Intrusion Detection Based on Outlier Detection Method

Prajowal Manandhar, and Zeyar Aung

***Abstract*—** Intrusion detection is an effective mechanism to deal with challenges in network security. The rapid development in networking technology has raised the need for an effective intrusion detection system (IDS) as traditional intrusion detection methods cannot compete against the newly advanced intrusion attacks. With increasing number of data being transmitted daily to/from a network, the system needs to detect intrusion in such a large data traffic quickly and reliably. Thus, the application of data mining/machine learning comes handy to identify such unusual attacks in an efficient and effective manner. In this paper, we propose an intrusion detection system based on outlier detection, which is an instance of data mining. We deploy a sequence of operations of principal component analysis (PCA), k-means clustering, and single-class support vector machine (SVM) in order to model the normal network traffic instances. We then feed the new instances to the resultant model to predict whether they are normal or attacks. We have applied our model to the KDD Cup '99 dataset and achieved promising results.

***Keywords*—**IDS, anomaly, outlier, PCA, k-means, SVM.

I. Introduction

INTRUSION detection is a network security mechanism to protect the computer network system from attacks. Advancement in the network technology has provided opening to the hackers and intruders to find unauthorized ways to enter into a new system. Therefore, as technologies evolve, there is also a risk of new threats existing with them. Thus, when a new type of invasion emerges, an intrusion detection system (IDS) needs to be able to act effectively and in a timely manner in order to avoid hazardous effects. In today's context, the biggest challenge could be dealing with a 'big data', i.e., a huge volume of network traffic data that gets collected dynamically in the network communications [1]. Therefore, intrusion detection has been one of the core areas of computer security whose objective is to identify these malicious activities in network traffic and importantly, protect the resources from the threat. Most of the IDS try to perform their task in real time but due to circumstances like degree of analysis and computation it needs to undergo, the real time performance is not always possible.

The main essence of using statistical methods for outlier detection in anomaly-based approach lies in analyzing and mining information from raw data, to improve learning capability to model normal behavior of the system. In order to ensure a high detection rate, we need to model our normal data properly. These anomaly-based approaches, when compared to signature-based ones, can lead to a faster execution but often result in high false positive rate. Since the intrusion patterns and normal patterns do not always comply with certain distributions nor is it linearly separable thus, this will cause problems in applying statistical learning methods, such as support vector machine (SVM) for intrusion detection.

In this paper, we propose an anomaly-based intrusion detection system using an outlier detection approach. We use a technique based on streamlining of principal component analysis (PCA), clustering, and single-class support vector machine (SVM) to model the normal network traffic instances. Then, we employ the resultant single-class SVM model to classify the new network traffic instances as either normal or malicious. We have experimentally evaluated our proposed scheme using the well-known KDD Cup '99 dataset [2], and found that our scheme can provide a good accuracy in detecting intrusion attacks to a network.

II. Background and Literature Review

A. Intrusion Detection Systems

The intrusion detection method can be categorized into two types based on its analysis methods:

i. Anomaly-based intrusion detection method
ii. Misuse or signature-based intrusion detection method.

In order to detect intrusions, the anomaly detection analyzes the deviation from the normal activities at user or system level whereas the misuse detection matches sample data to known intrusive patterns. Machine learning frequently forms the basis for anomaly method and it can detect novel attacks by comparing suspicious ones with normal traffics but has high false alarm rate due to difficulty of modeling practical normal behavior profiles for protected system [3]. With misuse method, pattern matching on known signatures leads to high accuracy for detecting threats but it cannot detect novel attacks as novel attack signatures are not available for pattern matching. The most of the current IDSs follow a signature-based or misuse approach which is similar to virus scanners, where events are detected after matching with specific pre-defined patterns known as signatures. The limitation identified with signature-based IDS is their downfall to detect new attacks and also neglect minor variations of known patterns. Besides that it is found to have significant administrative overhead cost attached with it in order to maintain signature

Prajowal Manandhar and Zeyar Aung are with Institute Center for Smart and Sustainable Systems (iSmart), Masdar Institute of Science and Technology, Abu Dhabi, PO Box 54224, UAE (e-mails: {pmanandhar, zaung}@masdar.ac.ae).

databases.

