Journal of King Saud University – Computer and Information Sciences (2015) 27, 77–92

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Identification of VoIP encrypted traffic using a machine learning approach

Riyad Alshammari [a,*], A. Nur Zincir-Heywood [b]

[a] College of Public Health and Health Informatics, King Saud Bin Abdulaziz University for Health Sciences, P.O. Box 22490, Riyadh 11426, Saudi Arabia
[b] Faculty of Computer Science, Dalhousie University, Halifax, Nova Scotia B3H 1W5, Canada

Received 22 April 2013; revised 16 December 2013; accepted 13 March 2014
Available online 24 January 2015

KEYWORDS

Machine learning;
Encrypted traffic;
Robustness;
Network signatures

Abstract We investigate the performance of three different machine learning algorithms, namely C5.0, AdaBoost and Genetic programming (GP), to generate robust classifiers for identifying VoIP encrypted traffic. To this end, a novel approach (Alshammari and Zincir-Heywood, 2011) based on machine learning is employed to generate robust signatures for classifying VoIP encrypted traffic. We apply statistical calculation on network flows to extract a feature set without including payload information, and information based on the source and destination of ports number and IP addresses. Our results show that finding and employing the most suitable sampling and machine learning technique can improve the performance of classifying VoIP significantly.

1. Introduction

Peer-to-Peer (P2P) Voice over Internet Protocol (VoIP) applications grow to be the most important communication services in the last few years for companies and individuals since the voice and video quality are very good and the calls are free for direct connection between two VoIP end users. VoIP products such as Gtalk (Gtalk, 2009), Primus (Primus, 2009) and Skype (Skype) hide its communication by implementing encryption and uses different techniques to bypass firewall and NAT restrictions. Therefore, an efficient classification algorithm to distinguish encrypted VoIP traffic is an essential requirement for managing network to ensure the proper utilization of bandwidth to critical user applications.

The conventional techniques to classify network traffic by using 'Deep Packet Inspection' (DPI) and port numbers based classification are becoming unsuccessful for the identification of encrypted VoIP applications. Therefore, many researches employ learning techniques using statistical features calculated from network flow traffic derived from the network communication on the transport layer excluding payload information (Alshammari and Zincir-Heywood, 2011; Erman et al., 2006; Karagiannis et al., 2005; Bernaille et al., 2006). This research paper employs three supervised learning algorithms: C5.0, AdaBoost and Genetic Programming (GP) since in our

* Corresponding author.
E-mail addresses: alshammarir@ksau-hs.edu.sa (R. Alshammari), zincir@cs.dal.ca (A. Nur Zincir-Heywood).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2014.03.013

(Alshammari and Zincir-Heywood, 2007, 2008, 2009a, b; Alshammari, 2008) and other researchers' previous work (Early et al., 2003; Haffner et al., 2005; Williams et al., 2006), they have been shown to provide good solutions. All three of these learning algorithms can produce solutions automatically in the form of models/rules which can be easily understood by human experts. We refer to these models/rules as signatures to identify the VoIP application. This is a very important property in order to employ the generated rules as signatures to classify traffic in practice. Furthermore, these learning models (C5.0, GP and AdaBoost) provide human readable solutions, hence, the solutions they generate are not a black box to the system administrators or network engineers. Additionally, other supervised learning algorithms (black box methods) such as Support Vector Machines (SVM) and Bayesian methods have significant memory overheads. Particularly, Bayesian methods require a lot of expertise to extract their potential. Conversely, C5.0 addresses the memory overheads of C4.5 (Quinlan, 2011) making for a very robust implementation. Likewise, AdaBoost and GP manage the memory very well. Additionally, as we have shown in Alshammari and Zincir-Heywood (2011), these techniques have the ability to select the most suitable features/attributes from a list of features. However, these techniques require two major steps. Firstly, features need to be defined to describe the traffic data to the algorithms. In this case, features can be calculated over flows representing multiple packets. Secondly, these techniques require to be trained to correlate the features to the desired traffic classes, i.e. labels, (supervised learning) and to create models/rules (we call these as signatures) as their solutions.

The number of network packets passing through high-speed links is massive and is affected by the applications used, the number of users and the capacity of the links. As a result, sampling network traffic for the aforementioned training classifiers becomes a vital procedure for dealing with huge volumes of traffic where resources are limited (e.g. hard disk, memory). The most challenging part of network traffic sampling is to be able to capture the behavior of an application by observing an adequate number of packets/flows. Therefore, to explore the capability of learning algorithms in generating robust signatures, we make the training and testing data sets totally different where the training data set is much smaller in size than the testing data set. Obviously, the size of the network traffic traces is huge from the learning algorithms' perspective. Subset sampling methods would reduce the amount of memory utilized and the required time of the Central Process Unit (CPU) to conduct training.

To this end, related work is reviewed in Section 2. Section 3 describes the feature sets, learning algorithms and the evaluation method employed. The subset sampling methods are presented in Section 4. Section 5 details the experimental results on selecting the suitable sampling method for traffic classification tasks. Finally, the conclusion and the future work are discussed in Section 6.

2. Traffic classification: specifically P2P and VoIP

In the literature, Bonfiglio et al. (2007) present one of the earlier studies in classifying Skype traffic using supervised learning techniques. They presented two methods for classifying Skype Peer-to-Peer (P2P) VoIP traffic. They used Pearson's Chi-Square (χ^2) test for the first method employing information extracted from the payload. For the second method, they used Naïve Bayesian Classifier using information based on packet length and packet arrival rate. They achieved best performance by combing the two methods (a 1% false positive rate and a 2–29% false negative rate). However, their classification methods are based on the inspection of payload information as well as using a priori information. Freire et al. (2008) used feature sets from HyperText Transfer Protocol (HTTP) request and response sizes, the number of requests and time to derive metrics to classify Skype and Gtalk flows from Web traffic based on (χ^2) value and the Kolmogorov–Smirnov distance. They achieved high performance for classifying both applications. Recently, Este et al. (2009) applied Support Vector Machines (SVM) for classifying only Transmission Control Protocol (TCP) bi-directional flows relying mainly on the packet size as the main feature. The SVM models are able to classify multiple applications such as HTTP, HTTPS (secure HTTP), BitTorrent, e-Donkey, Kazaa, Gnutella and MSN. They tested their methods on three traces, which were captured from different locations. They were able to achieve high performance on the e-Donkey flows but had poor results on other P2P applications such as Kazza and Gnutella. Huang et al. (2013) apply machine learning algorithms based on layer 7 (application layer) information by extracting attributes from the first 20 packets to a maximum of 200 packets. They also included TCP/UDP port numbers to the attribute set to be able to identify 59 network applications with high accuracy.

On the other hand, unsupervised learning methods have been used in network traffic classification as well. Bernaille et al. (2006) clustered network traffic by using an unsupervised learning method in order to label it according to the application protocols. They clustered the first five packets of TCP flows based on the packet size in each connection. They used the Euclidean distance and *K*-Means algorithm to build an online classifier consisting of fifty clusters to classify only TCP network flows. However, the classifier has problems handling similar flow sizes employed by different applications, basically labeling the flows the same way.

Erman et al. (2007) apply a semi-supervised technique for classifying such internet flow traffic as the Web, File Transfer Protocol (FTP), and P2P file sharing. The semi-supervised learning method consists of two methods. They used the Euclidean distance and the *K*-means algorithm for the first method to cluster traffic. The clusters contain pre-labeled flows and unlabeled flows. The second method involves using the maximum likelihood estimate for the pre-label flows inside the cluster to label the cluster into known network traffic classes. However, this provides misleading results if it applies on unbalanced data sets in which the data set consists of, say, two classes only (in a total of one hundred instances), 10 instances of FTP and 90 instances of P2P. Thus, the classifier can achieve an accuracy of 90% by labeling all the instances as the major class but the false positive rate for P2P would be 100%. Bacquet et al. (2010) employed five unsupervised learning algorithms that are DBSCAN, EM, MOGA, Basic K-means and Semi-supervised K-means for detecting SSH encrypted network traffic. They achieved best performance with MOGA based classifier.

Recently, Iliofotou et al. (2011) employed Traffic Dispersion Graphs (TDGs) for classifying P2P traffic (e.g. Gnutella,

e-Donkey and BitTorrent). Their approach worked by grouping the first sixteen bytes of the payload using the *K*-Means algorithm. These bytes act as categorizing features ranging from 0 to 255. Then, the TDGs are used to classify the clusters.

In summary, these works show that it is promising to classify network applications using Machine Learning (ML) based approaches. However, more research is required to determine between VoIP P2P and encrypted applications accurately since P2P and encrypted applications, such as Skype, allocate port numbers dynamically, and use the same port number for multiple applications such as ports 80 (HTTP) and 443 (SSL). Furthermore, Moore and Papagiannaki demonstrated that the classification based on IANA port list is accurate 70% of the time (Moore and Papagiannaki, 2005). Additionally, Madhukar and Williamson verified that IANA port list is misclassifying 30–70% of their flow network data (Madhukar and Williamson, 2006). Therefore, to the best of our knowledge this paper is the first work that investigates the issue of producing signatures that can classify P2P network applications robustly by using machine learning algorithms without using features/attributes based on IP addresses, TCP/UDP port numbers or payload information. However, in this work, we consider the effect of sub-sampling techniques since it can improve the generalization (robustness) of signatures (rules) learned automatically during the training phase of such techniques.

3. Methodology

In this research, the focus is on the classification of VoIP encrypted traffic, specifically Skype encrypted traffic using supervised learning algorithms. As discussed earlier, we are employing three supervised learning algorithms, namely C5.0, AdaBoost and Genetic Programming (GP), to generate signatures automatically to robustly classify VoIP encrypted traffic. The learning algorithms require the representation of the data via feature (attribute) set, labeling of the data, training of the learning algorithms and testing the solutions. The details of these steps and the data sets employed are presented in this section.

