Unsupervised clustering approach for network anomaly detection

Iwan Syarif [1,2], Adam Prugel-Bennett[1], Gary Wills[1]

[1] School of Electronics and Computer Science, University of Southampton, UK
{is1e08,apb,gbw}@ecs.soton.ac.uk
[2] Eletronics Engineering Polytechnics Institute of Surabaya, Indonesia
iwanarif@eepis-its.edu

Abstract. This paper describes the advantages of using the anomaly detection approach over the misuse detection technique in detecting unknown network intrusions or attacks. It also investigates the performance of various clustering algorithms when applied to anomaly detection. Five different clustering algorithms: k-Means, improved k-Means, k-Medoids, EM clustering and distance-based outlier detection algorithms are used. Our experiment shows that misuse detection techniques, which implemented four different classifiers (naïve Bayes, rule induction, decision tree and nearest neighbour) failed to detect network traffic, which contained a large number of unknown intrusions; where the highest accuracy was only 63.97% and the lowest false positive rate was 17.90%. On the other hand, the anomaly detection module showed promising results where the distance-based outlier detection algorithm outperformed other algorithms with an accuracy of 80.15%. The accuracy for EM clustering was 78.06%, for k-Medoids it was 76.71%, for improved k-Means it was 65.40% and for k-Means it was 57.81%. Unfortunately, our anomaly detection module produces high false positive rate (more than 20%) for all four clustering algorithms. Therefore, our future work will be more focus in reducing the false positive rate and improving the accuracy using more advance machine learning techniques.

Keywords: k-Means, EM clustering, k-medoids, intrusion detection system, anomaly detection, outlier detection

1 Introduction

Intrusion detection is a process of gathering intrusion-related knowledge occurring in the process of monitoring events and analyzing them for signs of intrusion [1][5]. There are two basic IDS approaches: misuse detection (signature-based) and anomaly detection. The misuse detection system uses patterns of well-known attacks to match and identify known intrusions. It performs pattern matching between the captured network traffic and attack signatures. If a match is detected, the system generates an alarm. The main advantage of the signature detection paradigm is that it can accurately detect instances of known attacks. The main disadvantage is that it lacks the ability to detect new intrusions or zero-day attacks [2][3].

The anomaly detection model works by identifying an attack by looking for behaviour that is out of the normal. It establishes a baseline model of behaviour for users and components in a computer or network. Deviations from the baseline cause alerts that direct the attention of human operators to the anomalies [3][4][5]. This system searches for anomalies either in stored data or in the system activity. The main advantage of anomaly detection is that it does not require prior knowledge of an intrusion and thus can detect new intrusions. The main disadvantage is that it may not be able to describe what constitutes an attack and may have a high false positive rate [2][3][4].

2 Clustering Algorithms

Clustering is a technique for finding patterns in unlabelled data with many dimensions. Clustering has attracted interest from researchers in the field of intrusion detection [5,6]. The main advantage of clustering algorithm is the ability to learn from and detect intrusions in the audit data without explicit descriptions (intrusion signatures) which usually provided by security experts.

There are two different approaches to clustering-based anomaly detection. The first approach is called unsupervised clustering where the anomaly detection model is trained using unlabelled data that consists of both normal as well as attack traffics. The second approach is called semi-supervised clustering where the model is trained using normal data only to build a profile of normal activity. The idea behind the first approach is that anomalous or attack data forms a small percentage of the total data. Based on this assumption, anomalies and attacks can be detected based on cluster sizes, large clusters correspond to normal data and the rest of the data points, which are outliers, correspond to attacks [5,6,7].

Eskin et al.[7] and Portnoy et al.[8] introduced the use of clustering algorithms to detect network traffics anomalies. Their algorithm starts with an empty set of clusters and generates the clusters with a single pass through the dataset. First, the data set needs to be normalized (convert into Z form), it then computes the distance between each new instance and its centroid. The cluster with the shortest distance is selected, and if that distance is less than or equal W (the cluster width) then the instance is assigned to that cluster, otherwise it creates a new cluster with this instance as the centre.