Anomaly based detection was started using statistics. Haystack [4] used statistics to monitor changes in user behavior. NSM also used statistics along with rules to monitor LAN traffic. The Nides [5] statistical component set the standard for statistical based intrusion detection as it computes a historical distribution of continuous and categorical attributes that gets updated over time and deviation are found using chi-square tests.

B. Data Mining for Intrusion Detection

Data mining is an approach used to identify useful informative patterns and relationships within the large amount of data by a variety of data analysis tools and techniques, such as classification, association, clustering, and visualization [6]. It is closely associated and has much overlapping with the area of machine learning.

Data mining has recently obtained popularity within many research fields, for example, marketing, bio-medical and telecommunication [7]. It has also been a preferred candidate for assisting in the analysis of network data. It can provide answers to the end-users about big data problems by filtering out big data into useful subsets of information. Essentially applying data mining tools to network data provides the ability to identify the various underlying contexts associated with the network being monitored. There are two important learning strategies in data mining technique: supervised learning and unsupervised learning.

In supervised learning, each instance of data is mapped to align with its associated label in order to find the interpretation of these pattern labels. Decision trees (DTs) and estimation techniques are examples of supervised learning. Unsupervised learning is to discover a number of pattern labels, subsets, or segments within the data, with no prior knowledge of the target classes or concepts. Examples of unsupervised learning are clustering, blind signal separation and self-organizing maps (SOM). Different data analysis techniques have been applied for IDS. Top-down Learning method is one such statistical technique which is used when there are general ideas about specific relations and hence is used along with mathematical computations to orient researches and studies.

Many data mining algorithms are considered good for attack detection; likewise DT is considered one of the most powerful and effective ways of detecting attacks in anomaly detection [8]. We have found other different data mining approaches like naive Bayes, neural networks, support vector machines (SVM) and Random Forest that have been used to analyze and learn the traffic system. Attack signatures are required for misuse detection which can be found using approaches like DT, neural networks, and SVM.

It was also perceived that ensemble learning which is a combination of multiple machine learning approaches, being used to improve the result of instruction detection as compared to the result of a single classifier. However it was learnt that each individual base classifier are needed to be independent of others in order to get effective and correct classification otherwise there would be no significant improvement from ensemble result. The hybrid classifier is a combinatory method based on algorithms like DT, k-nearest neighbors, fuzzy logic and neural networks which detects both anomaly and misuse attacks in a system [8]. The work presented by Faisal et al. [9], [10] provides a new insight towards intrusion detection by treating network events as a stream of data and using different data stream-based learning models.

C. Outlier Detection for Intrusion Detection

Outlier detection is yet another instance of data mining that is useful for intrusion detection. An outlier can be understood as an observation that deviates from the other normal observations as to cause suspicion that it was generated by a different mechanism. We can view outliers as of two types. The first type is the one that deviate significantly from others within their own network peripherals while the second type is the one whose patterns belongs to other network services other than their own service. It was also observed that researchers have tried two approaches either to model only normal data or both normal and abnormal data for finding intruders.

Modeling both the normal and abnormal data allows the system to be tight on false positive and false negative as both the normal as well as abnormal data needs to be modeled but it puts the limitation in modeling the abnormal patterns. Similarly, using only normal patterns allows the system to model the boundaries of normal data but due to the unknown nature of the boundaries of abnormal data, it gives a possibility of alarming false positive, as well. Thus, proper tuning needs to be done for defining the threshold of the system to balance the number of true positives and false positives. Outlier detection has been a popular technique for finding intrusions, which can be observed with the amounts of work done in papers [11], [12], [13] and [14]. Most of the research works have pointed out that it is extremely difficult to find out outliers directly from high dimensional datasets so, most of the researches proceeds with reducing the dimensionality of the dataset [13].

There are various approaches for Outlier Detection, which includes model-based approach, proximity-based approach and angle-based approach [15]. In Model-based approach, we apply the model to represent normal data points, and we assume those points that do not fit the model are outliers. Here, probabilistic tests based on statistical models, depth-based approaches and deviation-based approaches can be used for this outlier detection model. In proximity-based approaches, we examine the spatial proximity of each object in the data space and then we consider the object as an outlier if the proximity of an object deviates considerably from the proximity of other objects. Here, distance-based approaches and density-based approaches can be used for this outlier detection model. The rationale behind angle-based approach is that we measure the spectrum of pairwise angles between a given point and all other points. So, the outliers would be the points with a spectrum featuring high fluctuation problems.

