Baltic J. Modern Computing, Vol. 5 (2017), No. 1, pp. 70–86
http://dx.doi.org/10.22364/bjmc.2017.5.1.05

Applying One-Class Classification Techniques to IP Flow Records for Intrusion Detection

Muhammad Fahad UMER[1], Muhammad SHER[1], Yaxin BI[2]

[1] Department of Computer Science, Faculty of Basic and Applied Sciences, International Islamic University, Islamabad, Pakistan
[2] Faculty of Computing, University of Ulster, UK

fahad.phdcs62@iiu.edu.pk, m.sher@iiu.edu.pk, y.bi@ulster.ac.uk

Abstract. Flow-based intrusion detection systems analyze IP flow records to detect attacks against computer networks. IP flow records contain aggregated packet header information; therefore, the amount of data processed by the intrusion detection system is reduced. In addition, since no payload is analyzed, the end-to-end encryption does not affect the deployment of intermediate intrusion detection system. In this paper, we evaluate one-class classification techniques for detection of malicious flows at an initial stage of a multi-stage flow-based intrusion detection system. The initial stage uses minimal flow attributes and only decide if the IP flow is normal or malicious. Since there is only one class of interest (malicious) at the initial stage, we use one-class classification for detection of malicious flows. In this paper, we review available one-class classification techniques and evaluate them on a flow-based dataset to determine their performance for detection of malicious flows. Our results show that one-class classification techniques using boundary methods give best results in detection of malicious IP flows.

Keywords: Intrusion detection, IP flows, One-class classification

1 Introduction

Intrusion detection systems (IDS) secure computer networks from unauthorized access and cyber attacks. The intrusion detection systems analyze network traffic and raise an alert if an attack is detected. Traditional approaches for intrusion detection use payload and protocol based inspection. Payload-based inspection techniques scan complete packet payload to detect attacks and have full access to network traffic. However, payload inspection can slow down network traffic in high-speed backbone links (Husak et al., 2015). Also, payload inspection is not possible when packet content is encrypted. Protocol-based techniques check header fields of every packet against the protocol specification. Any out of range value in protocol fields of the packet header is considered malicious. Protocol inspection techniques are protocol specific and cannot be generalized for unknown protocols.

An alternative approach to payload and protocol inspection is the flow-based inspection. Flow-based inspection uses IP flow records for intrusion detection. An IP flow is defined as a set of IP packets passing through an observation point in the network during a certain time interval. All packets belonging to a particular flow have a set of common properties (Claise et al., 2013). A flow export and collection protocol collects flow data from the network. The flow export and collection protocol makes the IP flow records available to a flow analysis application in the desired format. A common flow export protocol is Cisco's Netflow, which is supported by almost all major vendors. Internet Engineering Task Force (IETF) adopted Netflow's version 9 and standardized it as IP Flow Information Exchange (IPFIX) protocol (Sperotto and Pras, 2011). IPFIX specifies a standard architecture for collection and processing of IP flow records.

Flow-based intrusion detection systems have several advantages over payload and protocol-based techniques (Golling et al., 2014). Flow-based approaches only inspect the packet headers and do not consume any resources in the analysis of packet payloads. Since no payload inspection is involved, flow-based intrusion detection is not affected by the use of encryption. IP flows contain aggregate information in the form of IP flows and are independent of the higher layer protocols. Flow-based techniques also have some disadvantages. These systems only rely on the header fields and have no access to relevant information residing in packet payloads. Flow-based detection systems are therefore unable to detect attacks which are hidden in packet payload and do not cause a significant change in traffic flow data. Also, the flow export and collection process involve a certain delay in intrusion detection during which slow and small ramped attacks can go undetected (Vykopal et al., 2013). Golling et al., 2014) have given a comparison of flow-based intrusion detection technique with other approaches.

In this paper, we evaluate one-class classification techniques for detection of malicious flows at the first stage of a multi-stage flow-based intrusion detection system (Umer et al. 2016). The multi-stage model separates malicious flows from normal flows in the first stage. The malicious flows are forwarded to a second stage where detail intrusion detection is performed while normal flows are discarded. The detail intrusion processes can involve additional flow attributes to determine the type and other information about the attack. Umer et al. 2016 propose the use of one-class classification for detection of malicious flows because there is only target class.

One-class classification techniques are used to determine the membership of an object with the only one target class. One-class classification techniques are employed when training data is available for only one class. Training data for other classes is either not available or difficult to obtain (Khan and Madden, 2014). The class for which the training examples are available is called target class. The focus of our work is to apply one-class classification for detection of malicious flows. We evaluate available one-class classification techniques on a flow-based dataset to determine their performance in flow-based intrusion detection. In our experiment, the training dataset only contains malicious IP flows and test dataset contains both normal and malicious flows. On the basis of results, we discuss the application of available one-class classification techniques for flow-based detection of malicious traffic.

The organization of the paper is as follows: Section 2 presents existing work on the use of one-class classification techniques for intrusion detection. Section 3 describes the concept of one class classification and reviews various one-class classification methods. In section 4, we propose the use of one-class classification for detection of malicious flow in the first stage of a multi-stage intrusion detection system. We evaluate different one-class classification techniques on a flow-based dataset and discuss the results in section 5. Finally, the conclusion and future work are presented in section 6.

