Unveiling Skype Encrypted Tunnels using GP

Riyad Alshammari, *Student Member, IEEE* and A. Nur Zincir-Heywood, *Member, IEEE*

Abstract— **The classification of Encrypted Traffic, namely Skype, from network traffic represents a particularly challenging problem. Solutions should ideally be both simple – therefore efficient to deploy – and accurate. Recent advances to team-based Genetic Programming provide the opportunity to decompose the original problem into a subset of classifiers with non-overlapping behaviors. Thus, in this work we have investigated the identification of Skype encrypted traffic using Symbiotic Bid-Based (SBB) paradigm of team based Genetic Programming (GP) found on flow features without using IP addresses, port numbers and payload data. Evaluation of SBB-GP against C4.5 and AdaBoost – representing current best practice – indicates that SBB-GP solutions are capable of providing simpler solutions in terms number of features used and the complexity of the solution/model without sacrificing accuracy.**

I. Introduction

The accurate identification of network traffic in relation to application type represents a challenging decision making activity. This activity is very important for network management tasks such as managing bandwidth and ensuring quality of service objectives for critical applications. One method to classifying network traffic is to inspect the payload of every packet, i.e. deep packet inspection. However, deep packet inspection becomes irrelevant when the payload is encrypted. Another method to classifying applications is using well-known Transmission Control Protocol (TCP)/User Datagram Protocol (UDP) port numbers. But, this approach becomes increasingly inaccurate when applications use dynamically allocated port numbers. Thus, different research groups have employed many machine learning techniques such as Hidden Markov models, Naïve Bayesian models, AdaBoost, RIPPER, or Decision Trees to this problem [1], [2], [3], [4], [5], [6], [7], [8], [9]. However, in general all these efforts show that even though it is easier to apply such techniques to well known public domain applications, more work is necessary to identify encrypted applications accurately.

In this work, we focus on Skype traffic as a case study of encrypted traffic classification. Skype is a proprietary Peer to Peer (P2P) Voice over IP (VoIP) application. Skype covers a collection of different encrypted behavior, which makes it difficult to distinguish from non-Skype traffic. Thus, in this work, we aim to develop a model that distinguishes Skype from non-Skype traffic without using IP addresses, port numbers or payload information. We believe that this will not only enable our model to generalize well from one network to another but will potentially enable us to apply such an approach to the classification of other encrypted applications, too. To achieve this, we are going to examine the utility of SBB-GP against the two most frequently preferred machine learning approaches for network traffic classification, specifically C4.5 and AdaBoost. The specific form of team-based GP employed to this real life application takes the form of the SBB paradigm[1] where this is known to be computationally efficient (care of a competitive coevolutionary formulation of active learning) and does not require the a priori specification of the number of team members (care of a bid-based model of cooperation) [10], [11].

The rest of this paper is organized as follows. Related work is discussed in Section II and an overview of Skype is given in Section III. Section IV presents the machine learning algorithms employed whereas Section V details the data sets and features. The experimental results are presented in Section VI. Finally, conclusions are drawn and future work is discussed in Section VII.

II. Related Work

Skype analysis has become popular in the last few years, in part due to the combination of the encrypted operation and dynamic nature of the port assignment making traditional methods of traffic identification redundant. Baset et al. present an analysis of the Skype behavior such as login, Network Address Translation (NAT) and firewall avoidance, and call setting up under three different network arrangements [12]. Suh et al. concentrate on the classification of relayed traffic and monitored Skype traffic as an application using relay nodes [13]. Relay node is part of the decentralized Skype network that can ease the routing of Skype traffic to bypass NATs and firewalls. They used several features such as inter-arrival time, bytes size ratio and maximum cross correlation between two relayed bursts of packets to detect Skype relay traffic. Their results show the technique is reliable in recognizing relayed Skype sessions but it might not be appropriate to classify all Skype VoIP traffic. Ehlert et al. find signatures containing different distinctiveness of Skype signaling traffic such as port usage, packet size and payload content [14]. However, they have employed deep packet analysis on the payload, which makes their signatures less generalizable given that the payload is encrypted. Bonfiglio et al. have adopted two different and complementary techniques to make a classification either when the flow ends or when enough packets are captured to detect Skype traffic [15]. These techniques were Chi-Square tests and Naïve Bayesian classifiers. However they also employed a

R. Alshammari is with Dalhousie University, Faculty of Computer Science, Halifax, NS B3H 1W5, Canada. (e-mail: riyad@cs.dal.ca)

A. N. Zincir-Heywood is with Dalhousie University, Faculty of Computer Science, Halifax, NS B3H 1W5, Canada (phone: 902-4943157; fax: 902-4921517;e-mail: zincir@cs.dal.ca)