3.1. Flow-based feature set

In this work, we represent the network traffic as a bidirectional flow connection between two hosts where the two hosts have the same 5-tuple (source and destination port numbers, source and destination IP addresses and the protocol). In these flows, the client-to-server connections represent the forward direction while the server-to-client connections represent the backward direction. Moreover, the maximum duration of a flow time-out is 600 seconds as used in IETF. TCP flows are ended either by flow time out or by connection teardown while UDP flows are ended by flow time out. We used the NetMate tool set (NetMate; Arndt, 2011) to generate the flows and calculated the statistical feature values, Table 1. Furthermore, we only include flows that have at least one packet in both directions and have payload of at least one byte.

Table 1 Flow feature employed.

	Abbreviation	Feature name
1	min_fiat	Minimum of forward inter-arrival time
2	mean_fiat	Mean of forward inter-arrival time
3	max_fiat	Maximum of forward inter-arrival time
4	std_fiat	Standard deviation of forward inter-arrival times
5	min_biat	Minimum of backward inter-arrival time
6	mean_biat	Mean backward inter-arrival time
7	max_biat	Maximum of backward inter-arrival time
8	std_biat	Standard deviation of backward inter-arrival times
9	min_fpkt	Minimum of forward packet length
10	mean_fpkt	Mean of forward packet length
11	max_fpkt	Maximum of forward packet length
12	std_fpkt	Standard deviation of forward packet length
13	min_bpkt	Minimum of backward packet length
14	mean_bpkt	Mean of backward packet length
15	max_bpkt	Maximum of backward packet length
16	std_bpkt	Standard deviation of backward packet length
17	proto	Protocol
18	Duration	Total duration
19	f_packets	Number of Packets in forward direction
20	f_bytes	Number of Bytes in forward direction
21	b_packts	Number of Packets in backward direction
22	b_bytes	Number of Bytes in backward direction

3.2. Labeling, training and testing data sets

In this paper, the label of a flow is a class which indicates the type of the IP traffic. Labels reflect the ground truth of a given data set. Thus, if the traffic type is known, a label is provided for each flow (data record) in the data sets.

Machine learning algorithms need training data to build its model. Once they are trained, they give an output model. The output model can be validated and tested on unseen data sets. Sampling a representative subset of data for training the learning algorithms is a difficult task. In this paper, Section 4 describes how the training data sets are sampled. Moreover, test data sets are important for determining the best learning algorithm based on the evaluation criteria on unseen data/network traffic and hence is an important step in identifying the robustness of the classifiers. Section 3.5 describes the test data sets employed in this paper in detail.

3.3. Machine learning algorithms deployed

Three learning algorithms are employed in this paper. These are C5.0, AdaBoost and GP. The C5.0 (Quinlan, 2010) is the commercial decision tree algorithm developed from the famous C4.5 decision tree algorithm. C5.0 includes all the properties of C4.5 and has additional new technologies such as boosting. The major advantage of C5.0 over C4.5 is efficiency, otherwise both algorithms remain the same (Quinlan, 2011). C5.0 builds its solution by recursively splinting the input space into regions where the splits are considered to be pure for all branches. The C5.0 uses entropy to calculate the proportion of exemplars

corresponding to the number of classes in the training data. In the case of impure split, the exemplars are separated to decrease the impurity. The next stage is calculating the information gain for each feature to reduce the entropy in the training data. Additional information on this algorithm can be found in Alpaydin (2004).

On the other hand, AdaBoost algorithm is a meta-learning algorithm that builds its solution incrementally by boosting weak learning classes to strong leaning classes from the training data set. The classes are built by the intersection of many weak simple classes (decision stumps) by using a voting scheme. AdaBoost algorithm generates many hypotheses where each decision stump would return $+1$ or -1. Additional information on this algorithm can be found in Alpaydin (2004).

Finally, the Symbiotic Bid-Based (SBB) genetic programming technique, which is part of the team based GP family, is also employed in this work. SBB depends heavily on coevolution (Lichodzijewski and Heywood, 2008) to build its model by employing three different populations, namely teams, points, and learners. A symbiotic relation exists between the learner population and the team population where a bidding strategy exists between the learner population and the team population. A linear representation is employed as a bidding strategy of the learner population of individuals. Furthermore, the individuals in the team populations bid against each other to compete on the training data. The point population is responsible for competitive co-evolutionary relationship between the team population and the point population that can scale the evolution on big data sets (de Jong, 2007). Additional information on the SBB based GP algorithm can be found in Lichodzijewski and Heywood (2008).

3.4. Evaluation criteria of learning algorithms

Two evaluation criteria are used in traffic classification to measure the performance of the learning algorithms. These are Detection Rate (DR) and the False Positive Rate (FPR). The DR, Eq. (1), reflects the total number of flows that are correctly classified from the in-class (the ones which the algorithm aims to classify):

$$\mathrm{DR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{1}$$

whereas the FPR, Eq. (2), reflects the total number of out-class flows that are classified incorrectly as in-class.

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}} \tag{2}$$

The desirable outcomes are to obtain a high percentage value for the DR and a low percentage value for the FPR. Moreover, the False Negative (FN) rate represents the total number of in-class flows that are classified as out-class flows.

In this paper, 50 runs are used to train each of the learning algorithms on each training data set to generate different models. To this end, we used 50 different confidence factors for C5.0, 50 different weight thresholds for AdaBoost and 50 different population initializations for SBB-GP. WEKA is used for running the AdaBoost, the Linux model given at Quinlan (2010) is used for running the C5.0 and the C++ implementation given at SBB-GP (2008) is used for running SBB-GP learning algorithms. The parameters of the three algorithms are listed in Tables 2–4, respectively. We used 50 runs for each algorithm on each training data set to ensure that the outcomes are not based on one off trails but rather are based on statistically significant trials. Furthermore, the non-dominated solutions were selected out of the 50 models. The non-dominated solutions are the distinctive solutions that ranked the best model based on the high value of DR and the low value of FPR out of all models. Then, the best learner out of the non-dominated learners is chosen based on the highest performance (again the highest DR and the lowest FPR).

Table 2 C5.0 parameterization.

	Description	Value
r	Use rule-based classifiers	True
b	Use boosting	False
p	Use soft thresholds	True
e	Focus on errors	True
s	Find subset tests for discrete attributes	False
c	Confidence Factor for pruning	5–54

Table 3 Weka parameterization for AdaBoost.

	Description	Value
classifier	The base classifier to be used	DecisionStump
numIterations	Number of iterations	10
seed	The random seed number	1
useResampling	Use resampling instead of reweighting	False
weightThreshold	Weight Threshold (default 100)	10–250

Table 4 SBB based GP parameterization.

	Description	Value
P_{size}	Point population size	90
M_{size}	Team population size	90
$t_{\max}$	Number of generations	30,000
p_{d}	Probability of learner deletion	0.1
p_{a}	Probability of learner addition	0.2
μ_{a}	Probability of learner mutation	0.1
ω	Maximum team size	30
P_{gap}	Point generation gap	30
M_{gap}	Team generation gap	60

3.5. Traces deployed

To show the effectiveness of the proposed approach, completely different data sets are employed for training and testing the classifiers. We have employed three network traffic traces captured on our campus network (Univ2007 and Univ2010) and our lab (2009 and 2010). Our campus network features a full-duplex T1 fiber optic link for the Internet connection, where these traces were captured. A commercial deep packet classifier, PacketShaper (2008), is used to label both traces. Furthermore, the traces are anonymized and the payload is removed because of the privacy issues.

We generated VoIP traffic using different applications on a testbed that we set up in the NIMS Lab in 2009 and 2010 at the University. This testbed involved several PCs connected

through the Internet and several network scenarios were emulated using many popular VoIP applications (e.g. Gtalk (Gtalk, 2009), Primus (Primus, 2009), Yahoo messenger (Yahoo, 2009)) and other background traffic such as Virtual Private Network (VPN) traffic, c torrent traffic and web TV and radio were also included.

Moreover, the effects (if any) of different types of access technologies (i.e. WiFi versus Ethernet) were also investigated, as well as their different combinations. Overall, we have conducted in 2009 over 200 experiments equivalent to more than 50 h of VoIP traffic and non-VoIP traffic. In these experiments, we generated and captured more than 61 GB of traffic at both ends, where approximately 32 billion packets were transmitted. This data set was made public at Alshamamri (2011).

In all cases, we have performed experiments under several different network scenarios. These scenarios include: (i) Firewall restrictions at one user end and no restrictions at the other end; (ii) Firewall restrictions at both ends; (iii) No restrictions at both ends; (iv) Use of wireless and wire-line connections; (v) Blocking of all UDP connections, and (vi) Blocking of all TCP connections. It should be noted here that during these experiments all the Internet communications went through the network's firewall. The firewall was configured to permit access to the aforementioned restrictions such as do not permit anything, or permit limited well known port numbers such as port 22, 53, 80 and 443. Wireshark (2008) and Peeker (2009) were used to monitor and control network traffic. NetPeeker was used to block ports and to allow either both TCP and UDP traffic, or only UDP, or TCP traffic in order to analyze the behavior of the VoIP application clients. On the other hand, Wireshark was used to capture traffic from both ends of the communication.

The general call set up between the caller and callee for voice calls is as follows: caller transmits a standard audio file to callee. We used an English spoken text (male and female audio files) without noise and a sample rate of 8 Hz, which was encoded with 16 bits per sample and can be downloaded at Signalogic (2009). The wav-file was played and then the output of Windows media player was used as input for VoIP application clients using a microphone. Wireshark was used to capture the traffic from both users' ends.

Brief statistics on the traces collected are given in Table 5. This shows that the data sets have different general properties based on the total number of flows and packets. Moreover, the data set is huge in size and we do not have enough resources in terms of memory and computational power to construct model from the entire data. Therefore, the training data are going to be chosen from a subset of the data.

4. Selecting training data sets

Weiss and Provost (2003) has pointed out the importance for the subset sampling and its effects on the performance of the classifier during training. We have evaluated three different sampling methods for selecting training data sets. These are: (i) uniform random N sampling, where N is either a fixed number of records (e.g. 30 K, 60 K, etc.); or N is a fixed percentage of records (e.g. 1%, 2%, etc.); (ii) stratified N sampling based on grouping, where N is either a fixed number of records (e.g. 30 K, 60 K, etc.); or N is a fixed percentage of records (e.g. 1%, 2%, etc.); and (iii) continuous data streams of either a specific time period (such as 30 min, 60 min and 90 min of traffic) or N sampling records (e.g. 30 K, 60 K, etc.). All random samplings are performed using uniform probability. Since the goal is to investigate which one of these techniques will enhance the automatic generation of robust signatures for classifying unknown VoIP traffic, the training data set was limited to be a subset of Univ2007, while test data sets consist of the rest of Univ2007 and the Univ2010 data sets.