Based on the previous assumption, the normal clusters (clusters which contain normal data) will have much larger number of instances compare to anomalous cluster. Eskin and Portnoy proposed an idea to model the normal traffic by selecting X percentage (where X is an input parameter) of clusters containing the largest number of instances and label them as normal and then they labeled the rest of the clusters as anomalous which might be considered as an intrusion/attack.

In this paper, we implement and compare the performance of five different clustering algorithms in our anomaly detection module which are k-Mean, improved k-Mean, k-Medoids, EM clustering and distance-based outlier detection algorithms.

2.1 k-Means

k-Means which is firstly proposed by James MacQueen, is a well-known and widely used clustering algorithm. k-Means is one of the simplest clustering algorithms in machine learning which can be used to automatically recognize groups of similar instances/items/objects/points in data training. The algorithm classifies instances to a pre-defined number of clusters specified by the user (e.g. assume *k* clusters). The first important step is to choose a set of k instances as centroids (centres of the clusters) randomly, usually choose one for each cluster as far as possible from each other. Next, the algorithm continues to read each instance from the data set and assigns it to the nearest cluster. There are some methods to measure the distance between instance and the centroid but the most popular one is Euclidian distance. The cluster centroids are always recalculated after every instance insertion. This process is iterated until no more changes are made. The k-Means algorithm is explained in this following pseudo code.

```
1. Select the total number of clusters (k)
2. Choose random k points and set as centroid
3. Calculate the distance from each instance to all
   centroids using Euclidean method
4. Assign each instance to the closest centroid
5. Recalculate the positions of the centroids
6. Repeat step 3-5 until the centroids do not change
```

2.2 k-Medoids

k-Medoids is a clustering algorithm similar to k-Means, which attempts to minimize the distance between points and its centre (centroid). A medoid is a data point which acts as an exemplar for all other data points in the cluster. The k-Means algorithm is very sensitive to outliers because if there is an object with a very large value, the data distribution may be biased or distorted [13]. In this case, k-Medoids is more robust to noise and outliers because in this algorithm the partitioning method is performed based on the principle of minimizing the sum of dissimilarities between each object in a cluster. The pseudo code of k-Medoids is explained below [13]:

```
1. Input a data set D consists of n objects
2. Input the number of clusters K
3. Select k objects randomly as the initial cluster
   centres or cluster medoids
4. Assign each object to the cluster with the nearest
   medoid
5. Calculate the total distance between the object and
   its cluster medoid
6. Swap the medoid with non-medoid object
7. Recalculate the positions of the k medoids
8. Repeat 4-7 until the medoids become fixed
```

2.3 EM Clustering

Expectation Maximization (EM) clustering is a variant of k-Means clustering and is widely used for density estimation of data points in an unsupervised clustering [14]. In the EM clustering, we use an EM algorithm to find the parameters which maximize the likelihood of the data, assuming that the data is generated from k normal distributions. The algorithm learns both the means and the covariance of the normal distributions. This method requires several inputs which are the data set, the total number of clusters, the maximum error tolerance and the maximum number of iteration.

The EM can be divided into two important steps which are Expectation (E-step) and Maximization (M-step). The goal of E-step is to calculate the expectation of the likelihood (the cluster probabilities) for each instance in the dataset and then re-label the instances based on their probability estimations. The M-step is used to re-estimate the parameters values from the E-step results. The outputs of M-step (the parameters values) are then used as inputs for the following E-step. These two processes are performed iteratively until the results convergence. The mathematical formulas of EM clustering are described in [14][15] and the pseudo codes can be found in [15].

