}\|\mathbf{w}\|^2\\ &\text{s.t. } y_i\left(\mathbf{w}.\mathbf{x}_i+b\right)\geq 1,\ i=1,\ldots,N\end{aligned} \tag{21}$$

The Lagrange multipliers method is used for solving the above optimization problem. The solution, i.e., the discriminant function or the optimal hyperplane is given by:

$$f\left(x\right)=\sum_{i\in T}\alpha_i y_i\left(\mathbf{x}_i.\mathbf{x}\right)+b \tag{22}$$

where α_i represents the Lagrange multiplier and T denotes the subset of training samples associated with the nonzero Lagrange multipliers, i.e., support vectors. The samples are mapped to a higher feature space through a mapping function and by using the kernel trick. The discriminant function of the nonlinear SVM classifier can be obtained by:

$$f\left(\boldsymbol{x}\right)=\sum_{i\in T}\alpha_i y_i K\left(\boldsymbol{x}_i,\boldsymbol{x}\right)+b \tag{23}$$

Table 1. Features description of NSL-KDD dataset [30].

Index	Feature name	description	type
1.	Duration	Length of connection	Continuous
2.	Protocol type	Type of protocol (TCP, UDP and ICMP)	Symbolic
3.	Service	Destination service (HTTP, FTP, Telnet and so on)	Symbolic
4.	Flag	Status of connection	Symbolic
5.	Src-bytes	No. of Bytes from source to destination	Continuous
6.	Dst-bytes	No. of Bytes from destination to source	Continuous
7.	Land	If the source and destination address are the same land=1/ if not, then 0	Symbolic
8.	Wrong-fragment	No. of wrong fragments	Continuous
9.	Urgent	No. of urgent packets	Continuous
10.	Hot	No. of hot indicators	Continuous
11.	Num-failed-logins	No. of unsuccessful attempts at login	Continuous
12.	Logged-in	If logged in=1/ if login failed 0	Symbolic
13.	Num-compromised	No. of compromised states	Continuous
14.	Root-shell	If a command interpreter with a root account is running root shell=1/ if not, then 0	Continuous
15.	Su-attempted	If an su command was attempted su attempted=1/ if not, then 0 (temporary login to the system with other user credentials)	Continuous
16.	Num-root	No. of root accesses	Continuous
17.	Num-file-creations	No. of operations that create new files	Continuous
18.	Num-shells	No. of active command interpreters	Continuous
19.	Num-access-files	No. of file creation operations	Continuous
20.	Num-outbound-cmds	No. of outbound commands in an ftp session	Continuous
21.	Is-host-login	is host login=1 if the login is on the host login list/ if not, then 0	Symbolic
22.	Is-guest-login	If a guest is logged into the system, is guest login=1/ if not, then 0	Symbolic
23.	Count	No. of connections to the same host as the current connection at a given interval	Continuous
24.	Srv-count	No. of connections to the same service as the current connection at a given interval	Continuous
25.	Serror-rate	% of connections with SYN errors	Continuous
26.	Srv-serror-rate	% of connections with SYN errors	Continuous
27.	Rerror-rate	% of connections with REJ errors	Continuous
28.	Srv-rerror-rate	% of connections with REJ errors	Continuous
29.	Same-srv-rate	% of connections to the same service	Continuous
30.	Diff-srv-rate	% of connections to different services	Continuous
31.	Srv-diff-host-rate	% of connections to different hosts	Continuous
32.	Dst-host-count	No. of connections to the same destination	Continuous
33.	Dst-host-srv-count	No. of connections to the same destination that use the same service	Continuous
34.	Dst-host-same-srv-rate	% of connections to the same destination that use the same service	Continuous
35.	Dst-host-diff-srv-rate	% of connections to different hosts on the same system	Continuous
36.	Dst-host-same-src-port-rate	% of connections to a system with the same source port	Continuous
37.	Dst-host-srv-diff-host-rate	% of connections to the same service coming from different hosts	Continuous
38.	Dst-host-serror-rate	% of connections to a host with an S0 error	Continuous
39.	Dst-host-srv-serror-rate	% of connections to a host and specified service with an S0 error	Continuous
40.	Dst-host-rerror-rate	% of connections to a host with an RST error	Continuous
41.	Dst-host-srv-rerror-rate	% of connections to a host and specified service with an RST error	Continuous