[1]Source code available from http://www.cs.dal.ca/˜mheywood/Code/SBB /SCM.9.r20081212.tar.gz

payload based classification scheme. Perenyi et al. proposed Skype identification algorithm depend on observable parts such as speech flows, timing of voice packet and candidate hosts found [16]. Their technique works on offline data and depends on detecting Skype hosts found on traditional IP and port number based identification and signaling flow information. The signaling flow information relies on a number of packets, their direction, size and time but no packet payload is required. Bonfiglio et al. introduced three approaches to classify Skype traffic [17]. The first approach is to classify Skype client traffic based on Pearson's Chi-Square test using information revealed from the message content randomness (e.g. the FIN and ID fields) introduced by the cypher and the header format. Their second approach is to classify Skype VoIP traffic made from Naïve Bayesian Classifier using packet arrival rate and packet length. Their third apporach is based on deep packet inspection combined with per-host analysis to correctly classify Skype traffic.

In contrast to the previous work, we specifically focus on encrypted tunnel identification, where identifying Skype encrypted tunnels is taken as a case study in this work. However, our proposed system can potentially be applied to any encrypted application since it applies flow based features without using the IP addresses, port numbers and payload data. It should be noted here, usage of port numbers, IP addresses and payload based features make the solutions less re-usable. In return, this causes a new feature set to be chosen for each application that needs to be identified. Whereas, we aim to use a generic feature set and let the machine learning algorithm employed to identify the subsets of it for classifying any given application. Recently, we have evaluated AdaBoost, Support Vector Machine, Naïve Bayesian, RIPPER and C4.5 using flow based features, where IP addresses, source/destination ports and payload information are not employed to classify encrypted traffic [3]. Results indicate the C4.5 based approach outperforms other algorithms on the data sets employed. However, in that work, we have not tested the Skype classifier in terms of how well it generalizes on different network data sets. Furthermore, we have also developed and compared SBB-GP based classifier against C4.5 [18] on SSH traffic classification. In that work, results show that SBB-GP based classifier was quiet competitive with the C4.5 based classifier. Thus in this work, we aim to perform a investigation of SBB-GP based classifier on classification of Skype encrypted tunnels as well as explore its robustness in terms of evaluating it on different network traces from different institutions.

III. Summary of Skype Application

Skype [19] is a very popular P2P VoIP client developed in 2002 by the developers of KaZaa that allows its users to communicate through voice calls, audio conferencing and text messages. Skype protocols are proprietary and an extensive use of cryptography is implemented by the Skype creators. Moreover, Skype employs a number of methods to circumvent NAT and firewall restrictions [12] which increase the difficulty of identifying it. Skype is based on P2P architecture except users authentication, which is performed based on a central architecture. Skype uses the TCP or the UDP protocols at the transport layer to provide its services. For network communication, Skype mostly prefers the UDP protocol. A more detailed description of Skype protocol can be found in [12].

IV. Classifier Methodologies

Given the general success of C4.5 and AdaBoost in previous studies [2], [3], [4], [5], [6], [7], [9], [20], we employ both models during this study in order to establish a performance baseline. A more detailed explanation of C4.5 and AdaBoost algorithms can be found in [21] whereas a more detailed explanation of SBB-GP can be found in [10]. The following will summarize the C4.5, AdaBoost and SBB-GP algorithms.

A. C4.5

C4.5 is a decision tree based classification algorithm. A decision tree is a hierarchical data structure for implementing a divide-and-conquer strategy of attribute based model building. It is an efficient non-parametric method applicable both to classification and regression. Non-parametric models divide the input space into local regions defined by a distance metric. In a decision tree, the local region is identified in a sequence of recursive splits in smaller number of steps. A decision tree is composed of internal decision nodes and terminal leaves. Each node *m* implements a test function *fm(x)* with discrete outcomes labeling the branches. This process starts at the root and is repeated until a leaf node is encountered. The value of a leaf constitutes the output. In the case of a decision tree for classification, the goodness of a split is quantified by an impurity measure, typically entropy based. Naturally, if the split is not 'pure', then the instances should be split to decrease impurity, and there are multiple possible attributes on which a split can be performed. Such a scheme is locally optimal, hence has no guarantee on finding the smallest decision tree.

B. AdaBoost

AdaBoost, Adaptive Boosting, is a meta-learning algorithm, which means that a strong classifier is built from a linear combination of weak (simple) classifiers. It incrementally constructs a complex classifier by overlapping the performance of possibly hundreds of simple classifiers using a voting scheme. These simple classifiers are called decision stumps. They examine the feature set and return a decision tree with two leaves. The leaves of the tree are used for binary classification and the root node evaluates the value of only one feature. Thus, each decision stump will return either +1 if the object is in class, or -1 if it is out class. AdaBoost is simple to implement and known to work well on very large sets of features by selecting the features required for good classification.