2 Related Work

Machine learning algorithms have remained in primary focus for designing intrusion detection systems (Liao et al., 2013). Similarly, one-class classification has also been applied to solve the intrusion detection problem. An ensemble of one-class classifiers for intrusion detection is proposed in (Giacinto et al., 2008). The authors have used a modular approach in which each module models a group of similar network protocols and services. Parzen density estimation, k-means, and v-SVM are used to construct the one-class classifier ensemble. The technique is evaluated on KDD99 dataset and results show that dividing the problem into different modules attains high detection rates with lower false alarm rates.

A flow-based intrusion detection system using one-class SVM classification is used in (Winter et al., 2011). The OC-SVM detects malicious flows and discards normal flows. A small subset of the flow-based dataset, developed by Sperotto et al., (2009), is used for evaluation.

A differential support vector data descriptor (SVDD) based one-class classification method to detect more harmful attacks using host-based intrusion detection is proposed in (Kang et al., 2012). Experimental results show that differentiated intrusion detection method performs better than existing techniques for detection of harmful attacks.

An application of one-class classification for intrusion detection in Supervisory Control and Data Acquisition (SCADA) networks is presented in (Nader et al., 2013). SCADA networks monitor and control industrial and public service processes such as nuclear power plants, electrical power grids, gas pipelines and water distribution systems. The authors have employed two one-classification methods; Support Vector Data Description (SVDD) (Tax, 2001) and Kernel Principal Component Analysis (Hoffmann, 2007). Both techniques used a SCADA network dataset for evaluation. Results indicate that both techniques tightly enclose the normal flow behavior in the SVM hypersphere and also detect intrusions.

Amer et al. (2013) proposed two enhancements in one-class SVM for unsupervised anomaly detection. Both enhancements reduce the effect of outliers on the SVM model during training. Authors have compared the proposed techniques with nine other unsupervised anomaly detection algorithms and obtained promising results.

An industrial communication intrusion detection algorithm based on one-class SVM is presented in (Shang et al., 2015). Authors have used particle swarm optimization (Couceiro and Ghamisi, 2016) to tune the kernel parameters.

A robust one-class SVM for outlier detection is presented in (Yang et al., 2016). The authors use dynamic weight assignment for training datasets for the smooth influence on one-class SVM. Experimental analysis shows the proposed weighted method has improved performance and robustness as compared to the conventional one-class SVM.

Survey of literature shows that although one-class classification has been in use for intrusion detection, there are many other one-classification techniques yet to be explored in detail for flow-based intrusion detection.

3 One-class classification

One-class classification is a particular case of binary classification which recognizes examples of only one class. The one-class classification is useful when training examples of only one class are available. Training sample for other classes is either not available or difficult to obtain (Khan and Madden, 2014). The class for which training samples are available is called a target class. An important application of one-class classification is intrusion detection where target class is malicious category of network traffic. The one-class classification based intrusion detection only detects malicious traffic and discards the normal traffic.

Mathematically, we assume that x_i is an training example from the target class dataset $X = \{x_1, x_2, x_3, ..., x_n\}$. The one-class classifier uses the training dataset X to learn a model with the optimized parameter set θ. After learning, the classifier can identify target class examples from unseen test dataset Z. The classifier defines an output function f using the optimized parameter set θ such that:

$$f(z_i, \theta) = c_i \tag{1}$$

Where z_i is an unseen example, and c_i is the class probability. The classifier uses a mapping function $h(z_i)$ over the output function $f(z_i)$ to classify unseen examples into target or outlier classes. If the class probability is higher than a pre-defined threshold t, the target class is selected otherwise the example is declared outlier.

$$h(z_i, \theta) = \begin{cases} target, & \text{if} f(z_i, \theta) \geq t \\ outlier, & \text{if} f(z_i, \theta) < t \end{cases} \tag{2}$$

Available one-class classification techniques include density estimation methods, boundary methods and reconstruction methods (Tax, 2001; Mazhelis, 2006).

3.1 Density Estimation

The density estimation methods calculate the density of target class from training data using a density estimation function $f(z)$. The function $f(z)$ calculates the density of a test example and uses a threshold value to accept the example in the target class. If the density of test example is higher than the threshold, it is classified into target class. Otherwise, the example is considered an outlier (Pimentel et al., 2014). We have evaluated three destiny estimation methods; simple Gaussian distribution using Mahalanobis distance, mixture of Gaussian and Parzen distribution.

3.1.1 Simple Gaussian distribution using Mahalanobis distance This technique uses the Mahalanobis distance to calculate the density of test example z in the Gaussian distribution:

$$f(z) = (z - \mu)^T \Sigma^{-1} (z - \mu) \tag{3}$$

where μ is mean and Σ is covariance matrix.

3.1.2 Mixture of Gaussian The simple Gaussian distribution does not fit most data distributions. To model more complex data, a mixture of different Gaussian distribution is used. Density at point z for mixture of different Gaussian distributions is defined using the following equation:

$$f(z) = \sum_{i}^{k} P_i e^{(z-\mu_i)^T \Sigma^{-1} (z-\mu_i)} \tag{4}$$

Where P_i is the probability that z belongs to ith Gaussian component, μ_i is mean and Σ is covariance matrix.