where $K(\boldsymbol{x}_i, \boldsymbol{x}) = \langle \Phi(\boldsymbol{x}_i), \Phi(\boldsymbol{x}) \rangle$ is the kernel function that $\boldsymbol{a}, \boldsymbol{b}$ denotes the inner product of two vectors $\boldsymbol{a}, \boldsymbol{b}$.

3.Experiments and Findings

The performance of the proposed intrusion detection framework is assessed in this section. The used dataset is NSL-KDD [28] which is introduced to solve some problems of the KDD Cup '99 dataset [29]. The KDD Cup '99 dataset is an intrusion detection benchmark containing examples of normal and attack connections. One of the main problems in the KDD dataset is the huge volume of redundant samples, which bias the learning algorithms towards the frequent samples. This dataset has 41 features reported in Table 1 [30]. All attacks (malicious behaviors) are classified into one of four

Table 2. Attacks categorization in NSL-KDD dataset [22].

Attack category	Sub Class of Attacks in training set	New Subclass of Attacks in testing set
Probe	ftp write, guess passwd, imap, multhop, phf, spy, warezclient, warezmaster	Mscan, Saint
DoS	back, land, neptune, pod, smurf, teardrop	Apache2, Mailbomb, Processtable
Remote-to-Local (R2L)	ipsweep, nmap, portsweep, satan	Sendmail, Named, Snmpgetattack, Snmpguess, Xlock, Xsnoop, Worm
User-to-Root (U2R)	Buffer overflow, perl, loadmodule, rootkit.	Httptunnel, Ps, Sqlattack, Xterm

Table 3. Number of samples in each class of the training and testing data.

Category	Main Class	Number of samples in training data	Number of samples in testing data
Attack	Probe	11656	1106
	DoS	45927	6480
	Remote-to-Local (R2L)	995	2199
	User-to-Root (U2R)	52	37
	Unknown attacks in testing data	--	2861
Normal	Normal	67343	9711

Table 4. Numeralization of symbolic attributes in NSL-KDD dataset [32].

Symbolic features	Numeralization
Protocol Type	tcp=1, udp=2, icmp=3
Service value	Private = 1 ; ftp_data = 2 ; eco_i = 3 ; telnet = 4 ; http = 5 smtp = 6 ; ftp = 7 ; ldap = 8 ; pop_3 = 9 ; courier = 10 discard = 11 ; ecr_i = 12 ; imap4 = 13 ; domain_u = 14 mtp = 15 ; systat = 16 ; iso_tsap = 17 ; other = 18 ; csnet_ns = 19 ; finger = 20 ; uucp = 21 ; whois = 22 netbios_ns = 23 ; link = 24 ; Z39_50 = 25 ; sunrpc = 26 auth = 27 ; netbios_dgm = 28 ; uucp_path = 29 ; vmnet = 30 ; domain = 31 ; name = 32 ; pop_2 = 33 http_443 = 34 ; urp_i = 35 ; login = 36 ; gopher = 37 exec = 38 ; time = 39 ; remote_job = 40 ; ssh = 41 kshel l = 42 ; sql_net = 43 ; shell = 44 ; hostnames = 45 echo = 46 ; daytime = 47 ; pm_dump = 48 ; IRC = 49 netstat = 50 ; ctf = 51 ; nntp = 52 ; netbios_ssn = 53 tim_i = 54 ; supdup = 55 ; bgp = 56 ; nnsp = 57 ; rje=58 printer = 59 ; efs = 60; X11 = 61 ; ntp_u = 62 ; klogin = 63 tftp_u = 64 ; red_i = 65 ; urh_i = 66 ; http_8001 = 67 aol =68 ; http_2784 = 69 ; harvest = 70
Flag Value	REJ = 1 ; SF = 2 ; RSTO = 3 ; S0 = 4 ; RSTR = 5 ; SH = 6 S3 = 7 ; S2 = 8 ; S1 = 9 ; RSTOS0 = 10 ; OTH = 11
Classification attack	Probe=1 DOS=2 R2L=3 U2R=4 Normal=0

main categories:

1)Probe: the information of networks is probed through scanning ports and host activities.