4.1. Uniform random N sampling method

Random N packets are sampled with uniform probability from the Univ2007 trace where there exist two classes. The two classes are Skype, representing the in-class, and non-Skype, representing the out-class. The non-Skype class includes all the network applications in the traces. Since the focus is to differentiate Skype VoIP encrypted traffic from non-Skype traffic, six training data sets with different N number of packets were sampled randomly with uniform probability. For example, when N is equal to 30 K – 30,000 flows –, 15,000 flow records from Skype and 15,000 flow records from non-Skype classes were sampled randomly for a total of 30,000 records. For the fixed number of records, six different N values were used: 30 K, 60 K, 100 K, 200 K, 400 K and 800 K. Six other different training data sets were sampled randomly with uniform probability where N represents a fixed percentage of records. The six different N values used are: 1%, 2%, 3%, 4%, 5%

Table 5 Brief statistics of network traffic traces used (in millions).

	Packets	Bytes	Flows
Univ2007	337 M	213 M	28 M
Univ2010	1838 M	1330,169 M	46 M
NIMSII: GTALK_2009	34 M	6,492 M	190,665
NIMSII: PRIMUS_2009	1 M	384 M	7529
NIMSII: Zfone_2009	1 M	138 M	28,553
NIMSIII: GTALK_2010	384 M	1256 M	14,847
NIMSIII: PRIMUS_2010	7 M	1367 M	21,802
NIMSIII: YAHOO_2010	8 M	1080 M	23,239
NIMSIII: Torrent_2010	21 M	17,791 M	412,345
NIMSIII: Radio_2010 stream	332,183	272 M	2236
NIMSIII: TV_2010 stream	5 M	4941 M	1803
NIMSIII: VPN_2010	32,079 M	26,728 M	74,302

and 6%. For instance, when N is equal to 1%, 1% of flow records from Skype and 1% of flow records from non-Skype classes were sampled randomly.

4.2. Stratified sampling method

Stratified sampling uses a priori information to explore whether this would improve the performance of classification methods by using grouping techniques. In other words, this technique investigates whether including applications which exhibit behavior similar to the Skype application to the training data set makes any difference in the training performance or not. In this case, the Univ2007 data sets are grouped so that each cluster contains data with similar properties. After that the classes (network applications e.g, FTP, HTTP, etc.) in each cluster are determined in order to select them for constructing the training data set. In order to build the clusters for the Univ2007 data sets, Self organizing feature maps (SOMs) (Kohonen, 1990) are employed. This is a well-known unsupervised learning technique, which is used to cluster and visualize high dimensional data into a topographical two-dimensional grid structure based on a neural network model (Ultsch, 1999).

4.2.1. Self organizing feature maps (SOMs)

SOMs are unsupervised neural networks which transform arbitrarily high dimensional input data space (n dimensional input data vector) to a low dimensional space that is most commonly viewed as neurons of two dimensional array. The aim of the SOM is to discover the fundamental structure of the input data space (feature map) while maintaining the properties of the input space. SOM builds a topologically preserving map which presents a visual arrangement of the neighbouring relationships of the points in the input data set where a human can simply notice groups/clusters and relations.

The learning process of the SOM starts by selecting randomly a sampled vector x from the input and calculating all the weight vectors based on a distance measurement. The Best Matching Unit (BMU) is the neuron which has the shortest distance to the input vector x, Eq. (3):

$$\|x - m_c\| = \min\{\|x - m_i\|\} \tag{3}$$

where x represents the input vector, w represents the weight vector, c represents the BMU and $\{\|\|\}$ represents the distance measure. In this research paper, the Euclidian distance was used. After finding the BMU, all weight vectors (neurons) are revised to make the BMU moving closer to the input vector, Eq. (4):

$$w_i(t+1) = w_i(t) + \alpha(t)h_{ci}(t)[x(t) - w_i(t)] \tag{4}$$

where the weight vector, $wi(t)$, specifies the location of the output unit index, i, in the data space at a time t. The learning rate is specified by α and h_{ci} is the neighborhood kernel close to the c (BMU). After convergence is reached, the resulting map is ordered topologically. Further details about the SOM can be found in Kohonen (2001). In order to apply SOM, the input data set needs to be normalized to prevent certain variables (features, e.g. the min_fpktl value) from having a higher impact than the other variables. This normalization will transform all the variables to be between 0 and 10 (log normalization).

In this paper, the SOM PAK package with the SOM Toolbox (Vesanto et al., 2000; Teuvo Kohonen and Kangas, 2000) is used to carry out the SOM-based experiments. The SOM PAK is written in c++ and can handle easily large data set while the SOM Toolbox is used to visualize the map. Clustering using a SOM involves experimenting with different parameters. The main parameters are the learning rate, the maximum number of iterations and dimensions of the map. The map dimensions have an effect on the number of clusters (units) the SOM generates. Since the data sets are relatively large, a bigger map size is needed. In this case, it is 6×6 (36 map units). The parameters for training the map are listed in Table 6.

After training of the SOM is finished, a Unified distance matrix is employed to visualize the grouping structure of the high weight vectors between neurons. U-matrix is a color-heated map which plots the distances of SOM neurons (Fig. 1). The color-heated map ranges from dark red through shades of yellow and green to dark blue, where red implies high values and blue implies low values. A dark red color means a large distance between neurons which indicates heterogeneous neighborhoods while a dark blue color means a small distance which indicates homogeneous neighborhoods. The dark color can represent the cluster separators while the light

Table 6 Parameters of the SOM.

Parameters	Values
X dimension	6
Y dimension	6
radius1	2
radius2	1
data Length (rlen_1)	rlen_1 multiply by 100
data Length (rlen_2)	rlen_1 multiply by 1000
alpha type	inverse_t
neighborhood	gaussian
alpha1	0.5
alpha2	0.05
topology	hexa

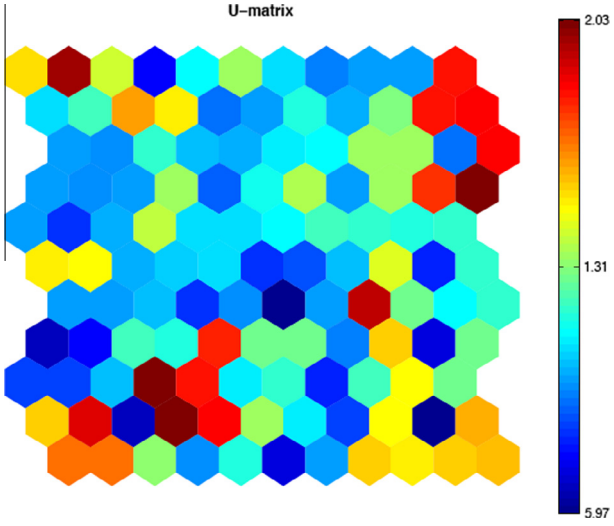


Fig. 1 Unified distance matrix.

color can represent the clusters. This color schema is useful when trying to find clusters in the data set without a priori knowledge.

Although the SOM is an unsupervised learning algorithm (in other words during training labels are not used), in post training the Univ2007 data labels are input to the SOM to study the following: (i) the number of applications in each neuron; (ii) how spread is the Skype application on the map; and (iii) how many applications are similar to Skype. Fig. 2 shows that Skype is in 11 neurons and shared similarities with eight applications (OTHER, MSN, HTTPS, HTTP, DNS, MAIL, SSH and FTP). This is because a Skype application is trying to mimic the behavior of other applications to avoid detection. Thus, based on this observation, the OTHER, MSN, HTTPS, HTTP, DNS, MAIL, SSH and FTP applications are going to be used as the out-class when sampling the training data set for the stratified sampling method ('a priori' method).

Table 7 lists the number of flow records for each of the nine applications (OTHER, SKYPE, MSN, HTTPS, HTTP, DNS, MAIL, SSH and FTP) on the Univ2007 data set. In this case, based on the SOM analysis, data are sampled randomly with uniform probability in different N fixed size samples and different N percentages from nine different applications (OTHER, SKYPE, MSN, HTTPS, HTTP, DNS, MAIL, SSH and FTP) to form both the in-class and the out-class for the training data set, the data set is balanced so that the in-class and out-class have the same number of flows. For the fixed size of N records, the number of out-class flows is divided by the number of out-class applications (e.g. for the 30 K classes 15,000/8 = 1875 flows for each class). If an application has a fewer number of flows in the sampled data set than the required number necessary for sampling, then, the missing flows would be sampled randomly from the OTHER class. For instance there were 7684 FTP flows in the Univ2007 data set, when sampled for the 200 K class training data set. The number of FTP flows is less than the 12,500 allocation for the FTP application so the missing flows would be taken from the OTHER class.

Table 7 Number of flows for each application in the Univ2007 trace.