3.1.3 Parzen Distribution Parzen density estimation uses a hypercube $\Re$ with dimension d and width h to calculate the density at point z. The hypercube $\Re$ is centered at z and density is calculated with respect to all examples x_i. We define a function ϕ which gives a value of 1 if an example x_i is within the hypercube $\Re$ or 0 otherwise.

$$\phi(\frac{z - x_i}{h}) = \begin{cases} 1, & \text{if } x_i \text{ is inside the hypercube} \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

$$f(z) = \sum_{i=1}^{N} \phi(\frac{z - x_i}{h}) \tag{6}$$

We use Gaussian estimate for the function $f(z)$ such that:

$$f(z) = \sum_{i=1}^{N} e^{(-(z-x_i)^T h^{-2} (z-x_i))} \tag{7}$$

3.2 Reconstruction methods

Reconstruction methods use training dataset to model the generating process for all examples. A reconstruction error is used to measure the fit of an actual example for the generating model. If the example does not fit the model and reconstruction error is high, the example is more likely an outlier (Pimentel et al., 2014). Reconstruction methods include neural networks (e.g. Auto-encoder neural networks, self-organizing map) and subspace-based approaches (e.g. Principle Component Analysis).

3.2.1 Auto-encoder Neural Networks The auto-encoder neural networks encode the input to pass through a compact hidden stage. The input is reconstructed (decoded) at the output stage and matched with original input to calculate the reconstruction error:

$$f(z) = ||z - z_{recons}||^2 \tag{8}$$

3.2.2 Self-organizing map Self-organizing maps is a clustering technique where high dimension input space is mapped to low dimension output clusters. The self-organizing map has input and output layers which connect with each other through a competitive learning network. The output layer contains all clusters. An unseen example is placed at the output layer and the distance from the closest cluster center is calculated. The distance is reconstruction error defined by:

$$f(z) = min_k||z - \mu_k||^2 \tag{9}$$

where k is number of output clusters.

3.2.3 Principle Component Analysis The Principle component analysis (PCA) is dimension reduction technique which maps the input space of size N to output space M such that $M < N$. The projection of input object z from N to M is defined by:

$$y = WW^Tz \tag{10}$$

where W is $k \times d$ matrix storing eigenvector for output space, $d = N$.

PCA use the reconstruction error as a classification function. The reconstruction error is defined as the squared distance between the projection and the original example:

$$f(z) = ||z - y||^2 \tag{11}$$

3.3 Boundary Methods

Boundary methods construct a boundary around the target class examples such that most of the target examples lie within the boundary. These methods use a threshold value for acceptance of outliers within the boundary. An unseen example z belongs to target class if it lies within the boundary. We have used ν-Support Vector Machine(SVM) (Schlkopf et al., 2001) and Support Vector Data Descriptors (Tax, 2001) for evaluation of IP flows records.

3.3.1 ν-SVM The ν-SVM construct a boundary around the target class examples in the form of a hyperplane during training (Schlkopf et al., 2001). An unseen example z belong to target class if it falls within the hyperplane and considered outlier otherwise.

SVM uses a feature mapping function $\phi : X \rightarrow H$ which maps the input feature space X to a high dimension feature space H (Li, 2015). The similarity between an input x and its class prediction y in feature space H is be calculated using a simple kernel function:

$$K(x, y) = (\phi(x).\phi(y))_H \tag{12}$$

To separate the input examples from the origin with maximum margin using a hyperplane, following quadratic minimizing function is applied:

$$min_{w,\xi,\rho}\frac{1}{2}\|w\|^2 + \frac{1}{m\nu}\sum_{i=1}^{m}\xi_i - \rho \tag{13}$$

Subject to

$$(w.\phi(x_i)) \geq \rho - \xi_i, \xi_i \geq 0 \tag{14}$$

where
ξ_i : Slack variable to penalize the outliers
$\nu \in (0, 1)$: A user-defined error control parameter and sets an upper bound on the fraction of outliers and a lower bound on the number of support vectors.
ρ : The maximal margin for hyperplane from the origin.

Using Lagrange multipliers and constructing a Lagrangian, the decision function for the classification of a test example z is defined as follows (Schlkopf et al., 2001):

$$f(z) = sgn(\sum_i \alpha_i k(z_i, z) - \rho) \tag{15}$$

3.3.2 Support Vector Data Descriptor(SVDD) The support vector data descriptors construct a hypersphere around the target class training examples. The sphere has its center a and with radius $R > 0$. The volume of the sphere is minimized such that it contains all training examples (Tax, 2001).