2)Denial of service (DoS): the access of legitimate users to the given machine or service is interrupted.

3)Remote to local (R2L): attacker imitates the behavior of local users to gain remote access to a sacrificed machine.

4)User to root (U2R): the limited access of a user is escalated to a root access like a super user by applying stolen credentials or malware infection.

Among four above attacks, detection of R2L and U2R are the hardest tasks because attacker mimics the behavior of legal or normal user [31]. This categorization is represented in Table 2 [22]. The number of samples in each class in the training and testing data is also given in Table 3. There are several symbolic features in the dataset that have to be converted into the numerical values for processing and analysis. As examples of the symbolic features, we can refer to the protocol type (TCP, UDP and ICMP) and service type (HTTP, FTP, Telnet, …). The values of the symbolic attributes are replaced by the numeric values as shown in Table 4 [32].

To evaluate the performance of the intrusion detection system, the detection accuracy of normal and attack classes, detection rate (DR) and false (positive) alarm rate (FAR) are used.

DR is a measure of correctly detection of attack samples with respect to all attack ones. FAR is a measure of wrongly detecting the normal samples as attack of all normal samples. DR and FAR are computed by using the following performance indicators:

- True Positive (TP): the number of attack samples that are correctly detected.
- True Negative (TN): the number of normal samples that are correctly classified.
- False Positive (FP): the number of normal samples that are falsely classified as attack.
- False Negative (FN): the number of attack samples that are falsely classified as normal.

DR and FAR are computed by [33]:

Table 5. Classification accuracy of normal and different attack classes.

Features	Probe	DoS	R2L	U2R	Normal
41 original features	85.62	87.99	**25.60**	8.11	90.10
5 distance features	1.54	39.37	22.06	**13.51**	63.08
10 DDE extracted features	73.24	53.33	16.73	10.81	63.84
41 DDE extracted features	76.31	81.91	6.28	8.11	**97.48**
5 distance features +10 DDE extracted features	76.49	79.18	8.05	**13.51**	97.06
5 distance features+41 DDE features	83.00	**92.15**	16.64	2.70	96.18
41 original features +10 DDE extracted features	**98.64**	86.33	11.96	**13.51**	95.58
41 original features +41 DDE extracted features	78.66	86.08	15.14	5.41	97.18

Table 6. Detection rate and false alarm rate of different feature sets.

Features	Detection rate (DR)	False alarm rate (FAR)
41 original features	66.43	9.90
5 distance features	31.56	36.92
10 DDE extracted features	51.03	36.16
41 DDE extracted features	59.22	**2.52**
5 distance features +10 DDE extracted features	61.48	2.94
5 distance features+41 DDE features	**68.21**	3.82
41 original features +10 DDE extracted features	66.39	4.42
41 original features +41 DDE extracted features	60.73	2.82

Table 7. Comparison with other methods

Method	Probe	DoS	R2L	U2R	Normal	Detection rate (DR)	False alarm rate (FAR)
Bagging boosting based on C5 decision trees (2000)	83.3	97.1	8.4	13.2	**99.5**	N/A	N/A
Decision forest (2000)	84.5	**97.5**	7.3	11.8	99.4	N/A	N/A
Combined classifiers (2003)	88.7	97.3	9.6	29.8	n/r	N/A	N/A
Rules based rough set theory (2006)	74.9	96.8	7.9	3.8	**99.5**	N/A	N/A
Two-layer Dimension Reduction and Two-tier Classification (TDTC) (2019)	87.32	88.20	**42**	**70.15**	94.43	**84.86**	4.86
Proposed (DDE features)	**98.64**	92.15	16.73	13.51	97.48	68.21	**2.52**

$$DR = \frac{TP}{FN + TP} \tag{24}$$

$$FAR = \frac{FP}{FP + TN} \tag{25}$$

The polynomial kernel with default parameters of LIBSVM is used for implementation of SVM classifier [27]. The data is normalized before giving to the feature extraction module, as explained before. Three types of features are experimented as input of the SVM classifier:

1) 41 original features of dataset (represented in Table 1).

2) 5 distance features. Each distance feature is defined as norm of the distance between each sample to the mean of 5 available classes (probe, DoS, R2L, U2R and normal):

$$d_i = \text{norm}\left(\boldsymbol{x}_i - \boldsymbol{m}_k\right); k = 1,2,3,4,5 \tag{26}$$

where d_i denotes the distance feature of sample $\boldsymbol{x}_i$, defined as norm of the difference between $\boldsymbol{x}_i$ and $\boldsymbol{m}_k$. In this formula, $\boldsymbol{m}_k$ is the mean vector of class k and $\text{norm}(\boldsymbol{a})$ is the Euclidean norm of vector $\boldsymbol{a}$.

3) $n_{f,new}$ features extracted by the DDE method. $n_{f,new} = 10,41$ are experimented in this work.

The classification accuracy of different classes are reported in Table 5. In addition, the detection rate and false alarm rate of different feature sets are represented in Table 6. The best case of each column is bolded in the Tables. The following conclusions can be found from the obtained results:

1)R2L and U2R attacks are the hardest attacks for identification. As seen, the detection accuracies of these attacks are much less than other ones.

2)The normal connections are identified with more detection accuracy than attacks.

3)By using 41 original features, the R2L attacks are detected with the highest accuracy.

4)The use of only 5 distance features fails to work.

5)The use of DDE features improves the detection accuracy compared to using the original features or the distance ones.

6)The highest detection rate is obtained when 41 DDE features are used beside 5 distance features.

7)The lowest false alarm rate is obtained when 41 DDE features are used.

8)The highest classification accuracy of the normal class is obtained by using 41 DDE features.

The proposed method is compared with the bagging boosting based on C5 decision trees [6], decision forest [7], combined several classifiers [5], rules based rough set theory

[11] and TDTC [22]. The results are reported in Table 7. Note that the best combination case of DDE features are reported for the proposed method. The best value for each assessment measure is bolded in each column. About different classes of attacks and normal, the following findings are obtained:

•For detection of attacks of probe, the proposed method provides the highest detection accuracy. After the proposed DDE based method, the TDTC method obtains good accuracy.

•For DoS attack, the decision trees method provides the best result. The combined classifiers and bagging method rank second and third, respectively with a small difference.

•For two attacks of R2L and U2R, TDTC ranks first with a significant difference with respect to others.

•For normal class, the bagging method and the rules based rough set theory provide the best detection accuracy.

•The highest detection accuracy is reported by TDTC.

•The lowest false alarm rate is achieved by the proposed DDE based method.

Generally, with a brief review on the obtained results, it can be found that the proposed method could obtain good results in all classes except two classes of R2L and U2R that are well recognized by TDTC. Eventually, although the highest detection accuracy is obtained by TDTC, but, the lowest false alarm rate is provided by the proposed method. By comparing the TDTC method with the proposed method, by considering 7 measures inclusive detection accuracy of probe, DoS, R2L, U2R, normal and also detection accuracy and false alarm rate, it can be found that, the proposed method is preferred in 4 measures (accuracy of probe, DoS, normal and false alarm rate) while TDTC is preferred in 3 measures (accuracy of R2L and U2R and detection rate).