III. PROPOSED ARCHITECTURE

The architecture of our model is provided in Fig. 1. This

architecture is adapted from the one used in our previous work [16]. (In the previous work, we performed intrusion detection by using tcpdump data files as inputs rather than the KDD dataset's feature vectors as in this current work.)

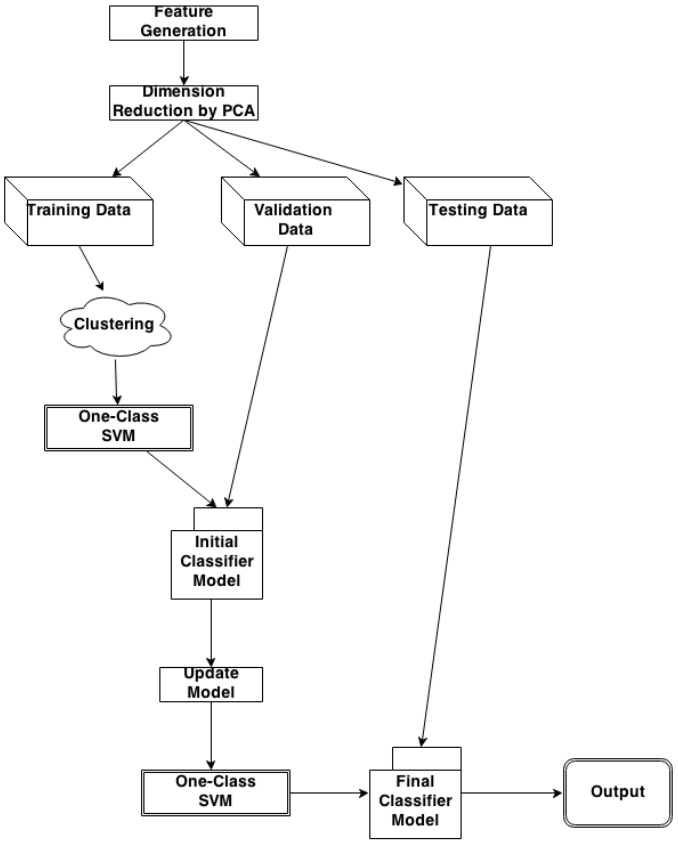

Fig. 1 Architectural model

A. Pre-Processing

The KDD dataset [2] consist of numeric and non-numeric data. First, non-numeric data were mapped to numeric data based on their ranking for each attribute. Among numeric data, continuous values were also converted to discontinuous values for the purpose of calculating 'information gain'. The feature selection can be done based on information gain and gain ratio, which signifies how much a feature contributes towards the resulting class for classification.

TABLE I
TRAINING, VALIDATION, AND TESTING SETS USED IN KDD CUP '99 DATASET

Data Set	Types	Count	Distinct Count	Percentage of Unique Data
Training Set	Normal	972780	812735	83.55
	Attacks	0	0	0.00
Validation Set	Normal	972780	812735	83.55
	Attacks	3925651	262179	6.68
Testing Set	Normal	60593	57876	95.52
	Attacks	250436	29349	11.72

B. Sampling and Feature Selection

The KDD dataset has been known for consisting of redundant data so; we have used a 'distinct SQL query statement' to get the unique sample of each data based on 28 features we have selected. In KDD dataset, we have a full dataset of size 708 MB and a test dataset of size 45 MB. Here, we have used a full dataset for two purposes: one as a training set and other as a validation set. The training set consists of only normal instances where as complete full dataset is used as a validation set.

TABLE II
INFORMATION GAINS OF VARIOUS ATTRIBUTES IN KDD DATASET.

