World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:1, 2009

Anomaly Detection and Characterization to Classify Traffic Anomalies
Case study: TOT Public Company Limited Network

O. Siriporn, and S. Benjawan

***Abstract*—**This paper represents four unsupervised clustering algorithms namely sIB, RandomFlatClustering, FarthestFirst, and FilteredClusterer that previously works have not been used for network traffic classification. The methodology, the result, the products of the cluster and evaluation of these algorithms with efficiency of each algorithm from accuracy are shown. Otherwise, the efficiency of these algorithms considering form the time that it use to generate the cluster quickly and correctly. Our work study and test the best algorithm by using classify traffic anomaly in network traffic with different attribute that have not been used before. We analyses the algorithm that have the best efficiency or the best learning and compare it to the previously used (K-Means). Our research will be use to develop anomaly detection system to more efficiency and more require in the future.

***Keywords*—**Unsupervised, Clustering, Anomaly, Machine Learning.

I. Introduction

THIS paper, a machine learning approach called clustering for classifying traffic statistics is used. Cluster analysis is one of the most methods for identifying classes amongst a group of objects, and has been used as a tool in many fields such as business, economics, bioinformatics and computer science. Cluster analysis group data objects base only on information found in the data that describe the objects and their relationships. The goal is that the objects with in a group similar (or related) to one another and different from (or unrelated to) within a group and the greater the difference between groups, the better or more distinct the clustering [6].

O. Siriporn is with Kasaetsart University, Bangkok, Thailand. She is now Assistant Professor and Database and Software Technology Researcher, Kasaetsart University, Bangkok, Thailand, She graduated from MS. (Computer Science), North Eastern Illinois University, 2527, B.Sc. (Statistical Computing), De Paul University, 2524 (e-mail: srp@ku.ac.th).

S. Benjawan is with the Computer Engineering Department, University of Kasaetsart, Bangkok, Thailand, on leave Database and Software Technology Research Laboratory. Work in TOT Public Company Limited, Bangkok, Thailand (e-mail: taan.2004@gmail.com).

Previous works by McGregor et al. [8] and Zander et al. [9] show that cluster analysis has the ability to group Internet traffic using only transport layer characteristics. In this paper, we confirm their observations by evaluating five clustering algorithms, namely K-Means [6], sIB, RandomFlatClustering, FarthestFirst, and FilteredClusterer, which to the best of our knowledge have not been previously applied to this problem. In section II we represent the related literature that related or useful for our work. In section III we describe about the clustering algorithms that we use for this paper. There are five algorithms include K-Means. In section IV we show the methodology in our work step by step start with traffic data collection, Data Preparing and Data Mining Software. In section V we describe the experimental results include evaluation method. In section VI and VII are the part of discussion and conclusion.

II. Related Literature

McGregor et al. introduce a new methodology (EM algorithm) for classification of the packet headers that divides the traffic into similar application types (single transaction, bulk transfer etc). They hypothesize the ability of using cluster analysis to group flows using transport layer attributes [8].

However, they do not evaluate the accuracy of the classification as well as which flow attributes produce the best results.

Erman et al. present classification of network traffic using port-based or payload-based analysis. They use two unsupervised clustering algorithms, namely K-Means and DBSCAN. They evaluate these two algorithms and compare them to the previously used AutoClass algorithm, using empirical Internet traces. The experimental results show that both K-Means and DBSCAN work very well and much more quickly than AutoClass. Their results indicate that although DBSCAN has lower accuracy compared to K-Means and AutoClass, DBSCAN produces better clusters.

Silva et al. developed for analysis in the network traffic, including the environment preparation for test and development, attribute selection, data reduction and data visualization are described. They choose network traffic data

from session TCP/IP packets, deriving from HTTP communication this data belong to a large dataset and found in several types so that they used clustering techniques base on neural network SOM are used for data reduction. They found the great advantage of SOM clustering is to allow analysis of the medium behavior of the monitored traffic and use RGCom tool. This application does not perform data reduction. They will improve the RGCom on the next challenges, adding option.

Münz et at. Present a novel flow-based anomaly detection scheme based on the K-mean clustering algorithm. Training data containing unlabeled flow records are separated into clusters of normal and anomalous traffic. The corresponding cluster centroids are used as patterns for computationally efficient distance-based detection of anomalies in new monitoring data.

III. Clustering Algorithm

A. K-Means Algorithm

The clustering algorithms is used to group unlabeled data. K-means is a prototype-based, partitional clustering technique that attempts' to find a user-specified number of clusters (K), which are represented by their centroids. This algorithm often used in many fields, because it is very simple algorithm and simple to understand.

The k-means algorithm takes the input parameter, k, and partitions a set of n objects into k clusters so that the resulting intracluster similarity is high but the intercluster similarity is low. Cluster similarity is measured in regard to the mean value of the objects in a cluster, which can be viewed as the cluster's center of gravity.

The k-means algorithm [13] proceeds as follows.

1. Defined number of clusters k.
2. Initialize the k cluster centroids. This can be done by arbitrarily dividing all objects into k clusters, computing their centroids, and verifying that all centroids are different from each other. Alternatively, the centroids can be initialized to k arbitrarily chosen, different objects.
3. Iterate over all objects and compute the distances to the centroids of all clusters. Assign each object to the cluster with the nearest centroid.
4. Recalculate the centroids of both modified clusters.
5. Repeat step 3 until the centroids do not change any more.