C. Overview of SBB-GP

1) Overview: The canonical framework for applying model based cases of evolution – such as GP – to the supervised learning domain of classification requires an individual to map exemplars from an attribute space to a class label space. An individual's program expresses the mapping. However, this is not the case under the SBB framework [22]. Instead the task is divided into two components: (1) deciding *which* exemplars to label, or the *bid*, and (2) suggesting class label, or the *action*. In the case of the individual's action, the assumption is made that an individual will always be associated with the same action (class label). Thus at initialization, a problem with C classes results in $\frac{PopSize}{C}$ of the individuals in the population being pre-assigned to each class. The assignment is defined by assigning a scalar a to each individual at initialization. Scalars are selected with uniform probability from the set $\{1, ..., C\}$. The actions are *not adapted* during evolution. Conversely, the task of deciding which subset of exemplars to label is expressed in terms of a bid. The individual with the maximum (winning) bid suggests its pre-assigned action as the class label. During training, individuals are rewarded for bidding high only on exemplars whose class label matches their action.

The most recent form of the bid-based framework makes extensive use of coevolution [10], with a total of three populations involved: a population of points, a population of learners, and a population of teams (Fig. 1). Individuals comprising a team are specified by the team population, thus establishing a symbiotic relationship with the learner population. Only the subset of individuals indexed by an individual in the team population compete to bid against each other on training exemplars. The use of a symbiotic relation between teams and learners makes the credit assignment process more transparent than in the case of a population wide competition between bids (as used in the earlier variant of the model [22]). Thus, variation operators may now be defined for independently investigating team composition (team population) and bidding strategy (learner population). The third population provides the mechanism for scaling evolution to large data sets. In particular the interaction between team and point population is formulated in terms of a competitive coevolutionary relation [23]. As such, the point population indexes a subset of the training data set under an active learning model (i.e. the subset indexed varies as classifier performance improves). Biases are enforced to ensure equal sampling of each class, irrespective of their original exemplar class distribution [24], whereas the concept of Pareto competitive coevolution is used to retain points of most relevance to the competitive coevolution of teams.

2) SBB Algorithm: The SBB model of evolution generates P_{gap} new exemplar indexes in the point population and M_{gap} new teams in the team population at each generation. Specifically, individuals in the point population take the form of indexes to the training data and are generated stochastically (subject to the aforementioned class balancing heuristic). New teams are created through variation operators applied

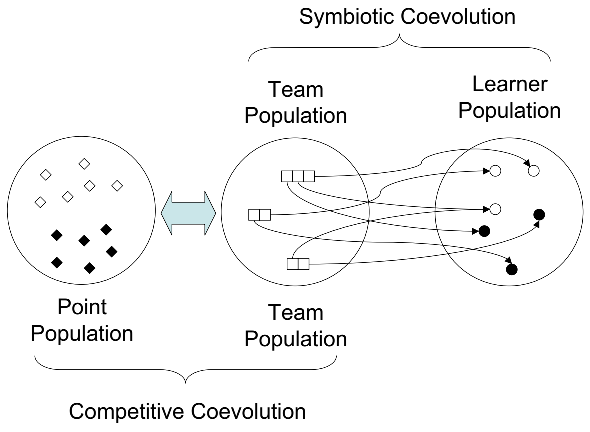

Fig. 1: Architecture of Symbolic Bid-based GP: Point to team populations are competitive, Team to learner populations are symbiotic (cooperative).

to the current team population. Fitness evaluation evaluates all teams against all points with $(P_{size} - P_{gap})$ points and $(M_{size} - M_{gap})$ teams appearing in the next generation. Pareto competitive coevolution ranks the performance of teams in terms of a vector of outcomes, thus the Pareto non-dominated teams are ranked the highest [23]. Likewise, the points supporting the identification of non-dominated individuals (distinctions) are also retained. In addition, use is made of competitive fitness sharing in order to bias survival in favor of teams that exhibit uniqueness in the non-dominated set (Pareto front).

Evaluation of team m_i on a training exemplar defined by point population member p_k results in the construction of an outcome matrix $G(m_i, p_k)$ in which unity implies a correctly classified exemplar, and zero an incorrectly classified exemplar. A distinction for point p_k is defined as

$$\begin{cases} 1 & \text{if } G(m_i, p_k) > G(m_j, p_k) \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

where unity implies that point p_k 'distinguishes' between team m_i and m_j, and the result of all such pairwise team comparisons defines the objectives for the point. The ensuing Pareto competitive coevolutionary process identifies the non-dominated teams and points supporting their identification.