FullSet InfoGain							
average merit			average rank			no.	attribute
0.816	+-	0.001	1	+-	0	5	src_bytes
0.672	+-	0.001	2	+-	0	3	service
0.632	+-	0.001	3	+-	0	6	dst_bytes
0.519	+-	0.001	4.1	+-	0.3	4	flag
0.519	+-	0.001	4.9	+-	0.3	30	diff_srv_rate
0.51	+-	0.001	6	+-	0	29	same_srv_rate
0.476	+-	0.001	7	+-	0	33	dst_host_srv_count
0.438	+-	0.001	8	+-	0	34	dst_host_same_srv_rate
0.41	+-	0.001	9	+-	0	35	dst_host_diff_srv_rate
0.406	+-	0.001	10.1	+-	0.3	38	dst_host_serror_rate
0.405	+-	0.001	10.9	+-	0.3	12	logged_in
0.398	+-	0.001	12	+-	0	39	dst_host_srv_serror_rate
0.393	+-	0.001	13	+-	0	25	serror_rate
0.384	+-	0.001	14	+-	0	23	count
0.379	+-	0.001	15	+-	0	26	srv_serror_rate
0.271	+-	0	16	+-	0	37	dst_host_srv_diff_host_rate
0.198	+-	0	17	+-	0	32	dst_host_count
0.189	+-	0	18	+-	0	36	dst_host_same_src_port_rate
0.142	+-	0	19	+-	0	31	srv_diff_host_rate
0.094	+-	0.001	20	+-	0	24	srv_count
0.088	+-	0	21	+-	0	41	dst_host_srv_rerror_rate
0.063	+-	0	22	+-	0	2	protocol_type
0.057	+-	0	23	+-	0	27	rerror_rate
0.052	+-	0.001	24	+-	0	40	dst_host_rerror_rate
0.052	+-	0	25	+-	0	28	srv_rerror_rate
0.036	+-	0	26	+-	0	1	duration
0.012	+-	0	27	+-	0	10	hot
0.01	+-	0	28	+-	0	8	wrong_fragment
0.007	+-	0	29	+-	0	13	num_compromised
0.004	+-	0	30	+-	0	16	num_root
0.002	+-	0	31	+-	0	19	num_access_files
0.001	+-	0	32	+-	0	22	is_guest_login
0.001	+-	0	33	+-	0	17	num_file_creations
0.001	+-	0	34	+-	0	15	su_attempted
0	+-	0	35	+-	0	14	root_shell
0	+-	0	36.1	+-	0.3	18	num_shells
0	+-	0	37.6	+-	0.49	7	land
0	+-	0	38.1	+-	1.58	11	num_failed_logins
0	+-	0	38.8	+-	0.87	21	is_host_login
0	+-	0	39.6	+-	0.49	20	num_outbound_cmds
0	+-	0	40.8	+-	0.6	9	urgent

The data distribution of normal and attack data is shown in Table I, which also gives the overview of the redundancy of the data used in KDD dataset. We have calculated information gains as shown in Table II, by means of which we have been able to figure out 28 features out of 41 that contributed mostly towards discriminating the given class from the others.

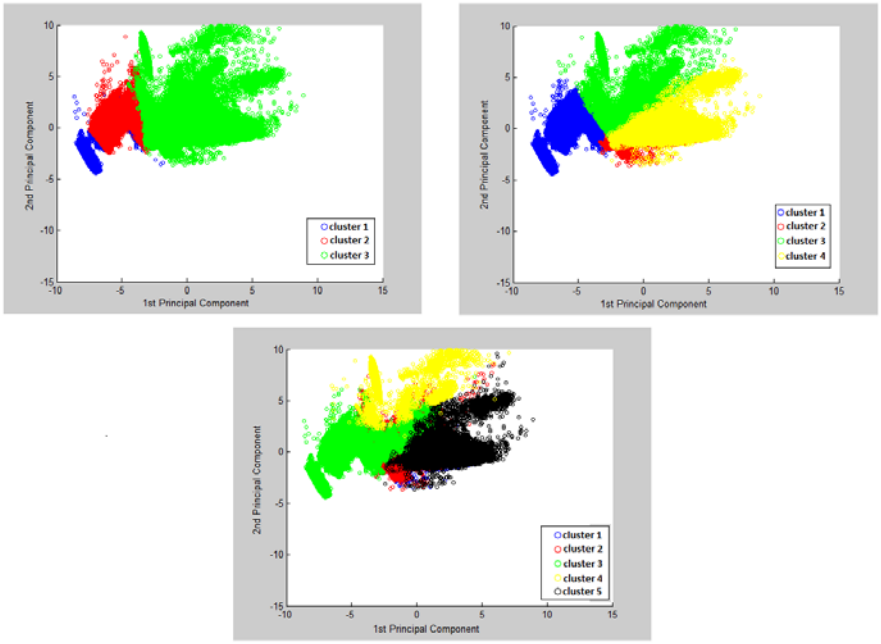

Fig. 2 K-means clustering results when k=3, 4, and 5 (only the first two principal components are shown).