4.Conclusion

An intrusion detection framework is proposed in this work. The DDE feature transformation is used for extraction of non-overlapped features with maximum differences between classes. The extracted features are given to the SVM classifier with polynomial kernel for classification of normal and attacks categories. Different combinations of the DDE features with 41 original features of NSL-KDD and 5 distance features defined as the distance norm of each sample to each of classes (probe, DoS, R2L, U2R and normal) are evaluated for intrusion detection. The experiments show that the use of DDE features and combination of them with distance features and the original ones can increase the detection rate and decrease the false alarm rate.

References

[1] H. Ngu, M. Gutierrez, V. Metsis, S. Nepal and Q. Z. Sheng, IoT Middleware: A Survey on Issues and Enabling Technologies, *IEEE Internet of Things Journal*, 4 (1) (2017) 1-20.

[2] Amouri, V. T. Alaparthy and S. D. Morgera, Cross layer-based intrusion detection based on network behavior for IoT, *2018 IEEE 19th Wireless and Microwave Technology Conference (WAMICON)*, Sand Key, FL (2018) 1-4.

[3] M. Frustaci, P. Pace, G. Aloi and G. Fortino, Evaluating Critical Security Issues of the IoT World: Present and Future Challenges, *IEEE Internet of Things Journal*, 5 (4) (2018) 2483-2495.

[4] E. Benkhelifa, T. Welsh and W. Hamouda, A Critical Review of Practices and Challenges in Intrusion Detection Systems for IoT: Toward Universal and Resilient Systems, *IEEE Communications Surveys & Tutorials*, 20 (4) (2018) 3496-3509.

[5] M. R. Sabhnani and G. Serpen, Application of machine learning algorithms to KDD intrusion detection dataset with in misuse detection context, In Proceedings of the international conference on machine learning: Models, technologies, and applications (2003) 209–215.

[6] B. Pfahringer, Winning the KDD99 classification cup: Bagged boosting. SIGKDD Explorations, 1 (2) (2000) 65–66.

[7] I.Levin, KDD-99 classifier learning contest LLSoft's results overview, SIGKDD Explorations, 1 (2) (2000) 67–75.

[8] D. Papamartzivanos, F. Gómez Mármol, G. Kambourakis, Dendron : Genetic trees driven rule induction for network intrusion detection systems, Future Generation Computer Systems, 79 (2) (2018) 558-574.

[9] J. Zhang, M. Zulkernine and A. Haque, Random-Forests-Based Network Intrusion Detection Systems, *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 38 (5) (2008) 649-659.

[10] Adel Nadjaran Toosi, Mohsen Kahani, A new approach to intrusion detection based on an evolutionary soft computing model using neuro-fuzzy classifiers, Computer Communications, 30 (10) (2007) 2201-2212.

[11] W. Xuren, H. Famei and X. Rongsheng, Modeling Intrusion Detection System by Discovering Association Rule in Rough Set Theory Framework, *2006 International Conference on Computational Inteligence for Modelling Control and Automation and International Conference on Intelligent Agents Web Technologies and International Commerce (CIMCA'06)*, Sydney, NSW (2006) 24-24.

[12] Y. Zhang, P. Li and X. Wang, Intrusion Detection for IoT Based on Improved Genetic Algorithm and Deep Belief Network, *IEEE Access*, 7 (2019) 31711-31722.

[13] D. Li, L. Deng, M. Lee, H. Wang, IoT data feature extraction and intrusion detection system for smart cities based on deep migration learning, International Journal of Information Management, 49 (2019) 533-545.

[14] V. V. Kumari and P. R. K. Varma, A semi-supervised intrusion detection system using active learning SVM and fuzzy c-means clustering, *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam (2017) 481-485.

[15] M. Hasan, Md. M. Islam, Md I. Islam Zarif, M.M.A. Hashem, Attack and anomaly detection in IoT sensors in IoT sites using machine learning approaches, Internet of Things, 7 (2019).

[16] K. A.P. d. Costa, J. P. Papa, C. O. Lisboa, R. Munoz, Victor Hugo C. de Albuquerque, Internet of Things: A survey on machine learning-based intrusion detection approaches, Computer Networks, 151 (2019) 147-157.