2.4 Outlier Detection Algorithms

Outlier detection is a technique to find patterns in data that do not conform to expected behaviour [6]. Most of the clustering algorithms do not assign all points to clusters but account for noise objects, in other words clustering algorithms are optimized to find clusters rather than outliers. Outlier detection algorithms look for outliers by applying one of the clustering algorithms and retrieve the noise set, therefore the performance of outlier detection algorithms depends on how good the clustering algorithm captures the structure of clusters.

Outlier detection can be divided into two approaches: distance-based outlier detection and density-based outlier detection. The first method, distance-based outlier detection algorithms, works on the assumptions that the normal data objects have a dense neighbourhood and the outliers are far apart from their neighbours. The second method works on assumptions that the density around a normal data object is similar to the density around its neighbours while the density around an outlier is considerably different to the density around its neighbours. The density-based approach compares the density around a point with the density around its local neighbours by computing an outlier score. This paper focuses on distance-based outlier detection algorithm only.

The distance-based outlier detection approach, which is based on the nearest neighbour algorithm was first introduced by Ng et al [16] and implements a well-defined distance metric to detect outliers, the greater the distance of the object to its neighbour, the more likely it is to be an outlier. This method calculates the distance between each pair of objects using a nested loop (NL) algorithm and then the objects which are far away from the majority are signed as outliers. The mathematical formulas of distance-based outlier detection methods and their pseudo codes are described in more details [16][17].

3 Experimental Setup

The following section describes the intrusion data sets used in the experiment, the performance metric used to evaluate the proposed system and the experimental settings and its results.

3.1 Intrusion Dataset

One of the most widely used data sets for evaluating IDS is the DARPA/Lincoln Laboratory off-line evaluation dataset or IDEVAL[9]. IDEVAL is the most comprehensive test set available today and it was used to develop the 1999 KDD Cup data mining competition [10]. In this experiment, we use the NSL-KDD intrusion data, which was provided to solve some problems in KDD'99, particularly that its training and test sets contained a huge number of redundant records with about 78% and 75% of the records being duplicated in the training and test sets, respectively. This may cause the classification algorithms to be biased towards these redundant records and thus prevent it from classifying other records [11].

Table 1. List of intrusions in training and testing data

Intrusions which exist in both training and testing data	**Intrusions which only exist in testing data**
back, buffer_overflow, ftp_write, guess_passwd, imap, ipsweep, land, loadmodule, multihop, neptune, nmap, phf, pod, portsweep, rootkit, satan, smurf, spy, teardrop, warezclient, warezmaster	apache2, httptunnel, mailbomb, mscan, named, perl, processtable, ps, saint, sendmail, snmpgetattack, snmpguess, sqlattack, udpstorm, worm, xlock, xsnoop, xterm

The intrusion data set consists of forty different intrusions classified into four main categories: DoS (Denial of Service), R2L (Remote to Local Attack), U2R (User to Root Attack) and Probing Attack. The training dataset consists of 25,191 instances and the testing dataset consists of 11,950 instances. The testing data set has many intrusions, which do not exist in the training data, as shown in Table 1.

3.2 Performance Metric

We use accuracy rate and false positive rate as the performance criteria based on the following metric shown in Table 2 below.

Table 2. Performance metric

		Actual Result	
		Intrusion	Normal
Predicted Result	Intrusion	True Positive (TP)	False Positive (FP)
	Normal	False Negative (FN)	True Negative (TN)

True Positive (TP) is a condition when an actual attack is successfully detected by the IDS. True Negative (TN) is a condition where normal traffic is detected as a normal, in other words there is no attack nor IDS alert is raised. False Positive (FP) is an alert that indicates that an attack is in progress when in fact there was no such attack. False Negative (FN) is a failure of IDS to detect an actual attack [12]. The accuracy rate and false positive rate are measured using the following formulae:

$$Accuracy\ rate = \frac{\mathrm{TP+FN}}{\mathrm{TP+TN+FP+FN}}\,(1),\ False\ Positive = \frac{FP}{\mathrm{TP+FP}}\,(2)$$

3.3 Misuse Detection Module

Our proposed misuse detection module consists of five phases: feature extraction, dimensionality reduction, classification algorithms, apply model and performance measurement & analysis as explained in Figure 1 below.