First, it randomly selects k of the objects. Each of which initially represents a cluster mean or center. For each of remaining objects, an object is assigned to the cluster to which it is the most similar, based on the distance between the object and the cluster mean. It then computes the new mean for each cluster. This process iterates until the criterion function converges. Typically, the squared-error criterion is used, defined as

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} |p - m_i|^2 \quad (1)$$

Where E is the sum of square-error for all objects in the database, p is the point in space representing a given object, and mi is the mean of the cluster Ci (both p and mi are multidimensional). This criterion tries to make the resulting k clusters as compact and as separate as possible. The k-means procedure is summarized as

B. FarthestFirst

The farthest-first traversal k-center algorithm (FFT) is a fast, greedy algorithm that minimizes the maximum cluster radius [23]. In FFT, *k* points are first selected as cluster centers. The remaining points are added to the cluster whose center is the closest. The first center is chosen randomly. The second center is greedily chosen as the point furthest from the first. Each remaining center is determined by greedily choosing the point farthest from the set of already chosen centers, where the furthest point (*x)* from a set (*S)* is defined as, $max_x\{min\{distance(x, j), j \in S\}\}$. The farthest-first traversal algorithm is shown below.

1. Randomly select first center.
2. For each remaining point calculate distance to the current center set. Select the point with maximum distance as new center.
3. Assign remaining points with calculate the distance to each cluster center and put it to the cluster with minimum distance.

C. k-Medoids or PAM

K-Medoids or PAM: Partitioning Around Medoids K-Medoids deals with the problem of outliers posed by k-Means. For example, if an object has a very different value from the other objects in a cluster then that object will distort the result with k-Means, because the mean of a cluster will have to be altered to deal with the object. k-Medoids is similar to k-Means except that the mean of each cluster is the object that is nearest to the "center" of the cluster [23].

The idea of partitioning with k-Medoids is to reduce the distance between all objects in a cluster and the most centrally located object in the cluster. The strategy is very similar to the k-Means strategy: first *k* of the *N* objects are inserted in *k* clusters arbitrarily and the medoid for each cluster is set equal to the object inserted in it. Each remaining object is inserted into the cluster whose medoid is most similar to it. Then each medoid *i* in a cluster may be swapped with one of the non-medoids *h*, as long as the total swapping cost TC_{ih} is negative. The total cost for swapping medoid *i* with non-medoid *h* is determined by using the function: $TC_{ih} = \Sigma_j C_{jih}$.

D. FilteredClusterer

The FilteredClusterer is meta-clusterer offers the possibility to apply filters directly before the cluterer is learned. The structure of the filter is based exclusively on the training data and test instances will be processed by the filter without changing their structure [24].

E. Sequential IB Clustering (sIB)

The sequential information bottleneck algorithm (sIB) assigns for each instance the cluster that has the minimum cost/distance to the instance. The trade-off beta is set to infinite so 1/beta is zero.

The algorithm starts from an initial partition C of the objects in X. The cluster cardinality $|C|$ and the joint probability $p(x, y)$ are required in advance. At each step of the algorithm, one object $x \in X$ is drawn out of its current cluster $c(x)$ into a new singleton cluster. Using a greedy merging criterion, x is assigned or merged into c^* so that $c^* = \text{argmin}_c\ dF(\{x\}, c)$. The merging cost, the information loss due to merging of the two clusters, represented as $dF(ci, cj)$, is defined as [14]:

$$d_F(c_i, c_j) = \left(p(c_i) + p(c_j)\right) \cdot D_{JS}\left[p(y|c_i), p(y|c_j)\right] \quad (2)$$

where D_{JS} is actually Jensen-Shannon (JS) [15] divergence and $p(ci)$ and $p(cj)$ are cluster prior probabilities. JS divergence is non-negative and equals zero if and only if both its arguments are the same and usually relates to the likelihood measure that two samples, independently drawn from two unknown distributions, are actually from the same distribution.

The sIB algorithm stops as the number of new assignments, among all objects X, to new clusters are less than a threshold, which means that so far the clustering results are "stable." Meanwhile, multiple random initialization is used to run sIB multiple times and select the results that has the highest cluster MI $I(C; Y)$, namely the least information loss $I(X; Y) - I(C; Y)$.

IV. Methodology

A. Detecting anomalies

One serious problem with any form of automatic detection of apparently incorrect data. Short of consulting a human expert, there is really no way of telling whether particular instance really is an error, or whether it just does not fit the type of model that is being applied. In statistical regression, visualizations help. It will usually be visually apparent, even to the nonexpert, if the wrong kind of curve is being fitted--a straight line is being fitted to data that lies on a parabola, for example.

One solution that has been tried is to use several different learning schemes—like a k-means, sIB, FilterdClusterer, RandomFlatClustering and Fasthestfirst Learner to filter the data.

B. Data Collection

The Ethereal is used to collected data, because it is very easy to collect several traffic data and easy to choose traffic with any attributes.

Ethereal [11] is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible. Ethereal isn't an intrusion detection system. It will not warn you when someone does strange things on your network that he/she isn't allowed to do. However, if strange things happen, Ethereal might help you figure out what is really going on.

Ethereal will not manipulate things on the network, it will only 'measure' things from it. Ethereal doesn't send packets on the network or do other active things (expert for name resolutions, but even that can be disabled).

Ethereal is an open source software project, and is released under the GNU General Public License (GPL). All source code is freely available under the GPL. It is a well-designed and easy to use GUI based program for sniffing an Ethernet interface for packets and making sense of them. Ethereal maps IP addresses, MAC addresses, and high and low level protocol fields to symbolic names for easier interpretation. It allows an interpretive look at any part of a packet, avoids showing the overwhelming but normally uninteresting portions of a packet. It has a very power full filtering capability that understands even such application-level protocols as DNS, SMTP, NFS, SMB (CIFS) and Quake. Ethereal will assemble IP packet into the higher-level TCP packets, for example, and show you the ASCII or hex data that is being transferred. It can be used for both good and evil [10]. Fig.1 is shown the example of data collection with Ethereal in many attributes.