Denoting the non-dominated and dominated points as $F(P)$ and $D(P)$ respectively, the SBB framework notes that as long as $F(P)$ contains less than $(P_{size} - P_{gap})$ points, all the points from $F(P)$ are copied into the next generation. On the other hand, if $F(P)$ contains more points than are allowed to survive, then the following fitness sharing heuristic is imposed to rank the collection of non-dominated points [25],

$$\sum_i \frac{d_k[i]}{1 + N_i} \tag{2}$$

where $d_k[i]$ is the *i*th entry of the distinction vector for p_k; and N_i is the sum of the *i th* entries over the distinction vectors across all points in $F(P)$ i.e., the number of points making the same distinction. Thus, points making the same distinction are weighted less than points making unique distinctions.

An analogous process is repeated for the case of team selection, with $(M_{size} - M_{gap})$ individuals copied into the next generation. Naturally, under the condition where the (team) non-dominated set exceeds this number, the fitness sharing ranking employs $F(M)$ and $D(M)$ in place of $F(P)$ and $D(P)$ respectively. The resulting process of fitness sharing under a Pareto model of competitive coevolution has been shown to be effective at promoting solutions in which multiple models cooperate to decompose the original $|C|$ class problem into a set of non-overlapping behaviors [10], [22].

Finally, the learner population of individuals expressing specific bidding strategies employs a linear representation. Bid values are standardized to the unit interval through the use of a sigmoid function, or $bid(y) = (1+\exp -y)^{-1}$, where y is the real valued result of program execution on the current exemplar. Variation operators take the form of instruction add, delete, swap and mutate, applied with independent likelihoods, under a uniform probability of selection. When an individual is no longer indexed by the team population it becomes extinct and deleted from the learner population. Conversely, during evaluation of the team population, exactly M_{gap} children are created pairwise care of team based crossover. Learners that are common to both child teams are considered to be the candidates for retention. Learners not common to the child teams are subject to stochastic deletion or modification, with corresponding tests for deletion/insertion at the learner population. The instruction set follows from that assumed in [10] and consists of eight opcodes ($\{cos, exp, log, +, \times, -, \div, \%\}$) operating on up to 8 registers, as per a linear GP representation.

V. Evaluation Methodology

As discussed earlier, in this work, the preferred models of classifications from Section II (C4.5 and AdaBoost) will be compared against SBB-GP using the flow based features for Skype encrypted tunnel classification.

A. Data Collection

In our experiments, the performance of the different machine learning algorithms is established on Dalhousie University traces and the Italy traces. Dalhousie traces were captured on the Dalhousie University Campus network by the University Computing and Information Services Centre (UCIS) in January 2007. Dalhousie is one of the biggest universities in the Atlantic region of Canada. There are more than 15000 students and 3300 faculty and staff. The UCIS is responsible for all the networking on the campus which includes more than 250 servers and 5000 computers. Moreover, the wireless network is enabled on the campus where thousands of users (students and staff) are connected daily. Dalhousie network is connected to the Internet via a full-duplex T1 fiber link. Full-duplex traffic on this connection was captured for 8 hours. Given the privacy related issues, data is filtered to scramble the IP addresses and each packet is further truncated to the end of the IP header so that all payload is excluded. Moreover, the checksums are set to zero since they could conceivably leak information from short packets. However, any information regarding size of the packet is left intact. Moreover, Dalhousie traces (University) are labeled by UCIS by a commercial classification tool called PacketShaper, which is a deep packet analyzer [26]. PacketShaper uses Layer 7 filters (L7filter) to classify the applications [27]. On the other hand, the Italy data set consists of 96 hours of Skype Traffic over TCP and UDP protocols [28]. The data set is captured on the main link at the Politecnico di Torino University campus. TCP Statistic and Analysis Tool (Tstat) and the traffic classification method employed are described in [17]. As described in section II, the creators of this data set classified Skype traffic based on deep packet inspection and per-host analysis. Furthermore, their second approach results in a classifier, which depends on Pearsons Chi-Square test using information from the message content (payload) and their third approach results in a classifier, which depends on Naïve Bayesian classification technique using packet arrival rate and packet length to classify the Skype traffic. The third approach is closer to our method however we have shown that C4.5 work much better than Naïve Bayesian [3]. In this work, we employed their two Skype End-to-End call traces. The first trace, which is captured over UDP, consists of voice only calls as well as voice plus video calls. The second trace is captured over TCP and consists of SkypeOut traffic. Brief statistics on the traffic data collected are given in Table I.