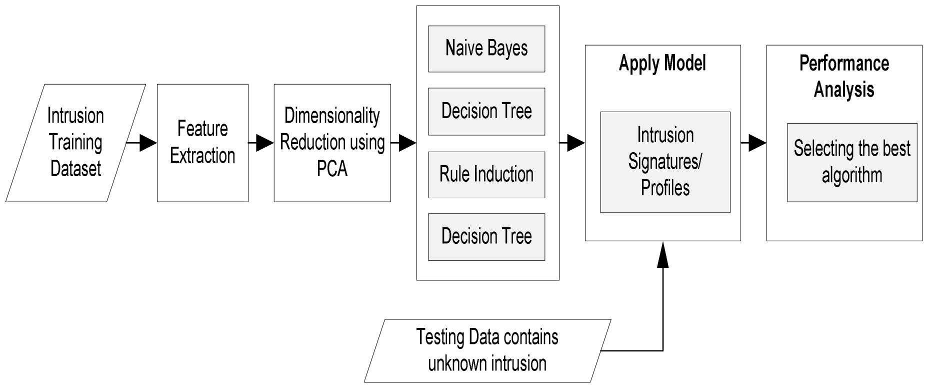
Intrusion Training Dataset
Feature Extraction
Dimensionality Reduction using PCA
Naive Bayes
Decision Tree
Rule Induction
Decision Tree
Apply Model
Intrusion Signatures/ Profiles
Performance Analysis
Selecting the best algorithm
Testing Data contains unknown intrusion

Fig. 1. Misuse Detection System Design

We use Principal Component Analysis (PCA) as a dimensionality reduction algorithm. Dimensionality reduction is the process of reducing the number of random variables under consideration, which is very useful to reduce the computational cost as well as to avoid over-fitting. After that, we apply four different classifiers (naïve Bayes, decision tree, rule induction and nearest neighbour) into the misuse detection module in order to find the best method in detecting intrusion based on accuracy, false positive and speed (computation time).

3.4 Anomaly Detection Module

We designed the anomaly detection module as shown in Figure 2 below. This module implements several unsupervised clustering algorithms which do not required labeled dataset. In the feature extraction module we select only numerical data and handle missing value, then we transform the data into normal form. Normalization is a popular method used to convert all attributes/variables to a common scale with an average of zero and standard deviation of one.

Intrusion data set consists of 41 attributes, which may have different scale and different distribution. Some attributes may have a wide range of values while other attributes are very narrowly distributed. These differences in distribution make it difficult to measure the similarities or significant differences between variables/ categories in the data sets. To solve this problem, we will convert the data set into normal form. Normalization (or standardization) is the most commonly used method to convert all attributes to a common scale with an average of zero and standard deviation of one.

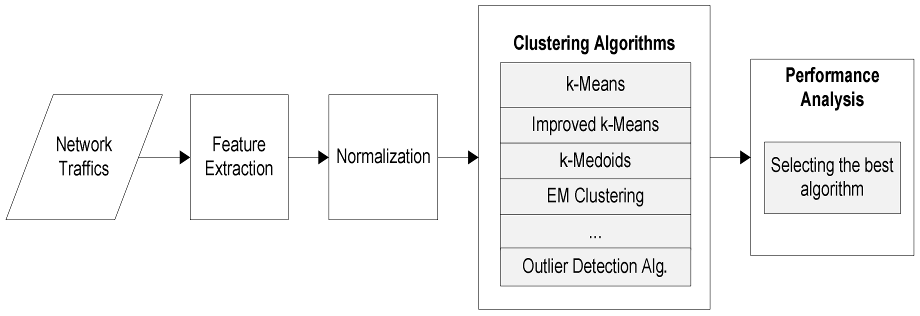

Fig. 2. Anomaly Detection System Design

Given a training dataset, the average and standard deviation feature vectors are calculated:

$$\text{avg\_vector}[\text{j}] = \frac{1}{N}\sum_{i=1k=0}^{N} \text{instance}_{\text{i}}[\text{j}]$$

$$\text{std\_vector}[\text{j}] = \sqrt{\frac{1}{N-1}\sum_{i=1}^{N}(instance_i[j] - avg\_vector[j])^2}$$

where vector[j] is the j^th element (feature) of the vector. Then each instance (feature vector) in the training set is converted as follows:

$$new\_instance[j] = \frac{instance[j] - avg\_vector[j]}{\text{std\_vector[j]}},$$

4 Experimental Results and Discussion

The following section discusses and analyses the results of both the misuse detection module and the anomaly detection module.

4.1 Misuse Detection Module

In the first experiment, we use only the training data which contains around 22 different types of intrusions and apply 10-fold cross validation in the misuse detection module. The results are shown in Table 3 below.

Table 3. Misuse Detection performance using 10 fold cross validation

Algorithm	Accuracy	False Positive
Naïve Bayes	89.59%	10.60%
Nearest Neighbour	99.44%	0.60%
Rule Induction	99.58%	0.40%
Decision Tree	99.56%	0.40%

Table 3 shows that misuse detection achieves very good results when detecting known intrusion. Three of the four algorithms (nearest neighbour, rule induction and decision tree) achieve more than 99% accuracy and the false positive rates are less than 1%.

In the second experiment, we use a testing data to evaluate the performance of intrusion model in the misuse detection module. The testing data contains 22 types of known intrusions and 18 types of unknown intrusions. The results of the second experiment are shown in Table 4 below.

Table 4. Misuse Detection performance using testing data

Algorithm	Accuracy	False Positive
Naïve Bayes	55.77%	34.80%
Nearest Neighbour	62.84%	20.90%
Rule Induction	63.69%	18.00%
Decision Tree	63.97%	17.90%

Table 4 shows that the misuse detection module does not perform well in detecting data which contains a large number of unknown intrusions where the highest accuracy is only 63.97% and the lowest false positive is 17.90%.

4.2 Anomaly Detection Module

We applied five unsupervised clustering algorithms which are k-Means, improved k-Means, k-Medoids, Expectation-Maximization (EM) clustering and distance-based outlier detection algorithm into the anomaly detection module and used an unlabelled dataset as an input and the results are shown in Table 5 below.

Compared to the misuse detection module which has an accuracy of only 63.97% (evaluated using testing data), the anomaly detection module has a better performance in detecting novel intrusion. These clustering algorithms are able to detect intrusions without prior knowledge. In this experiment, the distance-based outlier detection algorithm achieves the best accuracy with 80.15%, followed by EM clustering 78.06%, k-Medoids with 76.71%, improved k-Means 65.40% and k-Means 57.81%.

Table 5. Anomaly Detection accuracy using clustering algorithms

Algorithm	Accuracy	False Positive
k-Means	57.81%	22.95%
improved k-Means	65.40%	21.52%
k-Medoids	76.71%	21.83%
EM clustering	78.06%	20.74%
Distance-based outlier detection	80.15%	21.14%

Unfortunately, all of these algorithms have quite high positive rates with more than 20%. This means that there are around 20% of normal traffics predicted as intrusions. Because of the high positive rates, this anomaly detection module would not be viable in the real world. Therefore, our future work will be focused on how to reduce the false positive while still improving the accuracy.