C. Model Building

Once feature selection procedure is completed, the training feature set of normal network data is divided into clusters to represent the normal model by means of k-means clustering. The number of clusters to represent normal data is decided once the cluster analysis is done based on the silhouette value with its highest value obtained. Once 'k' is determined, Euclidean distance is used to measure the distance from k-centers to each point in training and validation sets. Thus, distance values from each k-center to training and testing data points are sent to single-class support vector machine (SVM).

In single-class SVM [17], we have used trial and error method using all the available kernel types and varying the degree and gamma values in the kernel function. The results of these two techniques, clustering and single-class SVM are combined and updated to create a final Model. We update our model with the true negatives i.e. 'normal instances' detected in our 1st model. Thus, updating our model helps to detect the anomaly as we have extracted out the probable intersected zone of attack and normal instances. Once we get our final model using the effective normal points from the selected validation feature set, we then test the testing feature set to predict the final output of our detection system.

IV. Results and Analyses

A. Clustering Analysis on Training Set

We have performed k-means clustering by using different 'k' starting from 2. The 'k' is chosen based on the highest mean of silhouette value obtained, which is 0.64 when k=5. Fig. 2 shows the clustering of the training set using different values for 'k' from 3 to 5 using the first two principal components. We observed that as we increase the value for 'k', the silhouette value started to increase and then started to decrease, which is depicted in Fig. 3.

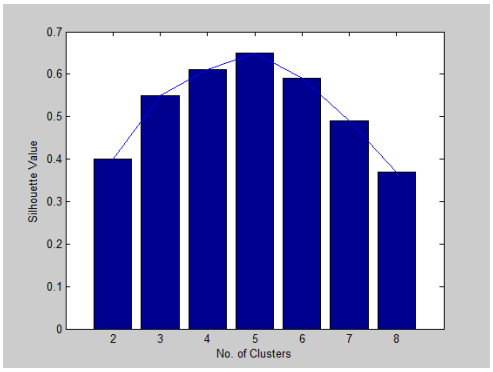

Fig. 3 Silhouette value for different number of clusters (k)

After conducting principal component analysis (PCA), we have chosen to proceed with first 8 principal components out of total features used, as it covered more than 99.00% of variance of data with these first 8 components which can be seen in Table III. We then start it off by performing clustering analysis in the training set and try to maximize the normal cases (i.e. in our case true negatives) in the validation set. The data distribution of the 1st and the 2nd principal component of training set and validation set can be seen in Fig. 4 and Fig. 5.