Following error function is minimized for SVDD:

$$F(R, a) = R^2 \tag{16}$$

Such that all target examples x_i lies within the sphere:

$$||x_i - a||^2 \leq R^2 \tag{17}$$

The strict data description of minimizing the sphere radius may not fit in all cases. To allow the possibility of outliers in the sphere, and to penalize the larger distances for x_i, slack variables are introduced. Thus, the minimization problem becomes:

$$F(R, a, \xi) = R^2 + C\sum_i \xi_i \tag{18}$$

Such that maximum target examples x_i lies within the sphere:

$$||x_i - a||^2 \leq R^2 + \xi_i, \xi_i \geq 0 \tag{19}$$

The C parameter is analogues to ν in ν-SVM and control the acceptable number of outliers.The parameters R, a, and ξ are optimized by using Lagrange multipliers and constructing a Lagrangian. We have following quadratic minimization problem (Tax, 2001):

$$L(R, a, \xi, \alpha, \gamma) = \sum_i \alpha_i x_i.x_i - \sum_{i,j} \alpha_i \alpha_j.x_j \tag{20}$$

A test example z is classified into target class if its distance from center a is less than the radius of the sphere. The decision function for classification is defined as follows:

$$f(z) = \begin{cases} 1, & \text{if } ||z - a||^2 \leq R^2 \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

4 System Model

Flow-based techniques evaluate the behavior and communication pattern of network attacks for intrusion detection. The goal of our experiment is to evaluate the application of available one class classification techniques for detection of malicious IP flows. Umer et al., (2016) propose a multi-stage model for flow-based intrusion detection. The multi-stage model uses one-class classification at first stage. The first stage separates malicious flows from normal flows. The normal flows are discarded while malicious flows are forwarded to a second stage. The second stage uses multi-class classification and classify the malicious flows under an attack type. The first stage uses a minimal set of attributes for quick detection of malicious flows. The second stage uses additional flow attributes and performs detail intrusion detection process. The multi-stage model is efficient because only malicious flows are analyzed in detail and a majority of the flow, being normal, are discarded at the first stage.

Figure 1 shows the architecture of multi-stage intrusion detection using one-class classification at first stage. The one-class classification is a supervised learning technique and uses a set of malicious IP flows for training. After training, the flow-based intrusion detection system is provided with the IP flow traffic for detection of malicious activity. The pre-processing stage converts IP flow records in a format accepted by the one-class classifier. The one-class classifier evaluates incoming IP flows and detects malicious flows. The normal flows are discarded and malicious IP flows are forwarded to the second stage of intrusion detection system. The second stage uses multi-class classification to assign an attack type label to a malicious flow.

We consider various one-classification methods including density estimation, reconstruction methods and boundary methods for detection of malicious IP flows. We evaluate the one-class classification techniques on a flow-based dataset and obtain results to determine the suitable one-class classification technique for detection of malicious IP flows.

5 Experimental Results

5.1 Dataset

We have used two scenarios of CTU-13 dataset for evaluation of one-class classification techniques. The CTU-13 dataset was created in CTU University, Czech Repub-