Applications	Number of flows
FTP	7684
SSH	18,993
MAIL	359,430
DNS	5,032,876
HTTP	5,670,386
HTTPS	1,144,505
MSN	344,408
OTHER	8,146,792
SKYPE	8,254,782

4.3. Continuous data stream

Network traffic traces are a real-time continuous stream of packets which are ordered explicitly by the timestamp of the packets. Typically, these continuous data streams have unique characteristics which depict the network infrastructure and user behavior. In Fig. 3, there are different peeks for TCP and UDP traffic for the Univ2007. For instance, at 16:30

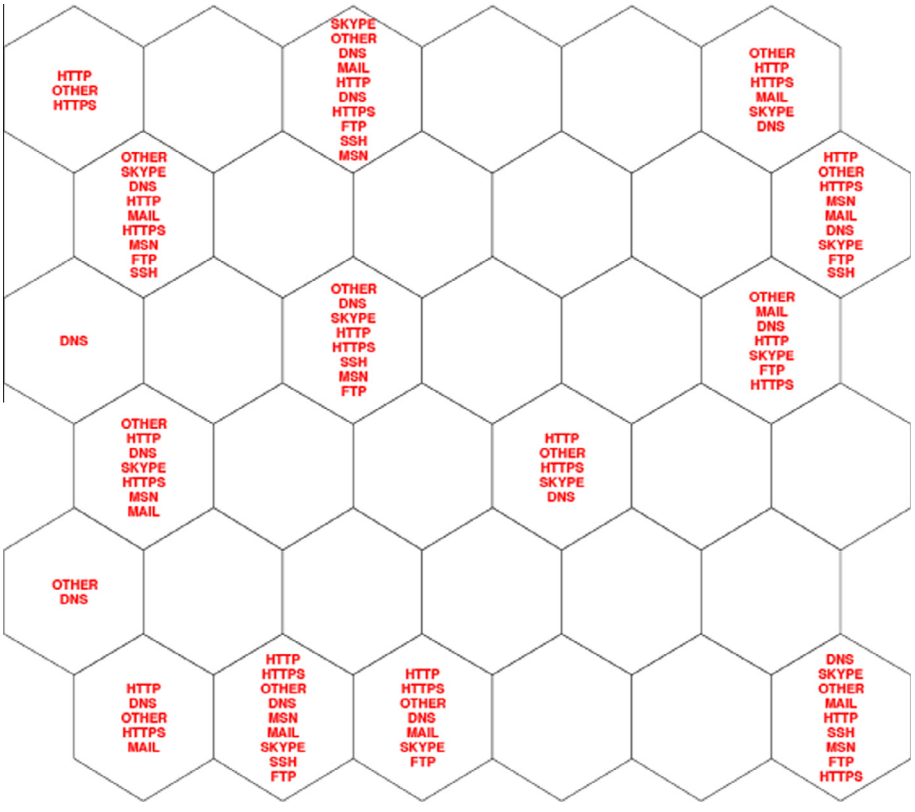


Fig. 2 Distribution of applications in each neuron.

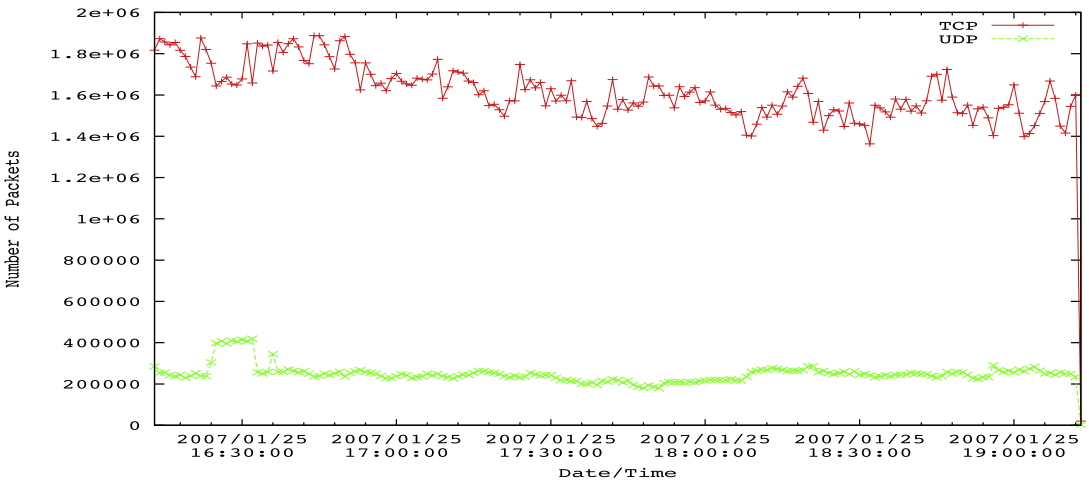


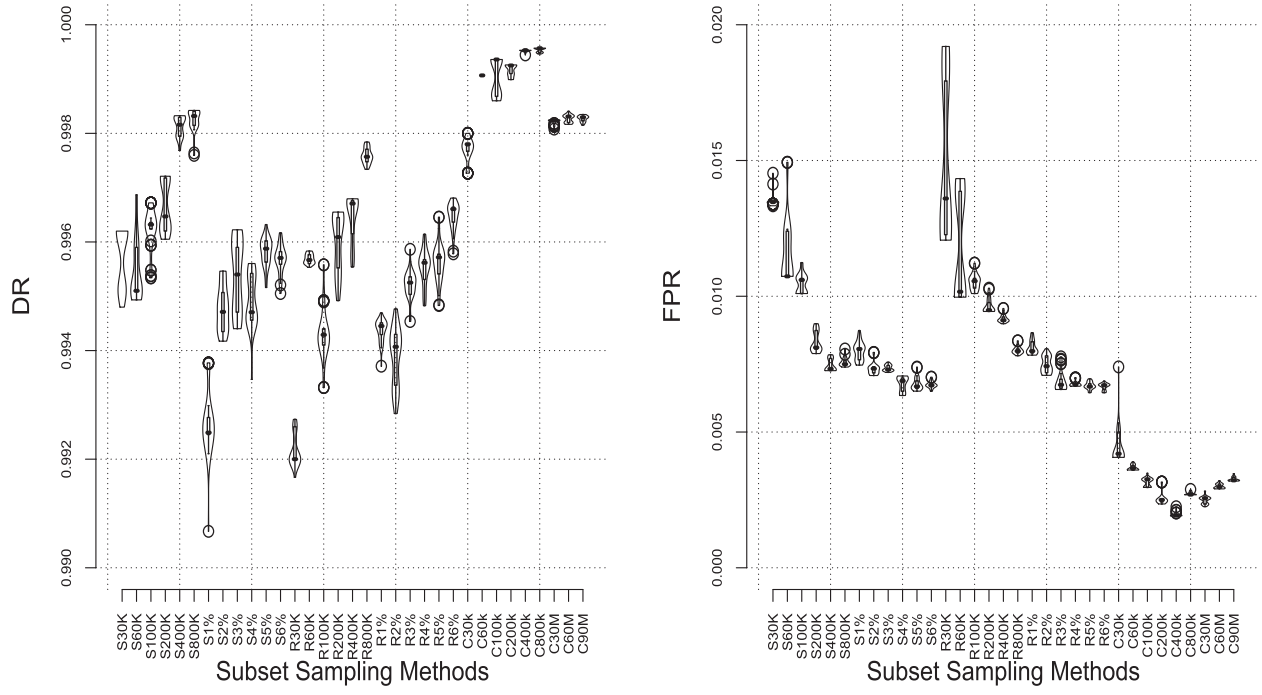
Fig. 3 Number of packets for TCP/UDP protocols in the Univ2007 trace.

PM there is an increase in the number of UDP packets while there is a decrease in the number of TCP packets. Moreover, the number of TCP packets fluctuates on the trace which shows that the University users tend to use more applications that run on TCP than UDP.

To capture this behavior in the training data set the flows were sampled based on the order in which they arrived in two different techniques. The first technique used is to sample a fixed size number of records as in the previous two Sections (4.1 and 4.2). For example, for the "First 30 K" method the first 15,000 Skype flow records and the first 15,000 non-Skype flow records were selected. The second technique used involves sampling over a continuous time period (e.g. first 30 min, first 60 min, etc.).

5. Results of experiments for subset sampling

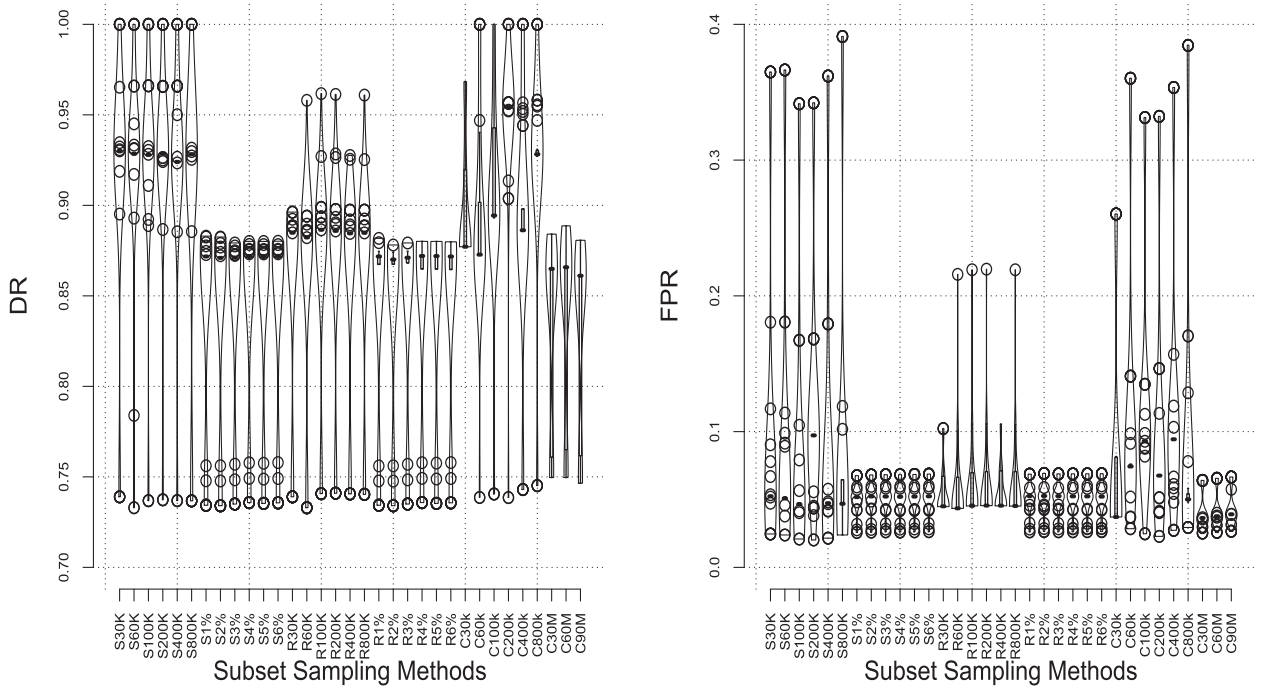
In total, 33 training data sets were sampled using the three subset sampling methods where the sizes of the training data sets vary from thousands of flow records to millions of flow records (e.g. sizes from 30,000 flow records to 14,554,340 flow records). For these experiments each classifier was trained initially on the training subset that is sampled from the Univ2007 flow data sets represented by the feature set given in Table 1. After that, the generated models of C5.0, AdaBoost and GP were tested on a validation data set, which is a subset of the Univ2010 test traces. The validation data set consists of randomly sampled (with uniform probability) 1000 flow records