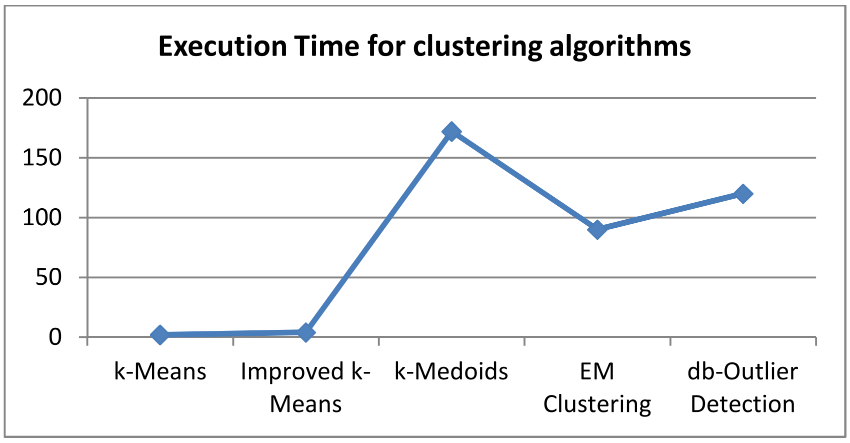

Fig. 3. Clustering algorithms execution time

Even though the distance-based outlier detection algorithm outperforms the other four algorithms in accuracy, unfortunately its computation time is relatively high. The k-Means algorithm is the fastest but its accuracy is the worst (57.81%), in contrast the k-Medoids algorithm is the slowest but its accuracy is relatively high (76.71%).

Since the distance-based outlier detection algorithm has achieved the highest accuracy, we continue our experiment by applying this algorithm in the anomaly detection module. Now we classify the intrusion dataset into four types of intrusion which are probing attacks, DoS attacks, R2L attacks and U2R attacks. The results are shown in Figure 4 below.

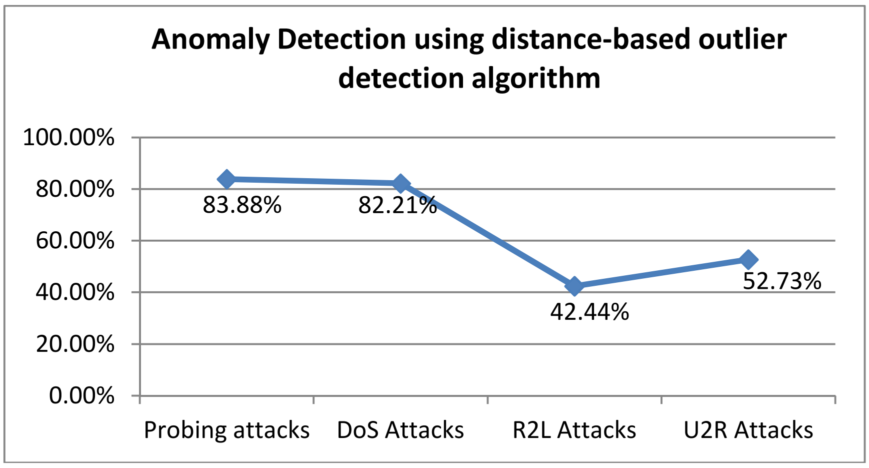

Fig. 4. Anomaly detection using distance-based outlier detection algorithm

This experiment shows that the distance-based outlier detection algorithm is able to detect probing attacks with 83.8% accuracy and DoS attacks with 82.21% accuracy. Unfortunately, this algorithm failed to accurately detect R2L attack (42.44%) and U2R attacks (52.73%). One reason is that the R2L attacks and U2R attacks have very similar behaviour with normal traffics which makes them very difficult to distinguish. Furthermore, the number of R2L and U2R attacks in intrusion dataset is very small compare to the whole data set. The number of R2L attacks is only 0.83% and U2R is only 0.04%.