TABLE I: An overview of network traces employed

	University	**Italy**
Total Packets	337,041,778	41,985,540
MBytes	213,562	8,147
% of TCP packets	86.51%	5.62%
% of TCP bytes	91.03%	3.75%
% of UDP packets	13.33%	94.38%
% of UDP bytes	8.95%	96.25%
% of Other packets	0.16%	0.0%
% of Other bytes	0.02%	0.0%
Total SSH Flows	19,384	N/A
Total non-SSH Flows	44,210,314	N/A
Total Skype Flows	8,664,137	389070
Total non-Skype Flows	35,565,561	0

Naturally, these traffic traces represent large data sets from a machine learning perspective. Thus, subset sampling is used to decouple the overall exemplar count from the subset over which training is conducted. Indeed when subsampling algorithms impose a simple class balance heuristic, performance of the resulting models is frequently better than when trained over all exemplars. Specifically, the balanced subset sampling heuristic results in performance correlated with maximization of the AUC (Area Under the Receiver Operating characteristic (ROC) Curve) statistic [29]. Thus, we performed subset sampling to limit the memory and CPU time required for training. In this case, Skype Dal Training

TABLE II: Flow based features employed

	Feature Name	Abbreviation
1	Protocol	proto
2	Duration of the flow	Duration
3	# Packets in forward direction	fpackets
4	# Bytes in forward direction	fbytes
5	# Packets in backward direction	bpackts
6	# Bytes in backward direction	bbytes
7	Min forward inter-arrival time	minfiat
8	Mean forward inter-arrival time	meanfiat
9	Max forward inter-arrival time	maxfiat
10	Std deviation of forward inter-arrival times	stdfiat
11	Min backward inter-arrival time	minbiat
12	Mean backward inter-arrival time	meanbiat
13	Max backward inter-arrival time	maxbiat
14	Std deviation of backward inter-arrival times	stdbiat
15	Min forward packet length	minfpkt
16	Mean forward packet length	meanfpkt
17	Max forward packet length	maxfpkt
18	Std deviation of forward packet length	stdfpkt
19	Min backward packet length	minbpkt
20	Mean backward packet length	meanbpkt
21	Max backward packet length	maxbpkt
22	Std deviation of backward packet length	stdbpkt

Sample is generated by sampling randomly selected (uniform probability) flows from different classes (FTP, SSH, MAIL, DNS, HTTP, HTTPS and Random UDP). The applications in the "Random UDP" class includes random UDP flow instances. In total, Skype Dal Sample consists of 600000 balanced flows (skype flows vs non-skype flows).

B. Feature Selection

Network traffic is represented using flow-based features. In this case, each network flow is described by a set of statistical features, Table II. Here, a feature is a descriptive statistic that can be calculated from one or more packets. To this end, NetMate [30] is employed to process packets, generate flows and compute feature values. Flows are bidirectional and the first packet seen by the tool determines the forward direction. Moreover, flows are of limited duration. UDP flows are terminated by a flow timeout. TCP flows are terminated upon proper connection teardown or by a flow timeout, whichever occurs first. The flow time out value employed in this work is 600 seconds [31]. The flows are defined by the features, we extract a similar set of features as in [2], form the input vector for the machine learning model, which then provides a label {Skype, non-Skype} for each flow. As discussed earlier, features such as IP addresses, source/destination port numbers and payload are excluded from the feature set to ensure that the results are not dependent on such biased features.

VI. Empirical Evaluation

In traffic classification, two metrics are typically used in order to quantify the performance of the classifier: Detection Rate (DR) and False Positive Rate (FP). In this case DR will reflect the number of Skype flows correctly classified and is calculated using $DR = \frac{TP}{TP+FN}$; whereas FP rate will reflect the number of Non-Skype flows incorrectly classified as Skype and is calculated using $FPR = \frac{FP}{FP+TN}$. Naturally, a high DR rate and a low FP rate are the most desirable outcomes. False Negative, FN, implies that Skype traffic is classified as non-Skype traffic, FP, False Positive, implies that non-Skype traffic is classified as Skype traffic. All three candidate classifiers are trained on the training data using fifty runs to generate 50 different models for each run. Weka [32] is employed with default parameters to run C4.5 and AdaBoost. Fifty runs of the C4.5 algorithm are performed using different confidence factors to generate different models for C4.5 and fifty runs of the AdaBoost algorithm are performed using different weight thresholds to generate different models for AdaBoost. The SBB-GP classifier default parameters are summarized in Table III. Fifty runs of the SBB-GP algorithm are performed using different population initializations to generate different models.