(a) DR (b) FPR

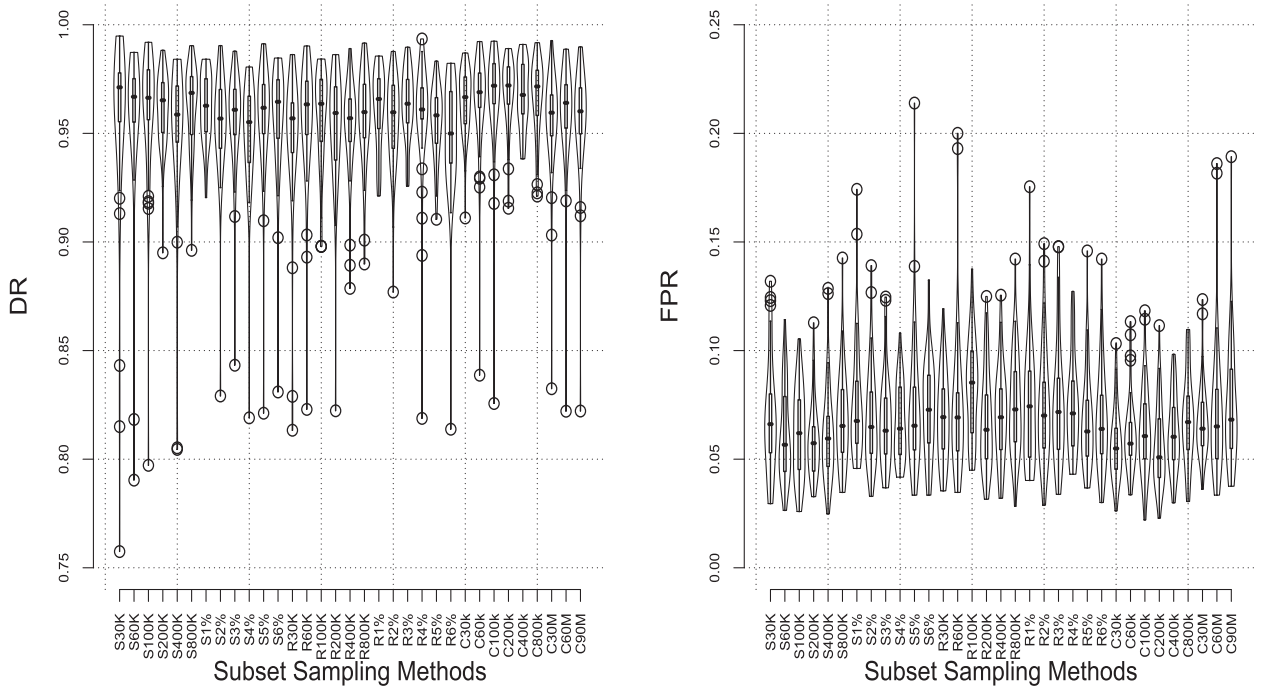
Fig. 4 Performance of C5.0 on the training data set for Skype. S = stratified, R = uniform random and C = continuous.

of ten applications from the Univ2010 trace for a total of 10,000 records. The ten applications were OTHER, SKYPE, P2P, MSN, HTTPS, HTTP, DNS, MAIL, SSH and FTP. The validation data were used to evaluate the most appropriate subset sampling method for generating generalized/robust signatures. Since the Univ2010 data set is a real network trace captured from the same location as the Univ2007 traces but at a different time period (in 2010 as opposed to in 2007) and



(a) DR (b) FPR

Fig. 5 Performance of AdaBoost on the training data set for Skype. S = stratified, R = uniform random and C = continuous.



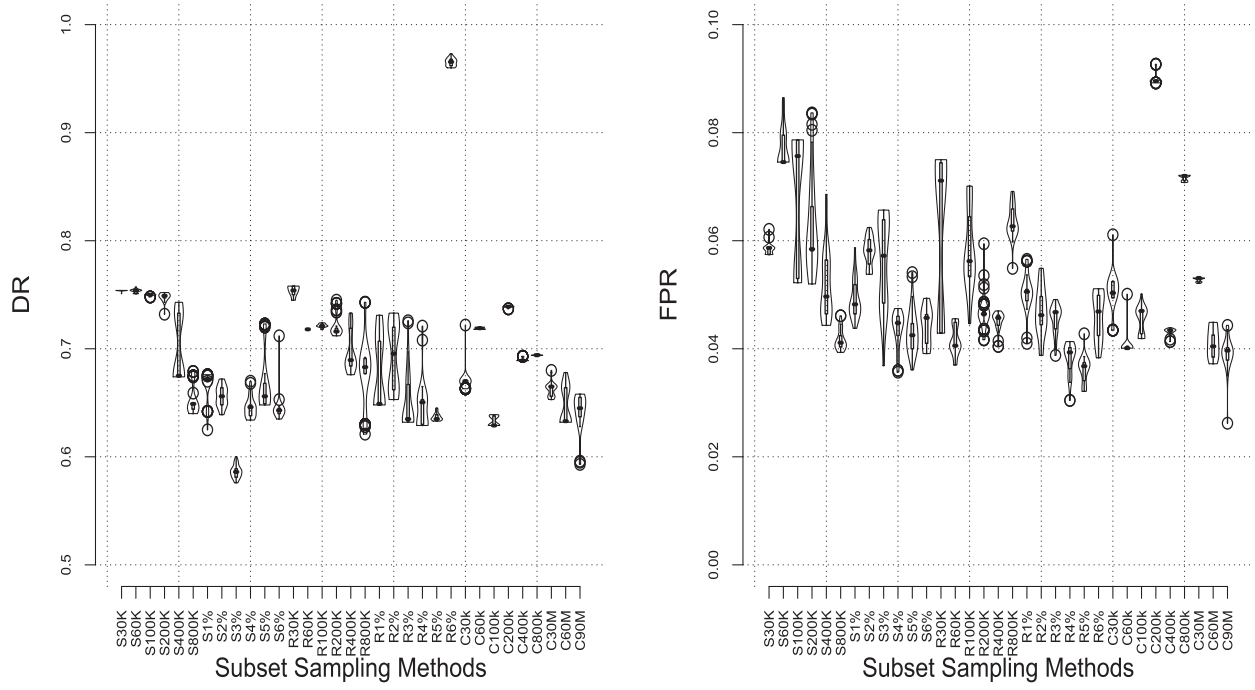
(a) DR (b) FPR

Fig. 6 Performance of GP on the training data set for Skype S = stratified, R = uniform random and C = continuous.

contains many applications, we consider it to be suitable for validating and testing the robustness of the classifiers.

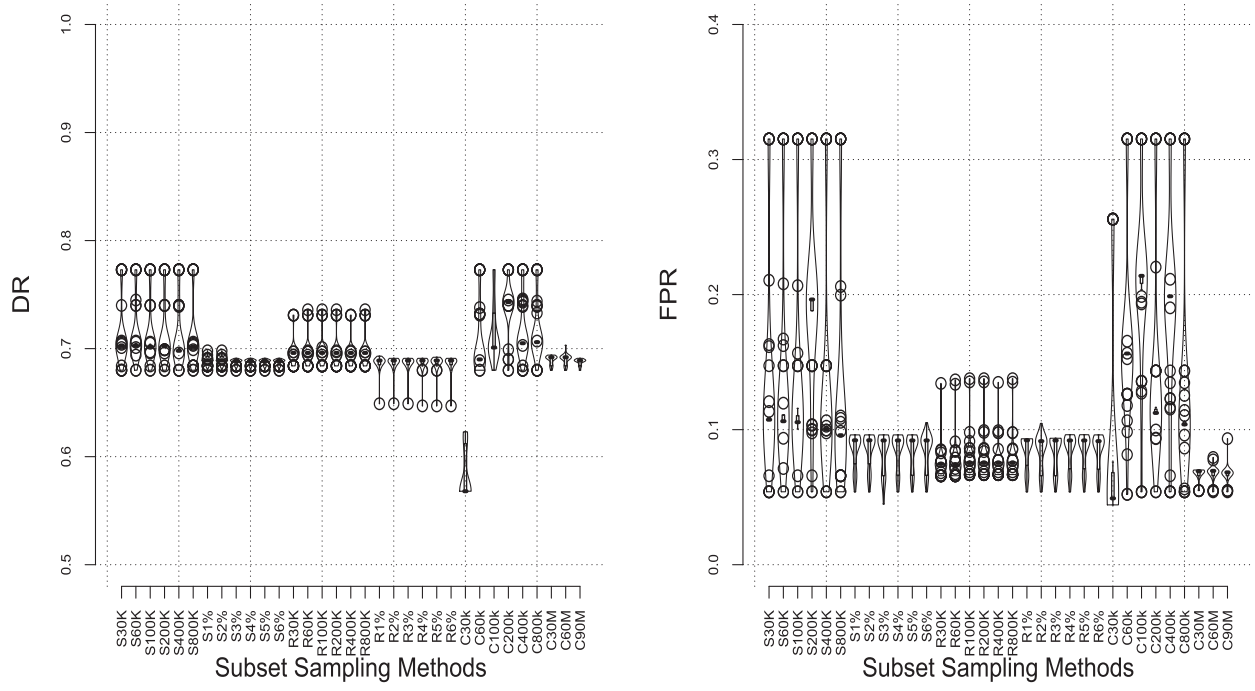
The training performances and the validation performances of all the 50 runs of each technique on each of the 33 training data sets are given using density of distribution/box plots for the uniform random N sampling method, stratified sampling method and continuous sampling method in Figs. 4–9, respectively.