5 Conclusions

Our experiment shows that the misuse detection technique achieves a very good performance with more than 99% accuracy when detecting known intrusion but it fails to accurately detect data set with a large number of unknown intrusions where the highest accuracy is only 63.97%. In contrast, the anomaly detection approach shows promising results where the distance-based outlier detection method outperforms the other three clustering algorithms with the accuracy of 80.15%, followed by EM clustering (78.06%), k-Medoids (76.71%), improved k-Means (65.40%) and k-Means (57.81%). Further experiment shows that the distance-based outlier detection performs very well in detecting probing attacks (83.88%) and DoS attacks (82.21%) but it fails to detect R2L attacks (42.44%) and U2R attacks (52.73%). Unfortunately, our anomaly detection module produces high positive rate (more than 20%) for all four clustering algorithms. Therefore, our future work will be focused in reducing the false positive rate and improving the accuracy using more advance machine learning techniques.

References

1. Gudadhe, M., Prasad, P., Wankhade, K.: A new data mining based network intrusion detection model. In: International Conference on Computer & Communication Technology (ICCCT'10), pp. 731-735. (2010)
2. Panda, M., Patra, M.R.: Ensemble of Classifiers for Detecting Network Intrusion. In: International Conference on Advances in Computing, Communication and Control (ICAC3'09), pp. 510-515. (2009)
3. Garcia-Teodoro,P., Diaz-Verdejo,J., Macia-Fernandez,G., Vazquez,E.: Anomaly-based network intrusion detection: Techniques, systems and challenges. Computer & Security, Volume 28, Issues 1-2, pp. 18-28. (2009)
4. Davis, J.J., Clark, A.J.: Data preprocessing for anomaly based network intrusion detection: A review. Computer & Security, Volume 30, Issues 6-7, pp 353-375. (2011)
5. Patcha,A., Park,Jung-Min: An Overview of Anomaly Detection Techniques: Existing Solutions and Latest Technological Trends. Computer Networks, Volume 51, Issue 12, 22 August 2007, pp. 3448-3470. (2007)
6. Chandola,V., Banarjee,A., Kumar, V.: Anomaly Detection: A Survey. ACM Computing Survey Journal, Volume 41 Issue 3, (2009)
7. Eskin,E., Arnold, A., Prerau,M., Portnoy, L., Stolfo, S.: A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data. In: Proceedings of the Seventeenth International Conference on Machine Learning. Morgan Kaufmann Publichsers Inc, pp. 255-262. (2000)
8. Portnoy, L., Eskin, E., Stolfo,S.: Intrusion detection with unlabeled data using clustering. In: Proceeding ACM Workshop on Data Mining Applied to Security. (2001)
9. DARPA Intrusion Detection Data Sets, http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html
10. KDD Cup'99 Intrusion Data Sets, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
11. Tavallaee M., Bagheri E., Lu W., Ghorbani A., "A Detailed Analysis of the KDD CUP 99 Data Set," *Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA)*, 2009.
12. Whitman, M.,E., Mattord, H.,J. Principles of Information Security. Course Technology, 4th Edition, ISBN: 1111138214. (2011)
13. Vermurugan,T., Santhanam, T.: Computational Complexity between K-Means and K-Medoids Clustering Algorithms for Normal and Uniform Distributions of Data Points. Journal of Computer Science 6 (3), pp. 363-368. (2010)
14. Seetha, J., Varadharajan,R., Vaithiyananthan, V. Unsupervised Learning Algorithm for Color Texture Segmentation Based Multiscale Image Fusion. European Journal of Scientific Research, ISSN 1450-216X, Vol 67, pp. 506-511 (2012)
15. Lu, Wei., Tong, Hengjian.: Detecting Network Anomalies Using CUSUM and EM Clustering. Advance in Computation and Intelligence, Volume 5821, pp. 297-308 (2009)
16. Knorr,E.M., Ng, R.T.: Finding intensional knowledge of distance-based outliers. In: VLDB'99: 25th Int. Conf. on Very Large Data Bases, pp. 211-222, San Francisco (1999)
17. Orair, G.H., Teixeira, C.H.C., Meira Jr., W., Wang, Ye, Parthasarathy, S. Distance-based Outlier Detection: Consolidation and Renewed Bearing. In: the 36th Int. Conf. om Very Large Data Bases, Singapore, (September 2010)