TABLE III: SBB-GP parameters

Parameter	Description	Value
P_{size}	Point population size.	90
M_{size}	Team population size.	90
t_{max}	Number of generations.	30000
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

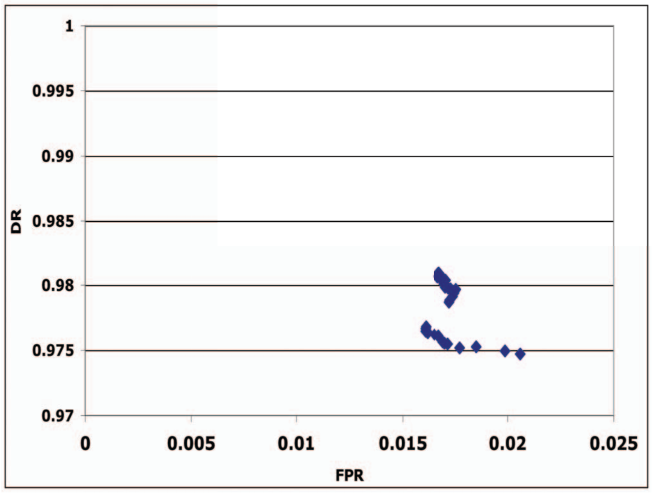

Fig. 2: ROC Curve plot for the C4.5 for training performance using Flow based Feature set for Skype (DR versus FPR)

Fig. 2, 3 and 4 summarize solutions; there are ten that are non-dominated for GP, five that are non-dominated for AdaBoost and three that is non-dominated for C4.5. These non-dominated solutions for GP, AdaBoost and C4.5 are then evaluated on the test data sets and compared based on their performance to detect Skype encrypted tunnels with high DR and low FPR.

TABLE IV: Best results out of 50 runs for each Classifier on training and testing data sets.

	C4.5		AdaBoost		GP	
	DR	**FPR**	**DR**	**FPR**	**DR**	**FPR**
Training Sample (subset of university)						
Non-Skype	0.98	0.02	0.95	0.17	0.91	0.13
Skype	0.98	0.02	0.83	0.05	0.87	0.09
University Traces (test)						
Non-Skype	0.92	0.02	0.96	0.15	0.94	0.08
Skype	0.98	0.08	0.85	0.04	0.92	0.06
Italy Traces (test)						
Non-Skype	0.0	0.40	0.0	0.89	0.0	0.15
Skype	0.60	0.0	0.11	0.0	0.85	0.0

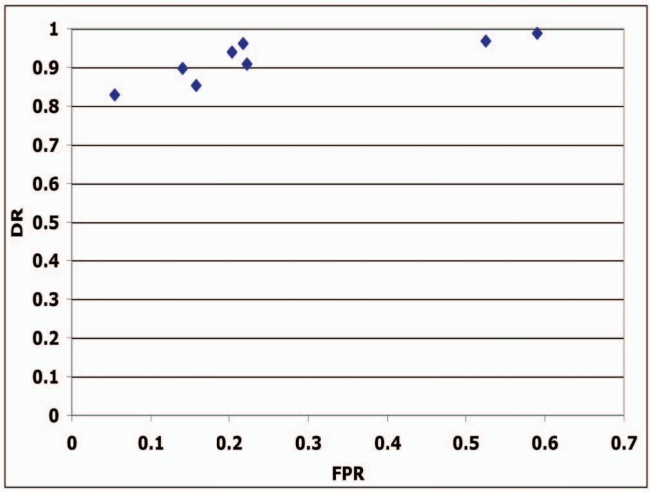

Fig. 3: ROC Curve plot for the AdaBoost for training performance using Flow based Feature set for Skype (DR versus FPR)

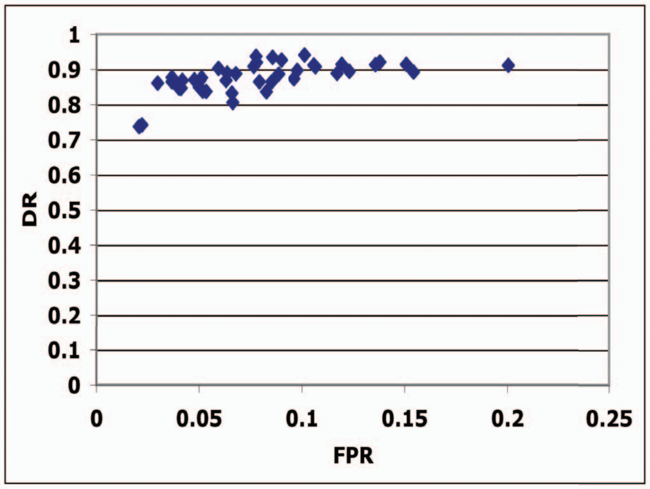

Fig. 4: ROC Curve plot for the SBB-GP for training performance using Flow based Feature set for Skype (DR versus FPR)

TABLE V: Standard Deviation of Results on the training data

	C4.5		AdaBoost		GP	
	DR	**FPR**	**DR**	**FPR**	**DR**	**FPR**
Training Sample (subset of university) x 50						
non-Skype	0.001	0.002	0.17	0.04	0.03	0.05
Skype	0.002	0.001	0.04	0.17	0.05	0.03

A. Results

Results are summarized in terms of both accuracy and model complexity. Accuracy provides us the performance of the learning models in terms of DR and FP rate. On the other hand, model complexity is also very important if such a classification system were to be used in real-time or near real-time environments.