It should be noted here that the best solution was selected based on the performance in terms of high DR value and low



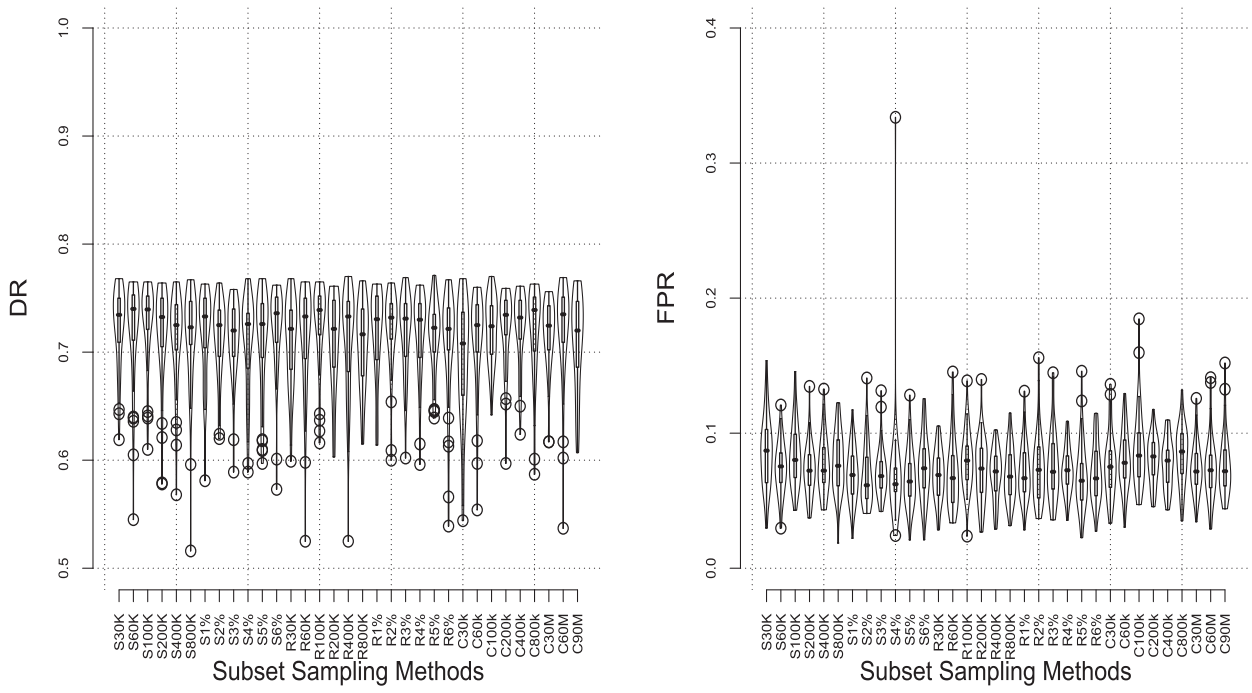
(a) DR (b) FPR

Fig. 7 Performance of C5.0 on the validation data set for Skype. S = stratified, R = uniform random and C = continuous.



(a) DR (b) FPR

Fig. 8 Performance of AdaBoost on the validation data set for Skype. S = stratified, R = uniform random and C = continuous.



(a) DR (b) FPR

Fig. 9 Performance of GP on the validation data set for Skype. S = stratified, R = uniform random and C = continuous.

FPR value obtained on the training for C5.0, AdaBoost and GP. Then the three learning classifiers are evaluated on the validation data. In summary, all models generated by the learning algorithms are constructed on a subset of the Univ2007 data set (training data) while the Post evaluation of the all the models was conducted using the validation data, which is a subset of Univ2010 data set where none were encountered during training. Table 12 lists the performances of the best models obtained on the validation data. Results on the validation data set have shown that C5.0 achieved the best performance as well as constantly obtained a lower FPR value and a higher DR value while using the uniform random sampling with the 6 percent (97% DR and 0.04% FPR). Moreover, the performance of the subset sampling techniques was compared by using a one-way ANOVA test based on the values of the DR and the FPR (one-way ANOVA test where $n = 50$ data points). One-way ANOVA statistical test shows that the mean of 50 runs of C5.0-based classifiers using the uniform random sampling with 6 percent is statistically significantly better than the mean of the 50 runs of the other learning algorithms on both the validation and training data sets, Tables 8–11. Thus the uniform random sampling method with 6 percent was chosen as the method for sampling the training dataset.

5.1. Results of experiments for the best subset sampling technique on the test data sets

The performance of the three trained models (C5.0, AdaBoost and GP) was tried out on the test data sets, namely the unseen Univ2007 test traces and the unseen Univ2010 traces.

The training performances of all the fifty runs for three learning classifiers were graphed using density of distribution/box plots (Fig. 10). By contrast, GP and AdaBoost have

Table 8 A one-way ANOVA statistical analysis test for the mean DR for the three learning algorithms for the subset sampling techniques on the training data set.

Source	SS	df	MS	*F*	Prob > *F*
Columns	15.7262	98	0.1605	125.4834	0
Error	6.2036	4851	0.0013		
Total	21.9297	4949			

Table 9 A one-way ANOVA statistical analysis test for the mean FPR for the three learning algorithms for the subset sampling techniques on the training data set.

Source	SS	df	MS	*F*	Prob > *F*
Columns	5.3618	98	0.0547	39.5126	0
Error	6.7171	4851	0.0014		
Total	12.0790	4949			

Table 10 A one-way ANOVA statistical analysis test for the mean DR for the three learning algorithms for the subset sampling techniques on the validation data set.

Source	SS	df	MS	*F*	Prob > *F*
Columns	8.2791	98	0.0845	117.6585	0
Error	3.4831	4851	7.1801e−04		
Total	11.7622	4949			

different range values in terms of DR and FPR values that implies both learning algorithms discover diverse solutions on different runs. The best performing solution was selected based on the performance in terms of a high DR value and low a FPR value obtained on the training for C5.0, AdaBoost and GP. Then the three learning classifiers are evaluated on the test data sets. In other words, all models from the learning algorithms are constructed from the Univ2007 Training data while Post evaluation of all the models was conducted on Univ2007 test and Univ2010 data sets, which none of the models generated by the three learning algorithms encountered during the training phase. The training performance was plotted as a scatter plot for the three classifiers, Figs. 11a–c. There are five non-dominated solutions for GP, four non-dominated solutions for AdaBoost and four non-dominated solutions for C5.0. The solution with the highest DR value and the lowest FPR was selected out of the non-dominated solutions each classifier to be tested on the test data sets.

Table 11 A one-way ANOVA statistical analysis test for the mean FPR for the three learning algorithms for the subset sampling techniques on the validation data set.

Source	SS	df	MS	*F*	Prob > *F*
Columns	4.9024	98	0.0500	63.1761	0
Error	3.8411	4851	7.9182e−04		
Total	8.7435	4949			

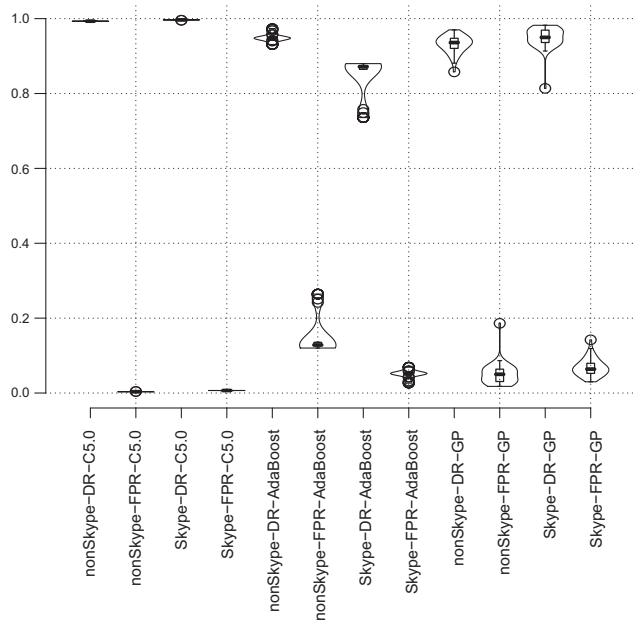
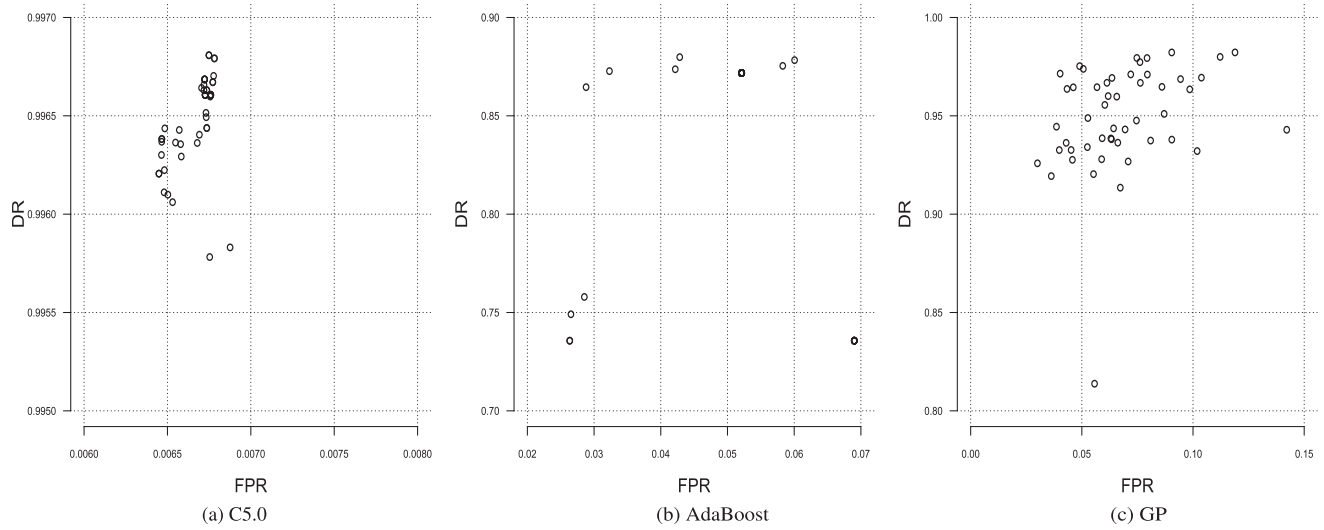


Fig. 10 DR and FPR results for Skype identification on the training data sets.