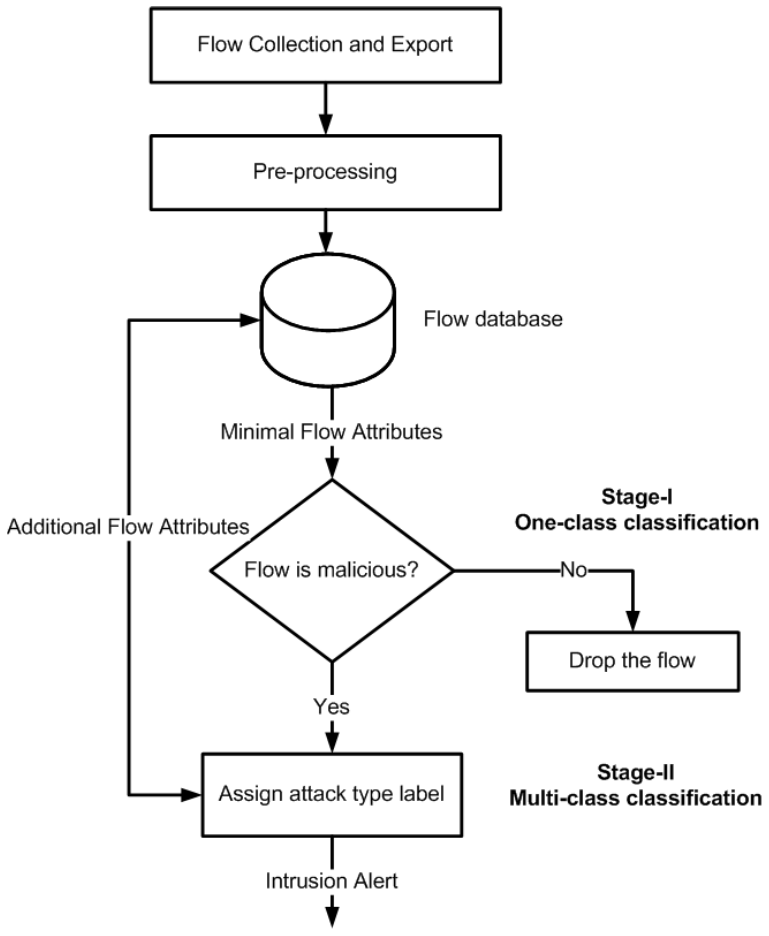


Fig. 1: A multi-stage flow-based intrusion detection model

lic (Garcia et al., 2014). The dataset consists of botnet traffic mixed with normal and background communication traffic. The traffic capture process consists of 13 different scenarios where a particular malware traffic was captured in each scenario. The environment for traffic capture consists of virtual machines running the Microsoft Windows XP SP2 operating system on top of a Linux Debian host. These virtual machines were bridged into the University network. The traffic was captured both on the Linux host and on the university network router connected to the Linux host. The authors have carefully assigned the ground-truth labels to the collected dataset. During ground-truth labeling process, all traffic was marked as background traffic. The normal label was given to the traffic that was originated from switches, proxies, and known legitimate computers. All traffic that came from the infected machines was labeled as botnet. Every captured scenario contains different types of attacks. The distinct attack types in the CTU-13 dataset are DDOS, SPAM, Click Fraud and Port Scan. We have used fourth and fifth scenario of the dataset for our experiment. These scenarios are smaller in size but contain all malicious botnet traffic except Click Fraud. Table 1 gives detail of the malware and traffic flow records in each scenario.

Table 1. Detail of IP flow records - CTU-13 intrusion dataset

Bot	**Characteristic**	**Total Flows**	**Botnet and C&C Flows**	**Normal Flows**	**Background Flows**
RBot	IRC, DDOS, US	1121076	1719	25268	1094040
Virut	SPAM, Port Scan, HTTP	129832	695	4679	124252

The CTU dataset is available in the form of bi-directional Netflow records. Every flow record has 15 attributes. Table 2 shows a sample of the dataset with important attributes. The duration field is used to calculate the flow duration. The protocol value shows the transport layer protocol type. Source and destination IP and Port address fields show sender and receiver of flow traffic. The total packets and total bytes fields contain the total number of packets and bytes transmitted in either direction. The label field shows the type of flow. We have extracted a subset of the CTU-13 dataset for evaluation of one-class classification techniques. The extracted dataset contains the same number of malicious flows. The normal flows are randomly sampled due to memory limitations of the available hardware. Detail of IP flows in the extracted subset are given in Table 3.

5.2 Performance Measures

We have evaluated one-class classification techniques using two well-known performance measures; Area under Receiver Operating Characteristic (ROC) curve (AUC) and F1 score (Wu and Banzhaf, 2010). The ROC curve plots the false alarm rate against true positive rate. In our experiment, ROC curve measures the target accepted rate as the outlier accepted threshold varies. The ROC curve is quantified by measuring the area

Table 2. Important flow attributes - CTU-13 intrusion dataset

Duration (msec)	**Prot.**	**Source IP Address**	**Source Port**	**Destination IP Address**	**Dest. Port**	**Total Packets**	**Total Bytes**	**Label**
3550.1823	udp	212.50.71.179	39678	147.32.84.229	13363	12	875	normal
0.0008	udp	84.13.246.132	28431	147.32.84.229	13363	2	135	normal
0.0003	tcp	217.163.21.35	80	147.32.86.194	2063	2	120	normal
0.0569	tcp	83.3.77.74	32882	147.32.85.5	21857	3	180	normal
3427.7680	udp	74.89.223.204	21278	147.32.84.229	13363	42	2856	normal
3086.5473	tcp	66.169.184.207	49372	147.32.84.229	13363	591	45931	normal

Table 3. IP Flow dataset for one-class classification

Training dataset		**Testing dataset**	
Malicious	Normal	Malicious	Normal
1859	0	810	112482

under the curve (AUC). The value of AUC near 1 denotes a good intrusion detection process.

The second performance measure is F1 score. F1 score is equal to the harmonic mean of precision and recall values:

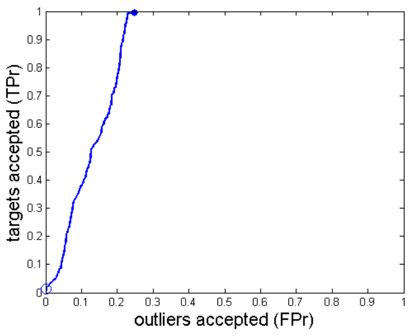
$$Precision = \frac{\text{number of true positives}}{\text{number of true positives + false positives}} \tag{22}$$

$$Recall = \frac{\text{number of true positives}}{\text{number of true positives + false negative}} \tag{23}$$

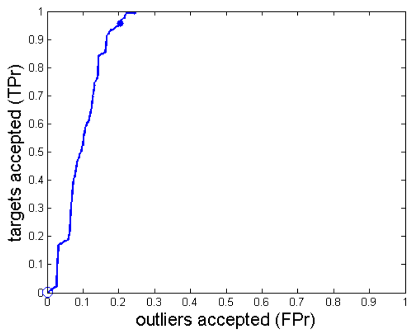
$$F1 - score = \frac{2 \times precision \times recall}{precision + recall} \tag{24}$$