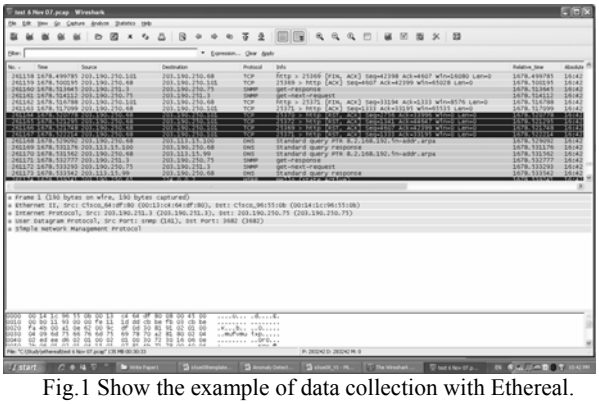


Fig.1 Show the example of data collection with Ethereal.

The data is collected for a day in TOT Network, for full 24 hours and for both link directions. There is a lot of traffic that shown in Table I.

TABLE I
SUMMARY OF TRAFFIC DATA SEPARATE BY PROTOCOL

Protocol	#Packets	#Bytes	% Packets of total	% Bytes of total
TCP	16,899	10,359,675	1.612	2.476
UDP	263,366	391,461,363	25.117	93.547
ICMP	20,480	2,267,014	1.953	0.542
DNS	52,710	12,760,619	5.027	3.049
IP	20,561	1,233,660	1.961	0.295
Others	674,559	384,724	64.331	0.092
Total	1,048,575	418,467,055	100	100

Table I is show the summary of traffic data that we collected on 6 November, 2007. There is a lot of total traffic so we should to show traffic separate by protocol type. The number of packets that use TCP protocol is 16,899 packets (1.61% of Total packet). The packets use UDP protocol is 263,366 packets (25.12% of Total packet). The packets use ICMP protocol is 20,480 packets (1.95% of Total packet). The packets use DNS protocol is 52,710 packets (5.03% of Total packet). The packets use IP protocol is 20,561 packets (1.96% of Total packet). Others protocol was observed in the trace, namely SNMP, ARP, HSRP, NTP, SSH, because of when we separated by protocol type they are too few number of packet of the Total packet. We decided to choose TCP and UDP protocol to study about the characteristics of traffic anomaly.

C. Preparing the data

The data is presented in a spreadsheet and database. RapidMiner is support ARFF format; so that we must covert data form a spreadsheet to ARFF format, because it is very easy to convert form a spreadsheet to ARFF format. The bulk of an ARFF file consists of a list of the instances, and attribute values for each instance are separate by commas. Most spreadsheet and database programs allow exporting data into a file in comma-separated value (CSV) format as a list of records with commas between items. If data is in CSV format it is automatically converted into ARFF format.

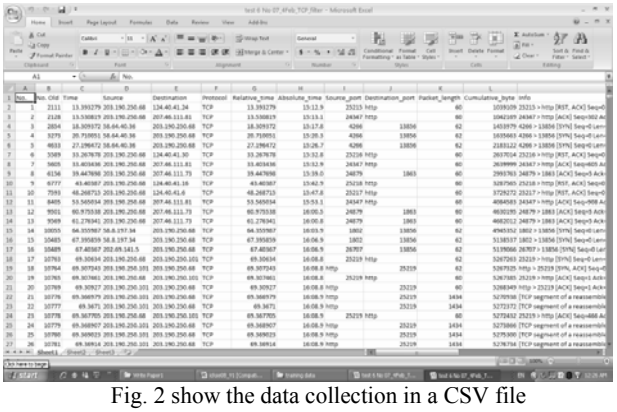


Fig. 2 show the data collection in a CSV file

D. Data Mining Software

In this paper, we explored the use of a machine learning approach called clustering for classifying traffic using only transport layer statistics. We chose TCP and UDP protocols to study about the characteristics of traffic anomaly, because they are a lot of packets when compare with total traffics. The RapidMiner application is used, because it has more than 400 data mining operators can be used and almost arbitrarily combined. It supports vary type of file, a large dataset and more functions are very easy to use.

E. RapidMiner

RapidMiner [10] is the world-wide leading open-source data mining solution due to the combination of its leading-edge technologies and its functional range. Applications of RapidMiner cover a wide range of real-world data mining tasks. The RapidMiner is used and explored our data. Simplify the construction of experiments and the evaluation of different approaches. Try to find the best combination of preprocessing and learning steps or let RapidMiner do that automatically for us.

This application has more than 400 data mining operators can be used and almost arbitrarily combined. The setup is described by XML files which can easily be created with a graphical user interface (GUI). This XML based scripting language turns RapidMiner into an integrated development environment (IDE) for machine learning and data mining. RapidMiner follows the concept of rapid prototyping leading very quickly to the desired results. Furthermore, RapidMiner can be used as a Java data mining library.

The development of most of the RapidMiner concepts started in 2001 at the Artificial Intelligence Unit of the University of Dortmund. Several members of the unit started to implement and realize these concepts which led to a first version of RapidMiner in 2002. Since 2004, the open-source version of RapidMiner (GPL) is hosted by SourceForge. Since then, a large number of suggestions and extensions by external developers were also embedded into RapidMiner. Today, both the open-source version and a close-source version of RapidMiner are maintained by Rapid-I.