In these experiments, we first trained each classifier on our training data set (which is sub-sampled from University traces) using the same feature set. Then, we tested each trained model (C4.5, AdaBoost and SBB-GP) on two different test data sets, namely, University traces (Dalhousie traces minus the training partition) and the Italy traces. Results given in Table IV shows that C4.5 based classification approach is much better than other machine learning algorithms employed in identifying the Skype traffic. Moreover, in the case of C4.5, much lower variance (Table V) implies that the corresponding solutions generalize to the wider case, implicit in the test results. In this case, C4.5 based system can correctly classify ≈98% of the instances with 8% FPR on the University Test trace. On the same data set, SBB-GP based system closely follows the performance of C4.5 based system (with ≈92% DR and ≈6% FPR whereas the AdaBoost based system performs the poorest of the three. However, introducing the entirely independent test set – Italy – indicated that C4.5 and AdaBoost had over-learnt the properties implicit in the training partition. Furthermore, SBB-GP was observed to provide the best case performance under the independent test partition (≈85% DR). The SBB-GP classifier was the most consistent performer across all test and training conditions. It should be noted here, the FP rate for Skype and DR for non-Skype are zero in the Italy traces because this network trace contains only Skype traffic.

These results demonstrate that it is not only possible to identify Skype encrypted tunnels without using IP addresses, port numbers and payload information but also it is possible to have a generic attribute set that can be employed to identify encrypted tunnels such as Skype (in our previous work, we have employed the same feature set for SSH tunnel identification [1], [2], [3], [4], [5], [20]). Moreover, these results show that the classification based system trained on data from one network can be employed to run on a different network without new training. Our results show that, the SBB-GP solution seems to generalize well from one network data to another and is therefore robust. In all cases, the approach adopted to attribute selection was to include as wide a set as possible and let the 'embedded' properties of the various learning algorithms establish which subset of attributes to actually employ. Given this capability, we are now in a position to review the attributes selected by each model, where this is readily achieved class-wise in the case of both C4.5 and SBB-GP. The summary for AdaBoost is not as straight-forward and will therefore be limited to the total set of attributes utilized, independent of class.

Table VI summarizes these findings. SBB-GP clearly uses a lower total count of attributes relative to C4.5, and a lower count of attributes for Skype detection. Conversely, C4.5 uses the largest set of attributes as a whole or class-wise. Each classifier also identifies attributes unique to their solution. For example, SBB-GP is the only model to choose the Protocol attribute since Skype application runs on both TCP and UDP protocols; while C4.5 utilizes the packet size and inter-arrival time attributes. Also of interest is the low level of overlap in shared attributes, with only 2 of 3 attributes shared between C4.5 and SBB-GP and 1 of 3 attributes shared between AdaBoost and SBB-GP under Skype detection.

In terms of solution complexity for the best model/solution, we note that AdaBoost generates 7 rules for Skype traffic and 10 rules for Non-Skype traffic; whereas C4.5 employs 101 rules for Skype classification and 104 rules to classify non-Skype traffic. Conversely, SBB-GP uses 2 individuals for Skype classification and 7 for non-Skype. We also note that instruction counts of the SBB-GP for the Skype classifying individuals was 3 and 13 instructions respectively; whereas individuals engaged in classifying non-Skype utilized 3, 4 (three off), 5 (two off) and 13 instructions. These results show the simplicity of GP solutions does not appear to be traded off for classifier complexity in the identification of Skype tunnels. In summary, machine learning algorithms such as AdaBoost, SBB-GP and C4.5 select the most appropriate attributes (among the set given) to build their classifier model. We used this information to distinguish 3 flow attributes from 22 flow attributes that are used in our experiments for identifying Skype tunnels using SBB-GP.

VII. Conclusions

Previous research on traffic classification indicated that embedded paradigms such as C4.5 and AdaBoost provided better classification performance than methods without this capacity. In this work, we are able to take this concept further by introducing a model for learning problem decomposition in classification that explicitly associates independent subsets of exemplars with models identified during training. Such a team-based model of learning is not a weak learner, thus solutions take the form of a small number of simple programs with an explicitly non-overlapping behavioral trait and independent attribute subspaces. Such a methodology is able to provide solutions that are competitive under independent training and test data sets, while returning solutions that are potentially capable of higher throughputs of data than provided under a single 'monolithic' classifier per binary classification model (such as C4.5). Thus, solutions from C4.5 utilized deep decision trees, and are expensive to evaluate from a throughput perspective. Solutions located by AdaBoost were not effective under the network from which training data was collected. Moreover, the weak learner methodology implicit in AdaBoost makes it much more difficult to associate attributes with class, reducing transparency of the resulting solution.