5.3 Results

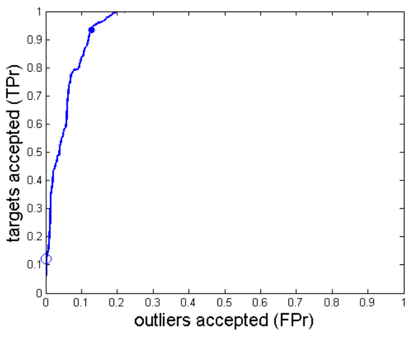
We have used Matlab, Weka, DDtools (Tax, 2015) and LibSVM (Chang and Lin, 2011) for performing the experiment. Table 4 gives detail results for various one-class classification techniques using multiple performance measures. During evaluation of density estimation methods, the ROC curves for Gaussian (Figure 2a) and Mixture of Gaussian (Figure 2b) show that these techniques are not very accurate and only give high detection rate with the inclusion of outliers. The ROC curve for Parzen density estimate (Figure 2c) shows that it initially performs better. However, the accuracy of Parzen density estimate decreases when target accepted rate rises above 0.5. The simple Gaussian gives a value of 0.8709 and 0.5461 for AUC and F1 score respectively. Mixture of Gaussian shows little improvement with AUC of 0.9024 and F1 score of 0.5773. The Parzen density estimate has AUC of 0.952and F1 score of 0.6715.



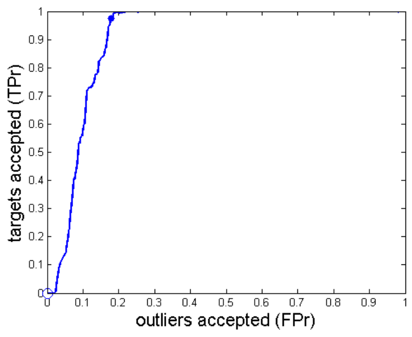
(a) Simple Gaussian



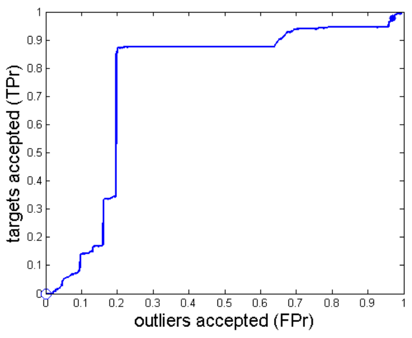
(b) Mixture of Gaussian



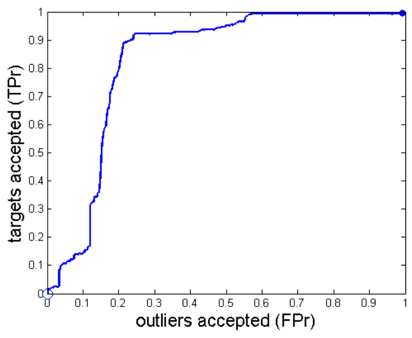
(c) Parzen Distribution



(d) Auto-encoder Neural Network



(e) Self-organizing Maps

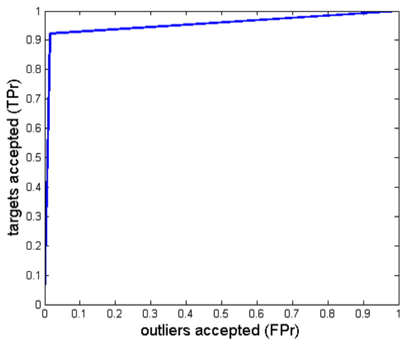


(f) Principle Component Analysis

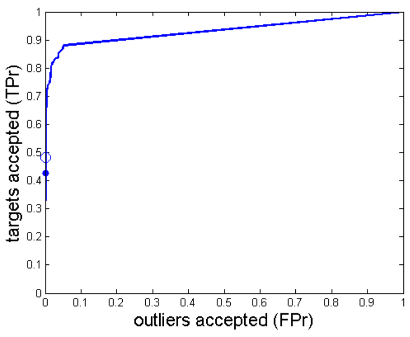
Fig. 2: ROC curves for density and reconstruction methods one-class classification techniques

Table 4. Results for one-class classification of malicious IP flows

One-class Classification Technique	Area under ROC Curve	Precision	Recall	F1 score
Density Estimation				
Simple Gaussian	0.8709	0.3763	0.9957	0.5461
Mixture of Gaussian	0.9024	0.4145	0.9561	0.5773
Parzen density estimation	0.952	0.5244	0.0.9336	0.6715
Reconstruction Methods				
Auto-encoder Neural Network	0.905	1.0	0.3604	0.5298
Self Organizing Maps	0.7549	0.1306	0.9785	0.2304
Principle Component Analysis	0.9288	0.1299	0.9957	0.2298
Boundary Methods				
ν-SVM	0.9297	0.9103	0.9125	0.9114
Support Vector Data Descriptor (SVDD)	0.9335	0.9258	0.8953	0.9102



(a) ν-SVM



(b) Support Vector Data Descriptor

Fig. 3: ROC curves for boundary methods one-class classification techniques

Reconstruction methods give average results for one-class classification. The ROC curve for auto-encoder neural network (Figure 2d) is similar to density estimation techniques. However, the ROC curves for self-organizing map (SOM) (Figure 2e) and PCA (Figure 2f) show that these techniques do not perform well and fail to achieve good target acceptance rate with high outlier acceptance ratio. The auto-encoder neural network shows better performance with AUC of 0.905 and F1 score of 0.5298. The AUC for SOM is 0.7549 and F1 score is 0.2304. The Principle component analysis (PCA) has AUC of 0.9288 and F1 score of 0.2298.