Although RapidMiner is totally free and open-source, it offers a huge amount of methods and possibilities not covered by other data mining suites, both open-source and proprietary ones.

The modular operator concept of RapidMiner (formerly YALE) allows the design of complex nested operator chains for a huge number of learning problems in a very fast and efficient way (rapid prototyping). The data handling is transparent to the operators. They do not have to cope with the actual data format or different data views - the RapidMiner core takes care of all necessary transformations. Read here about the most important features of RapidMiner.

Knowledge Discovery in Databases (KDD) is a complex and demanding task. While a large number of methods have been established for numerous problems, many challenges remain to be solved. Rapid prototyping is an approach which allows crucial design decisions as early as possible. A rapid prototyping system should support maximal re-use and innovative combinations of existing methods, as well as simple and quick integration of new ones.

RapidMiner is flexible operators for data in- and output in different file formats including such as Arff, C4.5, csv, Excel files, SPSS files and data sets from databases (Oracle, mySQL, PostgreSQL, Microsoft SQL Server, Sybase etc.) and more. Otherwise RapidMiner has Machine learning algorithms more than 100 learning schemes for regression, classification, and clustering tasks, including. It has Data preprocessing operators before the learning process, Performance evaluation is several validation and evaluation schemes to estimate the performance of learning or preprocessing on dataset. In visualization part there are logging and presenting results include the visualization 1D, 2D and 3D plots. There is Data Mining for Developers to Extending RapidMiner and Integration as a Data Mining Engine aims at software developers and analysts with background knowledge in development.

V. Experimental Results

A. K-means

The K-means algorithm is used to cluster the data sets from 16,899 records which may be normal and anomalous traffic with unlabeled. This operator represents a simple implementation. We assume that clusters are normal and anomalous traffic from the two difference clusters in the space. That means we defined the number of cluster k = 2. We collected the data with all attributes that we could collect (9 attribute: Time, Source, Destination, Relative_time, Absolute_time, Source_port, Destination_port, Packet_length Cumulative_byte). Because we thought all attribute that we

collected must important attributes for considering in each attribute. It may be useful for classify the cluster. First way we represented the plot view with three attribute (Time, Source_port, Destination_port) in scatter 3D color (Figure 7). The description of the first attribute (Time) is the time from beginning of the capture to the time when the packet was captured (in seconds). The description of the second attribute (Source_port) is identifies the sending port. The description of the third attribute (Destination_port) is identifies the receiving port. In cluster#0 there are 5,755 items (34%) and cluster#1 there are 11,144 items (66%).

In Fig. 3 and Fig. 4 show the example data distribution in scatter plot between each attribute and other attributes. This picture is useful for us to choose the attribute and cluster the data set with K-means algorithm.

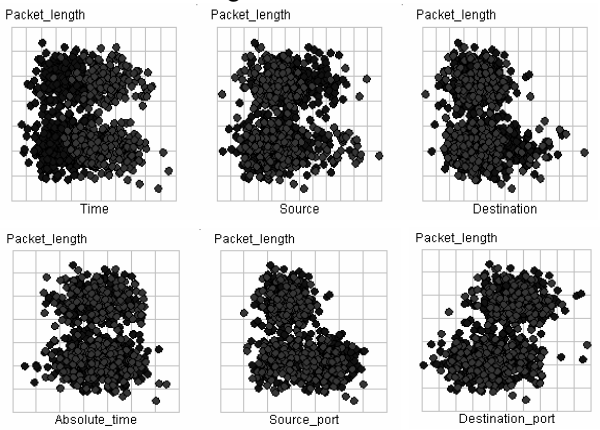


Fig. 3 Plot view in scatter matrix between Packet_length attribute and each attribute

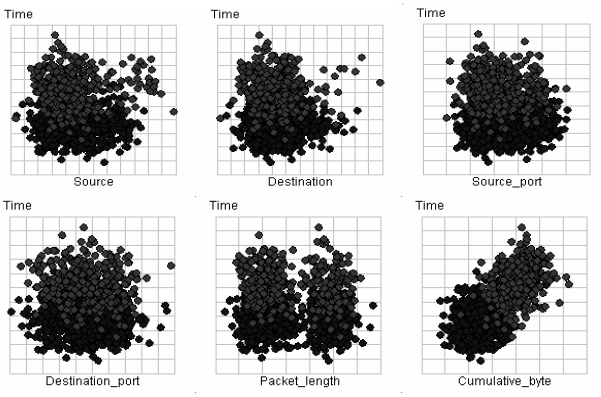
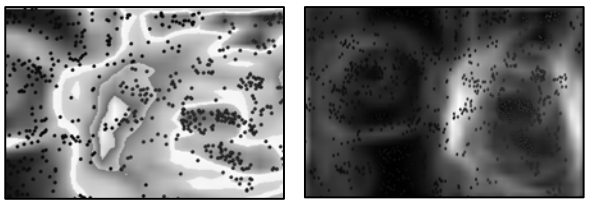


Fig. 4 Plot view in scatter matrix between Time attribute and each attribute

In Fig. 5 show data distribution with a Self-Organizing Map (SOM) plot using a Kohonen net for dimensionality reduction with gray scale (Fig. 5(a)) and Landscape (Fig. 5(b)). We plot between two attributes, time attribute and packet_length attribute. This visualization show clearly data cluster. Fig. 6 show a 2D plotter using two dimensions as axis, one dimension as density color, and one dimension for point colors. Plot with the same attribute in Fig. 5.