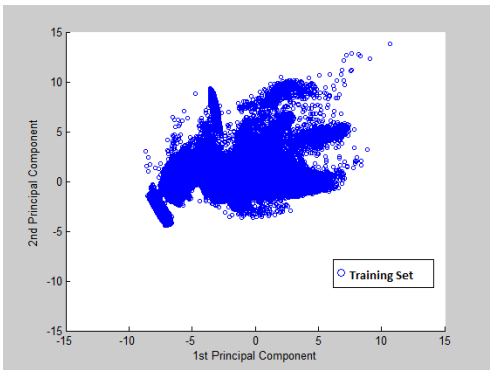

Fig. 4 Distribution of normal instances in training set

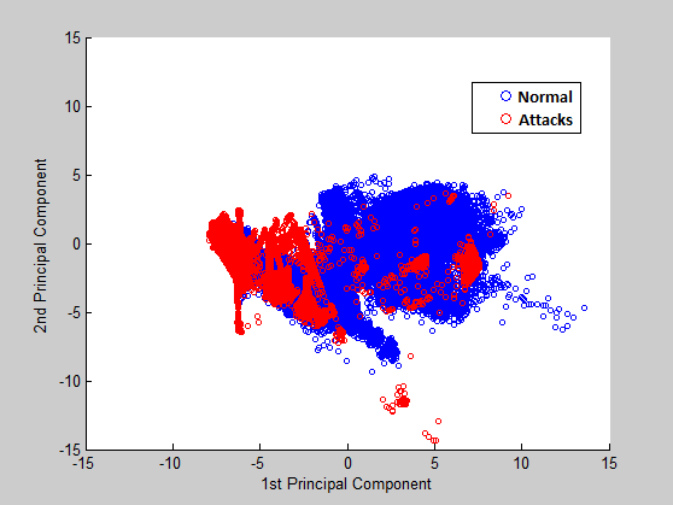

Fig. 5 Data distribution in validation set

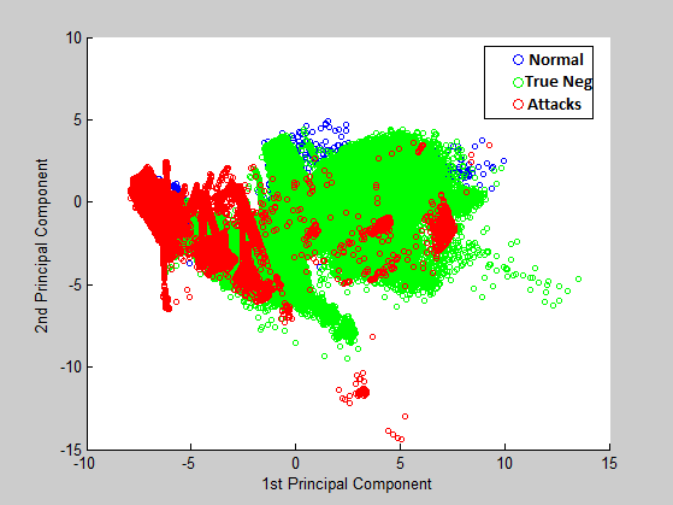

Fig. 6 Modeling normal instances (true negatives) in validation set

TABLE III
CUMULATIVE SCORES FOR THE FIRST 10 PRINCIPAL COMPONENTS

	Cumulative Scores i.e. (cumsum(latent)/sum(latent))		
Components	Training Set	Validation Set	Testing Set
Principal Component 1	0.4647	0.6802	0.7158
Principal Component 2	0.6454	0.7876	0.8032
Principal Component 3	0.7831	0.8607	0.8784
Principal Component 4	0.892	0.9241	0.9191
Principal Component 5	0.9546	0.9543	0.9493
Principal Component 6	0.9792	0.9776	0.9755
Principal Component 7	0.9864	0.9856	0.9847
Principal Component 8	0.9925	0.9917	0.9911
Principal Component 9	0.9945	0.9962	0.9953
Principal Component 10	0.9962	0.9972	0.9967

B. Modeling of Normal Instances

We achieved our best results where, we were able to model 90% (731,466 out of 812,735) of the total normal data of validation set as true negatives. Since, we have used training set which is extracted from validation set, containing only normal instances; the modeling of normal instances should have been 100% of the training dataset but, our model only uses these 90% because as we have used first 8 principal components that allows a slight variation between training and validation representation set due to presence of additional attack instances in validation set. Another reason could be single-class SVM tries to avoid over-fitting and allows slack variables to come into play. After modeling the true negatives as shown in Fig. 6, we then feed the test set (Fig. 7) to calculate the output from our model.

TABLE IV
RESULTS USING DIFFERENT NUMBER OF CLUSTERS WITH RBF KERNEL

	Methods		
	RBF when k=3	RBF when k=4	**RBF when k=5**
True Positive (TP)	29,230	29,243	**28,954**
True Negative (TN)	31,887	31,907	**40,111**
False Positive (FP)	15,989	15,969	**7,765**
False Negative (FN)	119	106	**395**

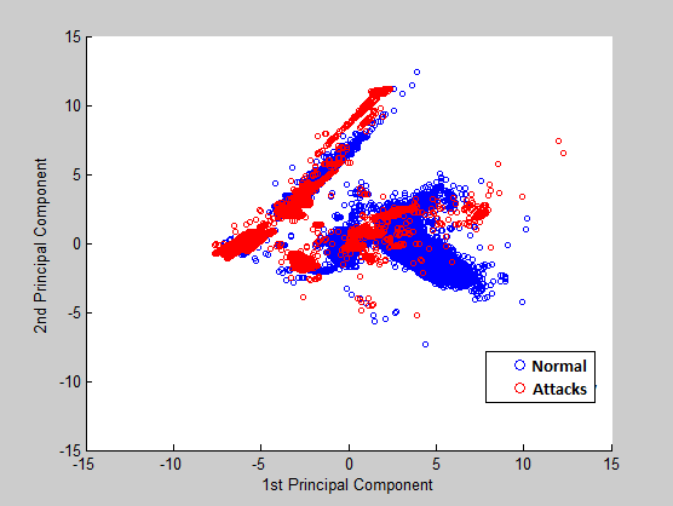

Fig. 7 Data distribution in testing set

C. Results

The results obtained in processing the KDD test set in our single-class SVM model using radial basis function (RBF) kernel can be seen in the following Table IV.