The results of the best models are shown on Table 13 on the test data sets. C5.0 achieved the highest performance with a better (high) DR value and a lower FPR value on the test data sets compared to AdaBoost and GP classifiers.

In summary, the effect of using three sampling techniques was investigated with a total of 33 training data sets using three learning algorithms, namely, C5.0, AdaBoost and GP. These results indicate that the rules (signatures) generated by the C5.0 classifier during training, are robust (transportable) enough in order to test on other network traffic data. Furthermore, our results also indicate that it is possible to have a well generalized (robust) set of signatures that are automatically generated with a standard set of features which can be used to classify encrypted VoIP Skype traffic.

The validation data were used to evaluate the most appropriate subset sampling method for generating generalized/robust signatures not to assess the performance of the three learning algorithms. Furthermore, the size of the test data sets is huge and therefore, evaluating the 33 training data sets on the test data sets would have required a very long time. Thus, the best training data set is selected through the results of the validation process. Once the best one is selected, it is next assessed on the test data sets. Since the Univ2010 data set is a real network trace captured from the same location as the Univ2007 traces but at a different time period and contains many applications, it is the most suitable one for validating and testing the robustness of the classifiers.



(a) C5.0 (b) AdaBoost (c) GP

Fig. 11 Scatter plot of the performance of the C5.0 classifier on the training data set for Skype classification (DR versus FPR).

Table 12 Results of the best models for each classifier on the validation data.

	C5.0		AdaBoost		GP	
	DR	FPR	DR	FPR	DR	FPR
Uniform random N sampling						
30 K	0.76	0.04	0.73	0.08	0.76	0.07
60 K	0.72	0.04	0.73	0.08	0.76	0.04
100 K	0.72	0.05	0.73	0.08	0.76	0.07
200 K	0.72	0.05	0.73	0.08	0.76	0.07
400 K	0.73	0.04	0.73	0.08	0.76	0.08
800 K	0.74	0.06	0.73	0.08	0.74	0.05
1%	0.73	0.05	0.68	0.05	0.76	0.06
2%	0.73	0.05	0.68	0.05	0.76	0.07
3%	0.72	0.05	0.68	0.05	0.74	0.04
4%	0.72	0.03	0.68	0.05	0.75	0.06
5%	0.65	0.03	0.68	0.05	0.75	0.04
6%	0.97	0.04	0.68	0.05	0.74	0.06
Stratified sampling						
30 K	0.75	0.06	0.68	0.05	0.76	0.07
60 K	0.76	0.08	0.68	0.05	0.77	0.08
100 K	0.75	0.05	0.68	0.05	0.76	0.09
200 K	0.75	0.05	0.68	0.05	0.76	0.08
400 K	0.74	0.07	0.68	0.05	0.76	0.09
800 K	0.68	0.04	0.68	0.05	0.77	0.06
1%	0.68	0.05	0.68	0.07	0.76	0.06
2%	0.67	0.06	0.68	0.05	0.76	0.07
3%	0.60	0.05	0.68	0.05	0.76	0.09
4%	0.66	0.04	0.68	0.05	0.75	0.07
5%	0.72	0.04	0.68	0.05	0.76	0.05
6%	0.71	0.04	0.68	0.07	0.75	0.08
Continuous data streams						
30 K	0.67	0.05	0.61	0.06	0.74	0.08
60 K	0.72	0.04	0.68	0.05	0.76	0.08
100 K	0.64	0.04	0.68	0.05	0.77	0.06
200 K	0.74	0.09	0.68	0.05	0.75	0.07
400 K	0.69	0.04	0.68	0.05	0.75	0.06
800 K	0.69	.07	0.71	0.06	0.75	0.07
30 min	0.68	0.05	0.68	0.05	0.75	0.07
60 min	0.68	0.04	0.70	0.08	0.75	0.07
90 min	0.65	0.04	0.68	0.05	0.77	0.06

Table 13 Best model results for Skype classification on the university data sets (training and testing data).

	C5.0		AdaBoost		GP	
	DR	FPR	DR	FPR	DR	FPR
Training Sample (subset of Univ2007)						
Non-SKYPE	0.993	0.004	0.957	0.120	0.936	0.031
SKYPE	0.993	0.005	0.957	0.120	0.969	0.064
Univ2007 Test data sets						
Non-SKYPE	0.993	0.005	0.957	0.120	0.936	0.031
SKYPE	0.995	0.007	0.880	0.043	0.969	0.064
Univ2010 Test data sets						
Non-SKYPE	0.956	0.169	0.932	0.189	0.922	0.144
SKYPE	0.831	0.044	0.811	0.068	0.856	0.078

5.2. Multi-classes classification for VoIP applications

In this section, three VoIP applications are employed. These applications are Skype, Gtalk and Primus softphone. To show the effectiveness of the proposed approach, evaluations are performed on different training and test data sets. The solution robustness is assessed when the training data are sampled from two data sets (Univ2007 and NIMSII traces) while testing is occurred on traces from different locations (Univ2007 and NIMSII test partitions and Univ2010, which were captured in

Table 14 Results for C5.0 classifier – multi-class – all traces.

	SKYPE		Non-VoIP		GTALK		PRIMUS	
Data Sets	DR	FPR	DR	FPR	DR	FPR	DR	FPR
Training	0.997	0.007	0.993	0.004	0.962	0.000	0.951	0.086
Univ2007	0.996	0.007	0.993	0.004	0.000	0.000	0.000	0.000
Univ2010	0.803	0.038	0.962	0.197	0.000	0.001	0.000	0.000
GTALK2009	0.000	0.002	0.000	0.035	0.963	0.000	0.000	0.000
PRIMUS2009	0.000	0.000	0.000	0.056	0.000	0.002	0.942	0.000
ZFONE2009	0.000	0.059	0.751	0.000	0.000	0.164	0.000	0.027
GTALK2010	0.000	0.012	0.000	0.074	0.914	0.000	0.000	0.000
PRIMUS2010	0.000	0.022	0.000	0.049	0.000	0.014	0.915	0.000
YAHOO2010	0.000	0.082	0.902	0.000	0.000	0.016	0.000	0.000
RADIO2010	0.000	0.003	0.986	0.000	0.000	0.012	0.000	0.000
TORRENT2010	0.000	0.040	0.921	0.000	0.000	0.039	0.000	0.000
TV2010	0.000	0.008	0.987	0.000	0.000	0.005	0.000	0.000
VPN2010	0.000	0.000	1.000	0.000	0.000	0.000	0.000	0.000

Table 15 Results for GP classifier – multi-class – all traces.

	SKYPE		Non-VoIP		GTALK		PRIMUS	
Data Sets	DR	FPR	DR	FPR	DR	FPR	DR	FPR
Training	0.941	0.078	0.977	0.051	0.963	0.029	0.954	0.008
Univ2007	0.950	0.098	0.858	0.048	0.000	0.029	0.000	0.016
Univ2010	0.816	0.046	0.930	0.085	0.000	0.122	0.000	0.001
GTALK2009	0.000	0.004	0.000	0.033	0.962	0.000	0.000	0.000
PRIMUS2009	0.000	0.001	0.000	0.016	0.000	0.001	0.983	0.000
ZFONE2009	0.000	0.043	0.669	0.000	0.000	0.288	0.000	0.000
GTALK2010	0.000	0.051	0.000	0.048	0.901	0.000	0.000	0.000
PRIMUS2010	0.000	0.074	0.000	0.040	0.000	0.002	0.884	0.000
YAHOO2010	0.000	0.000	0.910	0.000	0.000	0.074	0.000	0.017
RADIO2010	0.000	0.001	0.999	0.000	0.000	0.000	0.000	0.000
TORRENT2010	0.000	0.006	0.932	0.000	0.000	0.044	0.000	0.019
TV2010	0.000	0.001	0.982	0.000	0.000	0.008	0.000	0.009
VPN2010	0.000	0.053	0.947	0.000	0.000	0.000	0.000	0.000

Table 16 Results for AdaBoost classifier – multi-class – all traces.

Data Sets	SKYPE		Non-VoIP		GTALK		PRIMUS	
	DR	FPR	DR	FPR	DR	FPR	DR	FPR
Training	0.734	0.049	0.951	0.266	0.000	0.000	0.000	0.000
Univ2007	0.747	0.027	0.973	0.253	0.000	0.000	0.000	0.000
Univ2010	0.710	0.067	0.933	0.290	0.000	0.000	0.000	0.000
GTALK2009	0.000	0.005	0.000	0.995	0.000	0.000	0.000	0.000
PRIMUS2009	0.000	0.001	0.000	0.999	0.000	0.000	0.000	0.000
ZFONE2009	0.000	0.235	0.765	0.000	0.000	0.000	0.000	0.000
GTALK2010	0.000	0.000	0.000	1.000	0.000	0.000	0.000	0.000
PRIMUS2010	0.000	0.321	0.000	0.679	0.000	0.000	0.000	0.000
YAHOO2010	0.000	0.002	0.998	0.000	0.000	0.000	0.000	0.000
RADIO2010	0.000	0.001	0.999	0.000	0.000	0.000	0.000	0.000
TORRENT2010	0.000	0.058	0.942	0.000	0.000	0.000	0.000	0.000
TV2010	0.000	0.001	0.999	0.000	0.000	0.000	0.000	0.000
VPN2010	0.000	0.013	0.987	0.000	0.000	0.000	0.000	0.000

2007, 2009 and 2010, respectively). In these experiments, the training data set is labeled into multi-classes depending on VoIP applications (SKYPE, GTALK, PRIMUS, and non-VoIP). It should be noted here that 6% of the GTALK2009 and PRIMUS2009 data sets are sampled and added to the training data set as described in Section 4.1. Thus, the training data set con-

sists of SKYPE, non-VoIP, GTALK, and PRIMUS applications where each of them contains the following number of flows: 646,521, 1,235,055, 11,417, and 451, respectively.