(a) (b)

Fig. 5 show A Self-Organizing Map (SOM) plot
(a) Gray scale, (b) Landscape

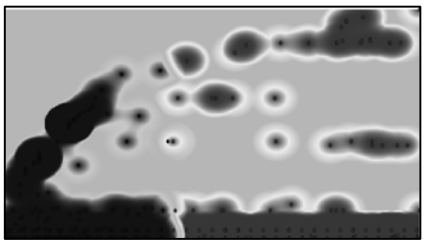


Fig. 6 show A 2D plotter using two dimensions

We used 3D color scatter plot produces 3D plots. The second 3D plot mode of RapidMiner is the 2D color plot mode. The first two dimensions build an 2D layer and the third dimension is mapped on different colors or alternatively different sizes. A 3D scatter plot colorizing a 4th dimension. The three attributes that we plot in 3D, x-axis is time attribute, y-axis is source_port attribute, z-axis is destination_port attribute and the difference color is shown the difference cluster data. Otherwise we plotted with other attributes.

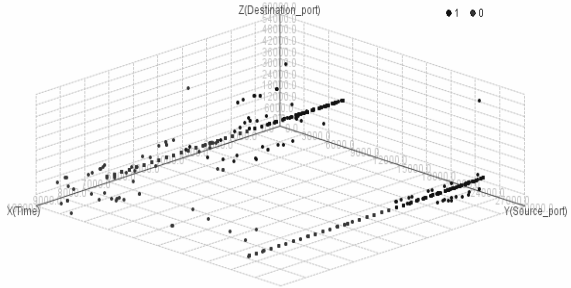


Fig. 7 plot view in scatter 3D color between three attribute (Source_port, Destination_port and time)

B. FarthestFirst

In cluster#0 there are 2746 items (16.25%) and cluster#1 there are 14,154 items (83.75%).

In Fig. 8 show the example data distribution in scatter plot between each attribute and other attributes.

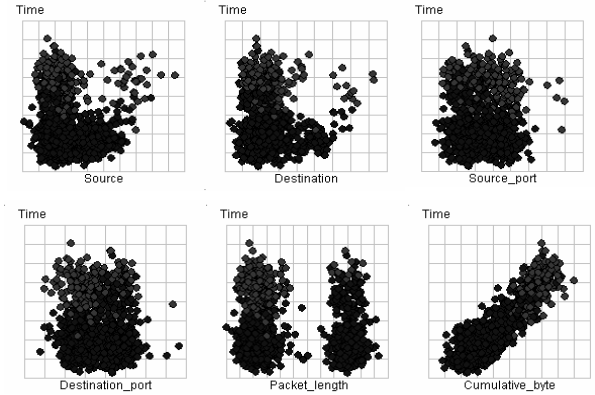


Fig. 8 plot view in scatter matrix between Packet_length attribute and each attribute

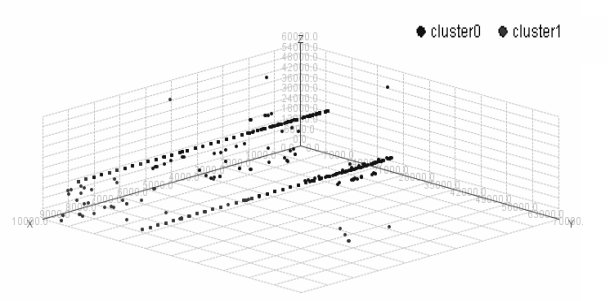


Fig. 9 plot view in scatter 3D color between three attribute (Source_port, Destination_port and time)

C. FilteredClusterer

In cluster#0 there are 6704 items (39.67%) and cluster#1 there are 10,195 items (60.33%).

In Fig. 10 show the example data distribution in scatter plot between each attribute and other attributes.

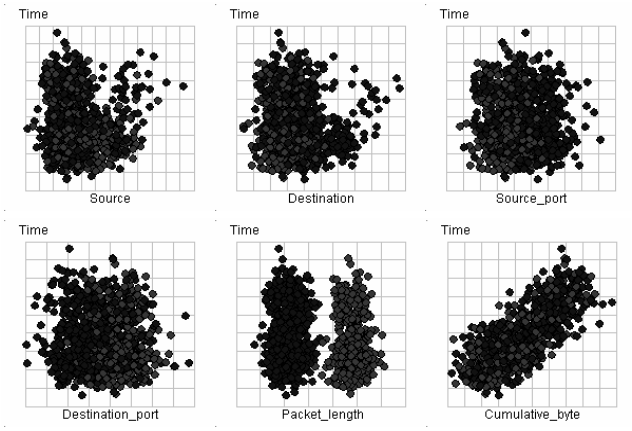


Fig. 10 plot view in scatter matrix between Packet_length attribute and each attribute

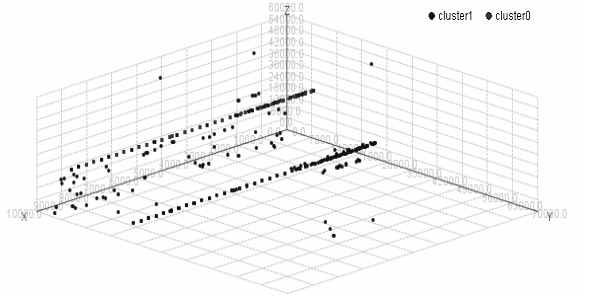


Fig. 11 plot view in scatter 3D color between three attribute (Source_port, Destination_port and time)

D. Sequential IB Clutering (sIB)

In cluster#0 there are 986 items (5.83%) and cluster#1 there are 15,913 items (94.17%).

In Fig. 12 show the example data distribution in scatter plot between each attribute and other attributes.