[17] S. Hajiheidari, K. Wakil, M. Badri, N. J. Navimipour, Intrusion detection systems in the Internet of things: A comprehensive investigation, Computer Networks, 160 (2019) 165-191.

[18] J. F. Charles Joseph, B. Lee, A. Das and B. Seet, Cross-Layer Detection of Sinking Behavior in Wireless Ad Hoc Networks Using SVM and FDA, *IEEE Transactions on Dependable and Secure Computing*, 8 (2) (2011) 233-245.

[19] S. Teng, N. Wu, H. Zhu, L. Teng and W. Zhang, SVM-DT-based adaptive and collaborative intrusion detection, *IEEE/CAA Journal of Automatica Sinica*, 5 (1) (2018) 108-118.

[20] P. Tao, Z. Sun and Z. Sun, An Improved Intrusion Detection Algorithm Based on GA and SVM, *IEEE Access*, 6 (2018) 13624-13631.

[21] Shi-Jinn Horng, Ming-Yang Su, Yuan-Hsin Chen, Tzong-Wann Kao, Rong-Jian Chen, Jui-Lin Lai, Citra Dwi Perkasa, A novel intrusion detection system based on hierarchical clustering and support vector machines,Expert Systems with Applications, 38 (1) (2011) 306-313.

[22] H. H. Pajouh, R. Javidan, R. Khayami, A. Dehghantanha and K. R. Choo, A Two-Layer Dimension Reduction and Two-Tier Classification Model for Anomaly-Based Intrusion Detection in IoT Backbone Networks, *IEEE Transactions on Emerging Topics in Computing*, 7 (2) (2019) 314-323.

[23] K. Fukunaga, Introduction to Statistical Pattern Recognition, 2nd ed. New York: Academic (1990).

[24] M. Imani, H. Ghassemian, High-Dimensional Image Data Feature Extraction by Double Discriminant Embedding, Pattern Analysis and

Applications, 20 (2) (2017) 473–484.
[25] G. Camps-Valls and L. Bruzzone, Kernel-based methods for hyperspectral image classification, *IEEE Trans. Geosci. Remote Sens.*, 43 (6) (2005) 1351–1362.
[26] M. Imani and H. Ghassemian, The Investigation of Sensitivity of SVM Classifier Respect to The Number of Features and The Number of Training Samples, 2nd International Conference on Sensors and Models in Photogrammetry and Remote Sensing, Tehran, Iran (2013) 209-214.
[27] C. Chang and C. Linin, LIBSVM—A library for support vector machines, 2008. Online Available: http://www.csie.ntu.edu.tw/~cjlin/libsvm.
[28] NSL-KDD Dataset, available online in https://github.com/defcom17/NSL_KDD.
[29] K. Siddique, Z. Akhtar, F. Aslam Khan and Y. Kim, KDD Cup 99 Data Sets: A Perspective on the Role of Data Sets in Network Intrusion Detection Research, in *Computer*, 52 (2) (2019) 41-51.
[30] D. Protic, Review of KDD Cup '99, NSL-KDD and Kyoto 2006+ datasets, Vojnotehnički Glasnik / Military Technical Courier, 66 (3) (2018) 580-596.
[31] R. Beghdad, Efficient deterministic method for detecting new U2R attacks, Comput. Commun., 32 (6) (2009) 1104–1110.
[32] G. K. D. Teyou, J. Ziazet, Convolutional Neural Network for Intrusion Detection System In Cyber Physical Systems, arXiv:1905.03168 (2019).
[33] W. Hu, W. Hu and S. Maybank, AdaBoost-Based Algorithm for Network Intrusion Detection, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 38 (2) (2008) 577-583.

HOW TO CITE THIS ARTICLE

M. Imani, Intrusion Detection in IOT based Networks Using Double Discriminant Analysis, AUT J. Model. Simul., 51(2) (2019) 211-220.

DOI: 10.22060/miscj.2019.16555.5161