Results are summarized in terms of DR and FPR. Tables 14–16 list the results for the three machine learning algorithms on the training and independent test traces. In this case, results show that C5.0 performs much better than GP and AdaBoost algorithms in classifying multiple VoIP applications. C5.0 achieves for Skype ≈100% DR and ≈1% FPR on Univ2007 Test partition, and ≈80% DR and ≈4% FPR on Univ2010 traces,. For Gtalk, C5.0 achieves ≈96% DR and ≈0% FPR on NIMSII traces, ≈91% DR and ≈0% FPR on NIMSIII traces. For Primus, C5.0 achieves ≈94% DR and ≈0% FPR on NIMSII traces. Moreover, the C5.0 classifier is the most consistent performer across all test and training conditions, while also being competitive with GP for Skype detection under university traces and Gtalk detection under NIMS traces. This not only shows that the model, which the C5.0 classifier learned during training is robust (generalized) enough to be tested on real world network traces, but also verifies that accurate differentiation between multiple VoIP applications is possible without employing port numbers, IP addresses and payload information.

6. Conclusion

The primary motivation addressed in this research paper is the challenging problem of finding robust rules (signatures) specifically to detect encrypted VoIP Skype network traffic. The classification of Skype VoIP traffic is viewed as a fundamental task for any network operations management group since it is essential for managing bandwidth budgets and to ensure QoS for important applications. This paper investigates how to form a training set when a machine learning based approach is used (as opposed to conventional approaches such as port numbers based classification or deep packet based inspection) for classifying network traffic without including port numbers, IP addresses, or payload information. To do so, traffic traces from our campus network were used.

Three different sampling techniques were studied on three learning algorithms (C5.0, AdaBoost and GP) that were trained on all the training data sets, which were sampled from Univ2007 network traffic traces and tested on the Univ2010 traces. Results indicate that uniform random sampling is the most appropriate method for achieving our objective. Indeed, this is an interesting result because before this study one might have thought that the knowledge of a priori information could be critical for the preparation of training data sets for learning algorithms. However, our results seem to indicate that a priori information is resulting in "over learning" on our data sets. Given the results obtained in this research paper, one of the future directions which can be followed would be to explore whether a similar trend would be seen for other network applications.

Acknowledgement

The Natural Science and Engineering Research Council of Canada (NSERC) grant supported this research paper.

References

Alpaydin, E., 2004. Introduction to Machine Learning. MIT Press.

Alshamamri, R., 2011 Downloading the nims data sets. http://web.cs.dal.ca/riyad/Site/Download.html (last accessed September, 2011).

Alshammari, R., 2008. Automatically classifying encrypted network traffic: a case study of ssh (May 2008).

Alshammari, R., Zincir-Heywood, A.N., 2007. A flow based approach for ssh traffic detection. In: Proceedings of the IEEE International Conference on System, Man and Cybernetics – SMC'2007.

Alshammari, R., Zincir-Heywood, A.N., 2008. Investigating two different approaches for encrypted traffic classification. In: PST '08: Proceedings of the 2008 Sixth Annual Conference on Privacy, Security and Trust, IEEE Computer Society, Washington, DC, USA, pp. 156–166.

Alshammari, R., Zincir-Heywood, N., 2009b, Generalization of signatures for ssh encrypted traffic identification. In: Computational Intelligence in Cyber Security. IEEE Symposium on CICS '09. pp. 167–174.

Alshammari, R., Zincir-Heywood, A.N., 2009a. Machine learning based encrypted traffic classification: Identifying ssh and skype. In: Computational Intelligence for Security and Defense Applications, 2009. IEEE Symposium on CISDA 2009. pp. 1–8.

Alshammari, R., Zincir-Heywood, A.N., 2011. Can encrypted traffic be identified without port numbers, ip addresses and payload inspection? Comput. Networks 55 (6), 1326–1350.

Arndt, D., 2011. How to: Calculating flow statistics using netmate. http://dan.arndt.ca/nims/calculating-flow-statistics-using-netmate/ (last accessed September, 2011).

Bacquet, C., Gumus, K., Tizer, D., Zincir-Heywood, A., Heywood, M.I., 2010. A comparison of unsupervised learning techniques for encrypted traffic identification. J. Inf. Assurance Security 5, 464–472.

Bernaille, L., Teixeira, R., Akodkenou, I., Soule, A., Salamatian, K., 2006. Traffic classification on the fly. SIGCOMM Comput. Commun. Rev. 36 (2), 23–26.

Bonfiglio, D., Mellia, M., Meo, M., Rossi, D., Tofanelli, P., 2007. Revealing skype traffic: when randomness plays with you. SIGCOMM Comput. Commun. Rev. 37 (4), 37–48.

de Jong, E., 2007. A monotonic archive for pareto-coevolution. Evolut. Comput. 15 (1), 61–93.

Early, J., Brodley, C., Rosenberg, C., 2003. Behavioral authentication of server flows. In: Proceedings of the 19th Annual Computer Security Applications Conference. pp. 46–55.

Erman, J., Arlitt, M., Mahanti, A., 2006. Traffic classification using clustering algorithms. In: MineNet '06: Proceedings of the 2006 SIGCOMM Workshop on Mining Network Data. ACM Press, New York, NY, USA, pp. 281–286.

Erman, J., Mahanti, A., Arlitt, M., Cohen, I., Williamson, C., 2007. Offline/realtime traffic classification using semi-supervised learning. Perform. Eval. 64, 1194–1213.

Este, A., Gringoli, F., Salgarelli, L., 2009. Support vector machines for tcp traffic classification. Comput. Networks 53 (14), 2476–2490.

Freire, E., Ziviani, A., Salles, R., 2008. Detecting skype flows in web traffic. In: Network Operations and Management Symposium, 2008. NOMS 2008. IEEE. pp. 89–96.

Google, Google talk (gtalk). http://www.google.com/talk/ (last accessed October, 2009).

Haffner, P., Sen, S., Spatscheck, O., Wang, D., 2005. ACAS: automated construction of application signatures. In: MineNet '05: Proceeding of the 2005 ACM SIGCOMM Workshop on Mining Network Data. ACM Press, New York, NY, USA, pp. 197–202.

Huang, N.-F., Jai, G.-Y., Chao, H.-C., Tzang, Y.-J., Chang, H.-Y., 2013. Application traffic classification at the early stage by characterizing application rounds. Inf. Sci. 232, 130–142, http://

dx.doi.org/10.1016/j.ins.2012.12.039. URLhttp://www.sciencedirect.com/science/article/pii/S0020025513000315.

IETF. http://www3.ietf.org/proceedings/97apr/97apr-final/xrtftr70.htm.

Iliofotou, M., chul Kim, H., Faloutsos, M., Mitzenmacher, M., Pappu, P., Varghese, G., 2011. Graption: a graph-based p2p traffic classification framework for the internet backbone. Comput. Networks 55 (8), 1909–1920.

Karagiannis, T., Papagiannaki, K., Faloutsos, M., 2005. BLINC: multilevel traffic classification in the dark. In: SIGCOMM '05: Proceedings of the 2005 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. ACM Press, New York, NY, USA, pp. 229–240.

Kohonen, T., 1990. The self-organizing map. Proc. IEEE 78 (9), 1464–1480.

Kohonen, T., 2001. Self-organizing maps, 3rd ed. In: Springer Series in Information Sciences, vol. 30. pp. 501.

Lichodzijewski, P., Heywood, M.I., 2008. Managing team-based problem solving with Symbiotic Bid-based Genetic Programming. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 363–370.

Madhukar, A., Williamson, C., 2006. A longitudinal study of p2p traffic classification. In: Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2006. 14th IEEE International Symposium on MASCOTS 2006. pp. 179–188.

Moore, A.W., Papagiannaki, K., 2005. Toward the accurate identification of network applications. In: Passive and Active Network Measurement: Proceedings of the Passive & Active Measurement Workshop. pp. 41–54.

NetMate. http://www.ip-measurement.org/tools/netmate/.

PacketShaper, 2008. http://www.packeteer.com/products/p-acketshaper/ (last accessed March, 2008).

Peeker, N., 2009. Netpeeker. http://www.net-peeker.com (last accessed October, 2009).

P.T.C. Inc, Primus softphone client. http://www.primus.ca/en/residential/talkbroadband/talkBroadband-softphone.htm (last accessed October, 2009).

Quinlan, J., 2010. See5-info. http://www.rulequest.com/see5-info.html (last accessed July, 2010).

Quinlan, J., 2011. See5-comparison. http://www.rulequest.com/see5-comparison.html (last accessed February, 2011).

SBB-GP, 2008. Symbiotic bid-based (sbb) paradigm. http://www.cs.dal.ca/~mheywood/Code/SBB/SCM.9.r20081212.tar.gz (last accessed March, 2008).

Signalogic, 2009. Speech codec wav samples. http://www.signalogic.com/index.pl?page=codecsamples (last accessed October, 2009).

Skype. http://www.skype.com/useskype/.

Teuvo Kohonen, J.H., Kangas, J., 2000. SOMPAK: The Self-organizing Map Program Package. Tech. Rep., Helsinki University of Technology.

Ultsch, A., 1999. Data mining and knowledge discovery with emergent self-organizing feature maps for multivariate time series. In: Kohonen Maps. Elsevier, pp. 33–46.

Vesanto, E.A.J., Himberg, J., Parhankangas, J., 2000. Som Toolbox for Matlab. Tech. Rep., Helsinki University of Technology.

Weiss, G.M., Provost, F.J., 2003. Learning when training data are costly: the effect of class distribution on tree induction. J. Artif. Intell. Res. (JAIR) 19, 315–354.

T.U. of Waikato, WEKA software. http://www.cs.waikato.ac.nz/ml/weka/.

Williams, N., Zander, S., Armitage, G., 2006. A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification. SIGCOMM Comput. Commun. Rev. 36 (5), 5–16.

Wireshark, 2008. http://www.wireshark.org/ (last accessed September, 2008).

Yahoo!, 2009. Yahoo messenger. http://messenger.yahoo.com/ (last accessed October, 2009).