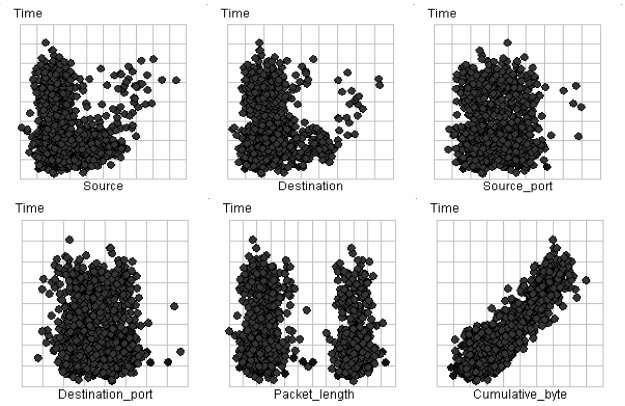


Fig. 12 plot view in scatter matrix between Packet_length attribute and each attribute

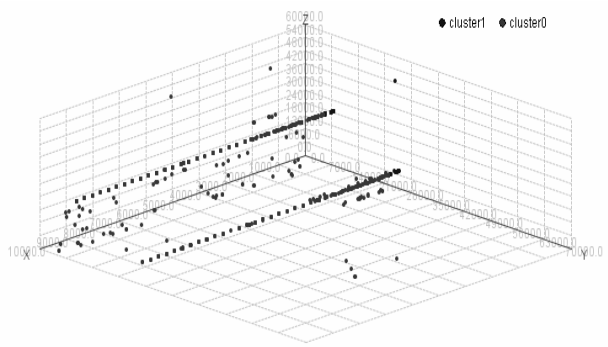


Fig. 13 plot view in scatter 3D color between three attribute (Source_port, Destination_port and time)

E. RandomFlatClustering

In cluster#0 there are 8,509 items (50.35%) and cluster#1 there are 8,390 items (49.65%).

In Fig. 14 show the example data distribution in scatter plot between each attribute and other attributes.

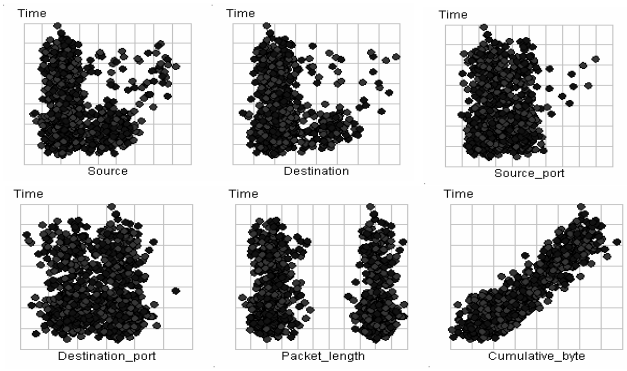


Fig. 14 plot view in scatter matrix between Packet_length attribute and each attribute

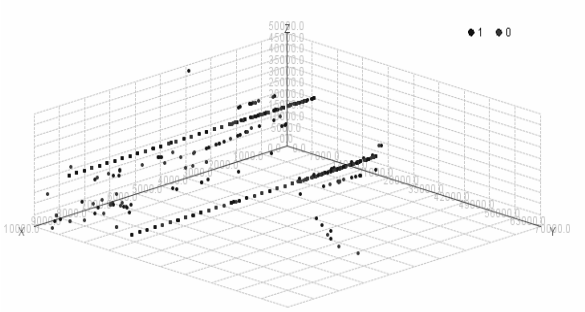


Fig. 15 plot view in scatter 3D color between three attribute (Source_port, Destination_port and time

F. Evaluation

The performance evaluator operator can be used for all types of leaning tasks. It will automatically determine the learning task type and will calculate the most common criteria for this type. The operator expects a test dataset as input, whose elements have both true and predicted labels, and delivers as output a list of most common performance values for the provided learning task type. When an input performance vector was already given, this is used for keeping the performance values. We used ItemDistributionEvaluator function to evaluate cluster models on how well the items are distributed over the clusters. An evaluator for centroid based clustering methods. Result of items distribution use k-means algorithm that measure with SumOfSquares is 0.551. For farthest-first algorithm that measure with SumOfSquares is 0.728. SumOfSquares for FilteredClusterer algorithm is 0.521. SumOfSquares for Sequential IB Clustering algorithm is 0.890 and SumOfSquares for RandomFlatClustering algorithm is 0.500.

We evaluated the correct classification of the data cluster with Precision, Recall [19] and F-measure. After the class label assigned to the item by a classifier we used NaiveBaye Classification to test the data with test mode 10-fold cross-validation.

Cross validation: In order to perform to measure classification error, it is necessary to have test data samples independent of the learning dataset that was used to build a classifier. However, obtaining independent test data is difficult or expensive, and it is undesirable to hold back data from the learning dataset to use for a separate test because that weakens the learning dataset. V-fold cross validation technique performs independent tests without requiring separate test datasets and without reducing the data used to build the tree. The learning dataset is partitioned into some number of groups called “folds” [20]. The number of groups that the rows are partitioned into is the ‘V’ in Vfold cross classification. 10 is the recommended and default number for “V”. It is also possible to apply the v-fold crossvalidation method to a range of numbers of clusters in kmeans or EM clustering, and observe the resulting average distance of the observations from their cluster centers.