Looking from the overall perspective of accuracy, f1-score, precision and recall; we achieved our best result using 5 clusters (i.e., k=5) along with other parameters such as 'nu' set to 0.01, 'g' set to 0.035 (i.e., 1/28 where, 28 is the number of features used) and with RBF kernel.

The attacks that were undetected by our approach using different cluster value are provided in Table V. We can see that attacks such as 'apache2' and 'mscan' were detected when we used cluster 3 and 4, but it resulted in bit higher false positive rate. The attacks such as 'buffer overflow', 'httptunnel', 'mailbomb', 'multihop', 'ps' and 'rootkit' were undetected while using 3 and 4 clusters, but they were detected when trying to lower the false positive rate while analyzing with 5 clusters. But it gave much better false positive rate which got reduced by almost half of what was detected while performing with 3 and 4 clusters.

http://dx.doi.org/10.15242/IIE.E0414077

TABLE V
UNDETECTED ATTACKS WHEN USING DIFFERENT CLUSTER NUMBERS (K).

No. Of Attacks	Class ID	Class Name	No. of attacks undetected when k=5	No. of attacks undetected when k=4	No. of attacks undetected when k=3
1	2	apache2.	0	1	1
2	4	buffer_overflow	1	0	0
3	7	guess_passwd.	233	57	63
4	8	httptunnel.	6	0	0
5	13	mailbomb.	61	0	0
6	14	mscan.	0	5	7
7	15	multihop.	1	0	0
8	25	ps.	1	0	0
9	26	rootkit.	2	0	0
10	31	snmpgetattack.	89	42	47
11	38	warezmaster.	1	1	1
Total no. of attacks undetected			395	106	119
Total no. of distinct attacks undetected			9	5	5

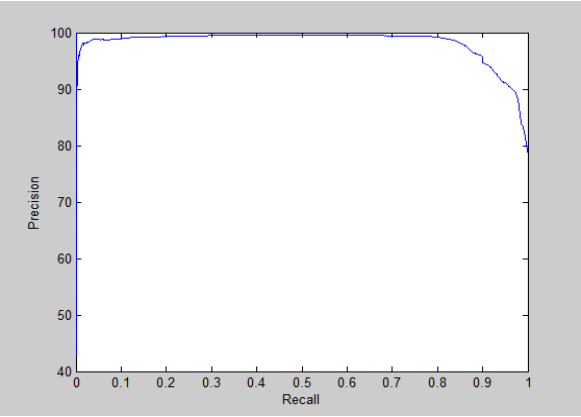

Fig. 8 Precision-recall graph (with k=5 and RBF kernel)

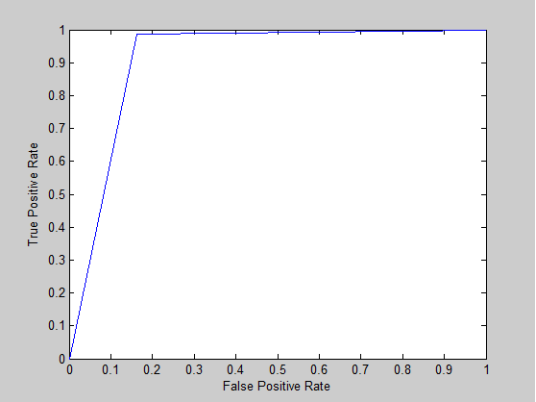

Fig. 9 ROC curve (with k=5 and RBF kernel)

Fig. 8 provides precision-recall graph. The best precision and recall obtained by our model is 78.85% and 0.98 respectively. But, it can be inferred from the graph that we can improve the 'precision' to 98% if we compromise the 'recall' by lowering it down to 0.8. And if we lower the 'recall' to 0.9 the 'precision' stays around 95%. The receiver operating characteristic (ROC) curve obtained is shown in Fig. 9. It has the area under the curve (AUC) of 0.9122. This means that with the probability of 0.9122, our method can rank a randomly chosen attack higher than a randomly chosen normal instance.