One-class classification using boundary methods gives best results in detection of malicious IP flows. The ROC curves for ν-SVM (Figure 3a) and support vector data descriptor (SVDD) (Figure 3b) are similar to each other. The curves shows that boundary methods give high target acceptance rate with least number of outliers. The ν-SVM has AUC of 0.9297 and F1 score of 0.9114. The SVDD also has similar results with AUC of 0.9288 and F1 score of 0.2298.

An interesting observation from the ROC curves of is that most of the classification method perform well if the

5.4 Discussion

In the first step, we have evaluated one-class classifiers techniques based on density estimation methods. The performance of density methods depends on the correct estimation of the density function. These techniques give good results if the data is well-sampled and a large training dataset is available. However, the density functions do not give accurate results if data is sparsely plotted in the feature space. Due to this limitation, these techniques do not have good performance for one-class classification. Our results also show that density based approaches have lower performance for one-class classification of malicious IP flows. The simple Gaussian method has lower accuracy because IP flow records and other data distributions do not fit in the bell curve of Gaussian representation. The constraint of a single bell curve is relaxed by using a mixture of Gaussian distributions. The mixture of Gaussian combine multiple Gaussian representations and also shows some improvement in results. However, still the mixture of Gaussian model is not an accurate representation of IP flows traffic distribution. The Parzen density method shows improved performance among all density estimation techniques. Parzen density estimation requires all training data to be made available during testing (Mazhelis, 2006). This is particularly challenging in the case of intrusion detection in network traffic where large training records are pushed on regular intervals.

In next step, the one-class classifier based on reconstruction methods have been evaluated. The reconstruction methods use a generating process to model the data. The parameters of the generating process are optimized for the correct representation of new objects. Although reconstruction methods can be applied for one-class classification problems, they are not primarily meant for this purpose 2001, 2014. In case the data does not fit the model, a bias value is used to minimize the reconstruction error. The bias value destroys the important characteristics of the dataset (Tax, 2001). Accurate modeling of IP flow records using reconstruction methods requires a large number of parameter to be optimized and can have high reconstruction error. These methods are computationally expensive for one class classification (Mazhelis, 2006). Another drawback of reconstruction methods is the difficulty in training high-dimensional spaces (Pimentel et al., 2014). This aspect limits the use of reconstruction methods for classification of IP flows because flow records can be very high dimensional as IPFIX includes around 280 attributes to define an IP flow record.

In next experiment, we have applied two neural network approaches which include auto-encoder neural network and self-organizing map. Results show that auto-encoder neural network has relatively good accuracy, but it requires some parameters to be specified by the user. These include the number of hidden layers, input units, and stopping criteria (Mazhelis, 2006). The auto-encoder neural network also needs a large training set for correct estimation of weights associated with hidden and input units. The self-organizing map (SOM) requires the user to specify the number of output clusters. Also SOM needs k^d neurons for d dimension dataset. It is computational expensive to estimate the weights of k^d neurons if both k and d are moderately higher (Tax, 2001). Another type of reconstruction method is Principle component analysis (PCA). PCA also has lower accuracy for one-class classification of IP flows. The PCA does not perform well if the data has variance in all feature direction. In this case, PCA is unable to reduce the dimensionality of data (Tax, 2001).

One-class classification techniques using boundary methods give best results for identification of malicious IP flows. The boundary methods are specifically focused on one-class classification and also perform better if limited training sample is available (Tax, 2001). These methods avoid the estimation of the complete probability density and can work efficiently if the dataset is high dimensional. In our experiment, we have used two SVM-based one class classification techniques; ν-SVM and Support Vector Data Descriptor (SVDD). The ν-SVM and SVDD use hyperplane and hypersphere respectively to enclose the target class examples. Finding the smallest sphere containing all target points is equivalent to find the segment containing the required points (Schlkopf et al., 2001). In our experiment, both methods show similar result and perform better than all reviewed classification methods. However, a drawback of SVM-based methods is the difficulty involve in the estimation of values for the parameter that controls the number of outliers within the boundary. The accuracy of SVM-based one-class classifiers is very sensitive to a slight change in the value of these parameters.

6 Conclusion and Future Work

In this paper, we have applied different one-class classification techniques for detection of malicious IP flows. The techniques include density estimation, reconstruction methods, and boundary methods. We used a flow-based dataset to train the one-class classifiers on malicious IP flows. The trained classifier is evaluated on a test dataset containing both normal and malicious IP flow records. We have used multiple performance measures including Area under ROC curve (AUC) and F1 score for comparison of results. On the basis of results, we discussed pros and cons of all techniques used for one-class classification. The results show that boundary methods i.e. SVM-based one-class classifiers give higher accuracy in identification of malicious IP flows. We, therefore, consider SVM-based one-classification techniques suitable for detection of malicious IP flows on the basis of experimental results.

In future, our focus will be on implementing a multi-stage intrusion detection system using SVM-based one-class classifier at the first stage. The second stage will classify IP flows into different attack categories and provide deep insight into the malicious traffic. Another point of interest will be to combine multiple one-class classifiers for improvement in performance. Use of unsupervised learning for one-class classification is also a promising research area.