The number of correctly classified data in cluster is referred to as the True Positives (TP). Any data that are not correctly classified are considered False Positive (FP). Any data that has not been assigned to cluster is labeled as noise. In the context of classification tasks, the terms true positives, true negatives, false positives and false negatives are used to compare the given classification of an item (the class label assigned to the item by a classifier) with the desired correct classification (the class the item actually belongs to). This is illustrated by the Table II below:

TABLE II
SUMMARY OF THE TERM CONTEXT

Obtained result/ classification	Correct result/ classification	
	tp (true positive)	fp (false positive)
	fn (false negative)	tn (true negative)

The accuracy is calculated as follows:

$$Precision = \frac{\#PairsCorrectlyPredictedInSameCluster}{\#TotalPairsPredictedInSameCluster}$$

That is,

$$Precision = \frac{tp}{tp + fp} \quad (3)$$

$$Recall = \frac{\#PairsCorrectlyPredictedInSameCluster}{\#TotalPairsActuallyInSameCluster}$$

That is,

$$Recall = \frac{tp}{tp + fn} \quad (4)$$

Pairwise F-measure is defined as the harmonic mean of pairwise precision and recall, the traditional information retrieval measures adapted for evaluating clustering by considering pairs of point-for every pair of points that do not have explicit constraints between them, the decision to cluster this pair into the same or different clusters is considered to be correct if it matchs with the underlying class labeling available for the points [21][22].

In statistics, the F-measure [17] or F-score is a measure of a test's accuracy. It considers both the precision *p* and the recall *r* of the test to compute the score [18] : *p* is the number of correct results divided by the number of all returned results and *r* is the number of correct results divided by the number of results that should have been returned. The F-measure can be interpreted as a weighted average of the precision and recall, where an F-measure reaches its best value at 1 and worst score at 0.

The traditional F-measure or balanced F-score (F1 score) is the harmonic mean of precision and recall:

$$F = \frac{2 \cdot (precision \cdot recall)}{(precision + recall)} \quad (5)$$

The general formula for non-negative real β is:

$$F_\beta = \frac{(1 + \beta^2) \cdot (precision \cdot recall)}{(\beta^2 \cdot precision + recall)} \quad (6)$$

The formula in terms of Type I and type II errors:

$$F_\beta = \frac{(1+\beta^2)\cdot(true\ positive)}{((1+\beta^2)\cdot true\ positive + \beta^2 \cdot false\ positive + false\ negative)} \quad (7)$$

Two other commonly used F measures are the F_2 measure, which weights recall twice as much as precision, and the $F_{0.5}$ measure, which weights precision twice as much as recall.

The F-measure was derived by van Rijsbergen (1979) so that F_β "measures the effectiveness of retrieval with respect to a user who attaches β times as much importance to recall as precision". It is based on van Rijsbergen's effectiveness measure:

$$E = 1 - (1 / (\alpha / P + (1 - \alpha) / R)) \quad (8)$$

Their relationship is

$$F_\beta = 1 - E \text{ where } \alpha = 1 / (\beta 2 + 1) \quad (9)$$

The summary of the cluster result classify by algorithms show below:

TABLE III
SUMMARY OF THE CLUSTER RESULT

Algorithms	# items of Cluster0	# items of Cluster1
K-means	5755	11144
W-sIB	986	15913
FilteredCluterer	6704	10195
FarthestFirst	2745	14154
RandomFlatClustering	8509	8390
K-Mediods	5749	11150

The result of the accuracy in Cluster0 classify by algorithms show below:

TABLE IV
RESULT OF THE ACCURACY IN CLUSTER0

Cluster0			
Algorithm	Precision	Recall	F-Measure
K-means	1	0.975	0.987
W-sIB	1	0.98	0.99
FilteredCluterer	1	0.998	0.999
FarthestFirst	0.998	0.897	0.945
RandomFlatClustering	0.507	0.574	0.539
K-Mediods			

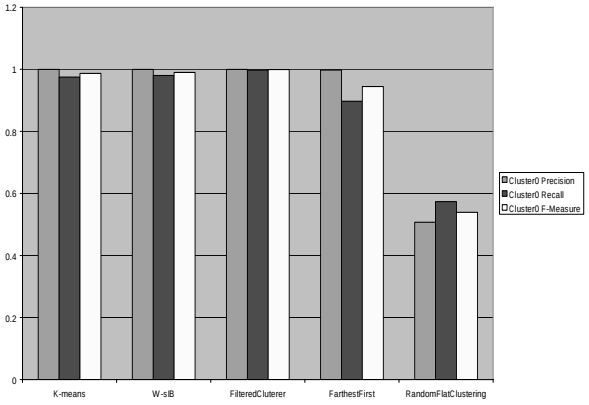


Fig. 16 show chart with precision, recall and F-measure of cluster0

The result of the accuracy in Cluster1 classify by algorithms show below:

TABLE V
RESULT OF THE ACCURACY IN CLUSTER1

Cluster1			
Algorithm	Precision	Recall	F-Measure
K-means	0.953	1	0.976
W-sIB	0.752	1	0.858
FilteredCluterer	0.999	1	0.999
FarthestFirst	0.652	0.991	0.787
RandomFlatClustering	0.501	0.433	0.464