V. CONCLUSION

In this paper, we have proposed an anomaly-based IDS using outlier detection. We use the normal instances to build a model and detect the attack ones which fall out of this model. Experimental results shows our scheme is effective, and it can form a basis for a robust IDS against newly advanced intrusion attacks.

REFERENCES

[1] S. Suthaharan and T. Panchagnula, "Relevance feature selection with data cleaning for intrusion detection system," in *Proc. 2012 IEEE Southeastcon Conference (SECon)*, 2012, pp. 1-6.
[2] https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
[3] F. Gharibian and A. A. Ghorbani, "Comparative study of supervised machine learning techniques for intrusion detection," in *Proc. 5th Annual Conference on Communication Networks and Services Research, (CNSR)*, 2007, pp. 350-358.
[4] S. E. Smaha, "Haystack: An intrusion detection system," in *Proc. 4th Aerospace Computer Security Applications Conference (ACSAC)*, 1988, pp. 37-44.
[5] H. S. Javitz and A. Valdes, "The NIDES statistical component: Description and justification," Technical Report, Computer Science Lab., SRI-Int., Menlo Park, California, USA, 1994.
[6] I. H. Witten and E. Frank, "Data Mining Practical Machine Learning Tools and Techniques," Morgan Kalfman, San Francisco, California, USA, 2005.
[7] R. Groth, "Data Mining: Building Competitive Advantage," Prentice Hall, Upper Saddle River, New Jersey, USA, 2000.
[8] H. Sarvari and M. M. Keikha, "Improving the accuracy of intrusion detection systems by using the combination of machine learning approaches," in *Proc. 2010 International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, 2010, pp. 334-337.
[9] M. A. Faisal, Z. Aung, J. Williams, and A. Sanchez, "Data-stream-based intrusion detection system for advanced metering infrastructure in smart grid: A feasibility study," *IEEE Systems Journal*, in press, 2014. http://dx.doi.org/10.1109/JSYST.2013.2294120
[10] M. A. Faisal, Z. Aung, J. Williams, and A. Sanchez, "Securing advanced metering infrastructure using intrusion detection system with data stream mining," in *Intelligence and Security Informatics*, Lecture Notes in Computer Science Vol. 7299, 2012, pp. 96-111. http://dx.doi.org/10.1007/978-3-642-30428-6_8
[11] D. Kershaw, Q. Gao, and H. Wang, "Anomaly-based network intrusion detection using outlier subspace analysis: A case study," in *Advances in Artificial Intelligence*, Lecture Notes in Computer Science Vol. 6657, 2011, pp. 234-239. http://dx.doi.org/10.1007/978-3-642-21043-3_28
[12] W. Da and H. S. Ting, "Distributed intrusion detection based on outlier mining," in *Proc. 2012 International Conference on Communication, Electronics and Automation Engineering (ICCEAE)*, Advances in Intelligent Systems and Computing Vol. 181, 2013, pp. 343-348.
[13] N. Devarakonda, S. Pamidi, V. V. Kumari, and A. Govardhan,"Outliers Detection as Network Intrusion Detection System Using Multi Layered Framework," in *Advances in Computer Science and Information Technology*, Communications in Computer and Information Science Vol. 131, 2011, pp. 101-111. http://dx.doi.org/10.1007/978-3-642-17857-3_11
[14] J. Zhang and M. Zulkernine, "Anomaly based network intrusion detection with unsupervised outlier detection," in *Proc. 2006 IEEE International Conference on Communications (ICC)*, 2006, vol. 5, pp. 2388-2393. http://dx.doi.org/10.1109/ICC.2006.255127
[15] H. P. Kriegel, P. Kroger, and A. Zimek, "Outlier detection techniques," in *Tutorial, 2010 SIAM International Conference on Data Mining (SIAM)*, 2010.
[16] P. Manandhar and Z. Aung, "Towards practical anomaly-based intrusion detection by outlier mining on TCP packets," submitted for review, 2014.
[17] B. Schölkopf, R. Williamson, A. Smola, J. Shawe-Taylor, J. Platt, "Support vector method for novelty detection," *Advances in Neural Information Processing Systems*, vol. 12, pp. 582-588, 2000.