References

Amer, M., Goldstein, M., Abdennadher, S. (2013). Enhancing one-class support vector machines for unsupervised anomaly detection. In *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, pages 8–15. ACM.

Chang, C., Lin, C. (2011). Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2(3):,27:1–27:27, 5 2011. ISSN 2157-6904. DOI10.1145/1961189.1961199. URL http://doi.acm.org/10.1145/1961189.1961199.

Claise, B., Trammell, B. Aitken, P. (2013). Specification of the IP flow information export (IPFIX) protocol for the exchange of flow information. Technical report, IETF.

Couceiro, M. Ghamisi, P. (2016). Particle swarm optimization. In *Fractional Order Darwinian Particle Swarm Optimization*, pages 1–10. Springer.

Garcia, S., Grill, G., Stiborek, J., Zunino, A. (2014). An empirical comparison of botnet detection methods. *computers & security*, 45:100–123.

Giacinto, G., Perdisci., R., Del Rio, M., Roli, F. Intrusion detection in computer networks by a modular ensemble of one-class classifiers. *Information Fusion*, 9(1):69–82.

Golling, M., Hofstede, R., Koch, R. (2014). Towards multi-layered intrusion detection in high-speed networks. In *Cyber Conflict (CyCon 2014), 2014 6th International Conference On*, pages 191–206. IEEE.

Hoffmann, H. (2007). Kernel PCA for novelty detection. *Pattern Recognition*, 40(3):863–874, 2007.

Husak, H. Velan P., Vykopal, J. (2015). Security monitoring of http traffic using extended flows. In *Availability, Reliability and Security (ARES), 2015 10th International Conference on*, pages 258–265. IEEE.

Kang, I., Jeong, M., Kong, D. (2012). A differentiated one-class classification method with applications to intrusion detection. *Expert Systems with Applications*, 39(4):3899–3905.

Khan, S., Madden, M. (2014). One-class classification: taxonomy of study and review of techniques. *The Knowledge Engineering Review*, 29(03):345–374.

Li, L., (2015). Support vector machines. In *Selected Applications of Convex Optimization*, pages 17–52. Springer.

Liao, H., Lin, C. Lin, Y. Tung, K. (2013). Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24.

Mazhelis, O. (2006). One-class classifiers: a review and analysis of suitability in the context of mobile-masquerader detection. *South African Computer Journal*, 36:29–48.

Nader, P., Honeine, P., Beauseroy, P. (2013). Intrusion detection in SCADA systems using one-class classification. In *Signal Processing Conference (EUSIPCO), 2013 Proceedings of the 21st European*, pages 1–5. IEEE.

Pimentel, M., Clifton, D. Clifton, L. Tarassenko, L. (2014). A review of novelty detection. *Signal Processing*, 99:215–249.

Schölkopf, B., Platt, J, Shawe-Taylor, J., Smola, A. Williamson, R. (2001). Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471.

Shang, W., Li, L., Wan, M, Zeng, P. Industrial communication intrusion detection algorithm based on improved one-class SVM. In *2015 World Congress on Industrial Control Systems Security (WCICSS)*, pages 21–25. IEEE.

Sperotto, A., Pras, A. (2011). Flow-based intrusion detection. In *Integrated Network Management (IM), 2011 IFIP/IEEE International Symposium on*, pages 958–963. IEEE.

Sperotto, A., Sadre, R. Van Vliet, F., Pras, A., (2009). A labeled data set for flow-based intrusion detection. In *IP Operations and Management*, pages 39–50. Springer.

Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A. Stiller, B., (2010). An overview of IP flow-based intrusion detection. *Communications Surveys & Tutorials, IEEE*, 12(3):343–356.

Tax, D., (2001). *One-class classification*. PhD Thesis, TU Delft, Delft University of Technology, Netherlands.

Tax, D., (2015). Ddtools, the data description toolbox for matlab, version 2.1.2.

Umer, M. F., Sher, M., Khan, I., (2016). Towards Multi-Stage Intrusion Detection using IP Flow Records. *International Journal of Advanced Computer Science and Applications(IJACSA)*, 7 (10):343–347.

Vizvry M, Vykopal J. (2013). Flow-based detection of RDP brute-force attacks. *In Proceedings of 7th International Conference on Security and Protection of Information (SPI 2013).*

Winter, P., Hermann, E., Zeilinger, M., (2011). Inductive intrusion detection in flow-based network data using one-class support vector machines. In *New Technologies, Mobility and Security (NTMS), 2011 4th IFIP International Conference on*, pages 1–5. IEEE.

Wozniak, M., (2014). Hybrid classifiers–methods of data, knowledge, and classifier combination, ser. *Studies in Computational Intelligence, Springer*, 519, 2014.

Wu, S., Banzhaf, W., (2010). The use of computational intelligence in intrusion detection systems: A review. *Applied Soft Computing*, 10(1):1–35.

Yang, J., Deng, T., Sui, R. (2016). An adaptive weighted one-class svm for robust outlier detection. In *Proceedings of the 2015 Chinese Intelligent Systems Conference*, pages 475–484. Springer.

Received August 31, 2016 , revised December 22, 2016, accepted February 28, 2017