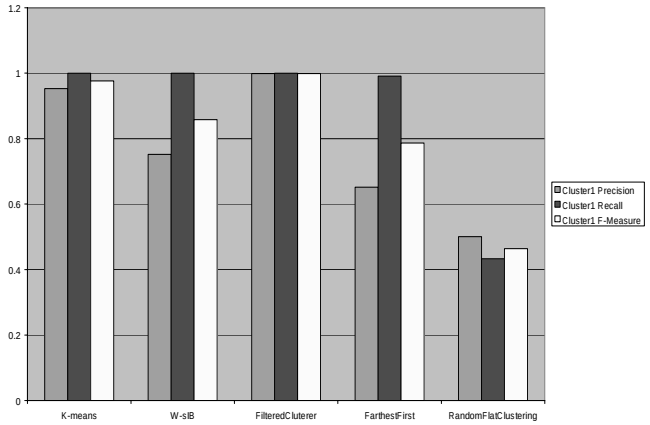


Fig. 17 show chart with precision, recall and F-measure of cluster1

VI. Discussion

In cluster0 the algorithms that give highest precision are K-means, W-sIB and FilteredCluterer. The precision value of them is 1, it means these algorithms give the best efficiency. The algorithms that give lowest precision are RandomFlatClustering is 0.507. The Filterclusterer give the highest recall value and F-measure are 0.998 and 0.999. The RandomFlatClustering algorithm gives lowest precision, recall and F-measure value. Fig. 16 compare precision, recall and F-measure value of cluster0, this chart is told that FilteredCluterer algorithm give the best efficiency. It is the same result in cluster1, FilteredCluterer algorithm give the best efficiency. The precision, recall and F-measure value of cluster1 are 0.999, 1 and 0.999. For RandomFlatClustering algorithm we thought it did not suit for this data type (Network traffic data) it may be suit for other data type such as bioinformatics.

VII. Conclusion

This paper we represents an unsupervised clustering algorithm namely sIB, RandomFlatClustering, FarthestFirst, and FilteredClusterer that have previously not been used for network traffic classification. We evaluate this algorithm and compare it to the previously used (K-Means). In our experimental result the FilteredClusterer algorithm gives the best efficiency. The K-means algorithm gives the best products of clusters and work quickly. In the future we will evaluate the best efficiency of the algorithm from accuracy with known data and real time traffic data. We will use other clustering algorithms that give the best efficiency more than these algorithms and continue to test other attributes.

VIII. Acknowledgment

This research was supported by Database and Software Technology Research Laboratory.

IX. References

[1] K. Ramah, H. Ayari, and F. Kamoun, "Traffic Anomaly Detection and Characterization in the Tunisian National University Network", *Networking*, 2006, pp. 136-147.

[2] A. Lakhina, M. Crovella, and C. Diot, "Mining Anomalies Using Traffic Feature Distributions", *Technical Report BUCS-TR-2005-002*, Boston University, 2005.

[3] M. Shyu, S. Chen, K. Sarinnapakorn, and L. Chang, "A Novel Anomaly Detection Scheme Based on Principal Component Classifier', In *Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop*, in conjunction with the Third IEEE International Conference on Data Mining (ICDM'03), pp.172-179, Melbourne, Florida, USA, 2003.

[4] G. Münz, S. Li, and G. Carle, "Traffic Anomaly Detection Using K-Means Clustering", In GI/ITG Workshop MMBnet, 2007.

[5] L.S. Silva, T.D. Mancilha, J.D.S. Silva, A.C.F. Santos, e A. Montes, "A Framework for Analysis of Anomalies in the Network Traffic", In INPE'06, São José dos Campos, December 2006.

[6] P. Tan, M. Steinbach, V. Kuman, "*Introduction to Data Mining*", Addison Wesley, 2006.

[7] J. Erman, M. Arlitt, A. Mahanti, "Traffic Classification Using Clustering Algorithms", In *SIGCOMM'06 MineNet Workshop*, Pisa, Italy, September 2006.

[8] A. McGregor, M. Hall, P. Lorier, and J. Brunskill, "Flow Clustering Using Machine Learning Techniques", In *PAM 2004*, Antibes Juan-les-Pins, France, April 19-20, 2004.

[9] S. Zander, T. Nguyen, and G. Armitage, "Automatic Traffic Classification and Application Identification using Machine Learning", In *LCN'05*, Sydney, Australia, Nov 15-17, 2005.

[10] RapidMiner Homepage, http://rapid-i.com/content/ blogcategory/38/69/

[11] Ethereal Homepage, http://www.rootsecure.net/content/ downloads/pdf/ethereal_guide.pdf

[12] M. Ester, H. Kriegel, J. Sander, and X. Xu, "A density-based Algorithm for discovering Clusters in Large Spatial Databases with Noise", In *2nd Int. Conf. on Knowledge Discovery and Data Mining (KDD 96)*, Portland, USA, 1996.

[13] J. MacQueen, "Some methods for classification and analysis of multivariate observations", In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, University of California Press*, 1976, pp. 281-297.

[14] Winston H. Hsu, and Shih-Fu Chang, "Visual Cue Cluster Construction via Information Bottleneck Principle and Kernel Density Estimation", In *CIVR 2005*, pp. 82-91.

[15] Zheng-Yu Niu, Dong-Hong Ji, Chew Lim Tan, "Using cluster validation criterion to identify optimal feature subset and cluster number for document clustering", In *Information Processing and Management' 06*, 2006.

[16] icml2006 …Precision , Recall

[17] http://en.wikipedia.org/wiki/F-score

[18] http://en.wikipedia.org/wiki/Precision_and_recall

[19] J. Davis and M. Goadrich, "The Relationship Between Precision-Recall and ROC Curves", In *ICML'06*, 2006

[20] M. Pirooznia, J. Y Yang, M. Qu Yang and Y. Deng, "A comparative study of different machine learning methods on microarray gene expression data", In *BIOCOMP'07*, June 2007.

[21] Michael W. Berry, Umeshwar Dayal, Chandrika Kamath and David Skillicorn, "Proceedings of the Fourth SIAM International Conference on Data Mining", p 338, 2004

[22] B. Sugato ,"Semi-supervised Clustering: Learning with Limited User Feedback", November 2003

[23] A. William, "Clustering Algorithms for Categorical Data", September 2006.

[24] RapidMiner Homepage, http://downloads.sourceforge.net/yale/ rapidminer -4.2 -guimanual.pdf