Future work will follow similar lines to perform more tests on different and/or larger data sets in order to continue to test the robustness of the classifiers as well as exploring the appropriateness of other machine learning algorithms. Evaluation under other encrypted applications as well as exploring the possibilities for integrating our approach with approaches employing host based behavior are also of interest.

Acknowledgments

This work was in part supported by MITACS, NSERC and the CFI new opportunities program. Our thanks to John Sherwood, David Green and Dalhousie UCIS team for providing us the anonymized University traffic traces. All research was conducted at the Dalhousie University, Faculty of Computer Science NIMS Laboratory, http://www.cs.dal.ca/projectx.

References

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

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

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

[4] R. Alshammari, A. N. Zincir-Heywood, and A. A. Farrag. Performance comparison of four rule sets: An example for encrypted traffic classification. *Privacy, Security, Trust and the Management of e-Business, World Congress on*, 0:21–28, 2009.

[5] R. Alshammari and N. Zincir-Heywood. Generalization of signatures for ssh encrypted traffic identification. In *Computational Intelligence in Cyber Security, 2009. CICS '09. IEEE Symposium on*, pages 167–174, 30 2009-April 2 2009.

[6] J. Early, C. Brodley, and C. Rosenberg. Behavioral authentication of server flows. In *Proceedings of the 19th Annual Computer Security Applications Conference*, pages 46–55, 2003.

[7] P. Haffner, S. Sen, O. Spatscheck, and D. Wang. ACAS: automated construction of application signatures. In *MineNet '05: Proceeding of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 197–202, New York, NY, USA, 2005. ACM Press.

TABLE VI: Features used by All Classifiers for in-class (Skype) and out-class (non-Skype)

	C4.5		**GP**		**AdaBoost**	
	in-class	**out-class**	**in-class**	**out-class**	**in-class**	**out-class**
1	min_fpktl	min_fpktl	min_bpktl	min_bpktl	min_fpktl	min_fpktl
2	min_bpktl	min_bpktl	proto	proto	max_bpktl	max_bpktl
3	total_fvolume	total_fvolume	total_fvolume	min_fiat	min_bpktl	min_bpktl
4	max_bpktl	duration		mean_fiat	total_fpackets	total_fpackets
5	max_fiat	std_bpktl		std_fiat	duration	duration
6	std_bpktl	max_fiat		duration		
7	total_fpackets	max_bpktl		mean_fpktl		
8	duration	min_biat				
9	min_biat	min_fiat				
10	total_bpackets	std_fpktl				
11	max_fpktl	mean_bpktl				
12	std_fpktl	max_biat				
13	min_fiat	mean_biat				
14	mean_bpktl	max_fpktl				
15	total_bvolume	mean_fpktl				
16	mean_biat	total_bvolume				
17	mean_fpktl	std_fiat				
18	std_fiat	total_fpackets				
19	mean_fiat	mean_fiat				
20		total_bpackets				

[8] A. W. Moore and D. Zuev. Internet traffic classification using bayesian analysis techniques. In *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 50–60, New York, NY, USA, 2005. ACM Press.

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

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

[11] A. McIntyre and M. Heywood. Cooperative problem decomposition in Pareto competitive classifier models of coevolution. In *European Conference on Genetic Programming*, volume 4971 of *Lecture Notes in Computer Science*, pages 289–300, 2008.

[12] S. A. Baset and H. G. Schulzrinne. An analysis of the skype peer-to-peer internet telephony protocol. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–11, 2006.

[13] D. K. Suh, D. R. Figueiredo, J. Kurose, and D. Towsley. Characterizing and detecting relayed traffic: A case study using skype. *in INFOCOM 06: Proceedings of the 25th IEEE International Conference on Computer Communications*, Apr 2006.

[14] S. Ehlert, S. Petgang, T. Magedanz, and D. Sisalem. Analysis and signature of skype voip session traffic. *in CIIT 2006: 4th IASTED International Conference on Communications, Internet, and Information Technology*, page 8389, Nov/Dec 2006.

[15] D. Bonfiglio, M. Mellia, M. Meo, and D. Rossi. Detailed analysis of skype traffic. *Multimedia, IEEE Transactions on*, 11(1):117–127, Jan. 2009.

[16] M. Perényi, A. Gefferth, T. D. Dang, and S. Molnár. Skype traffic identification. In *GLOBECOM*, pages 399–404. IEEE, 2007.

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

[18] R. Alshammari, P. I. Lichodzijewski, M. Heywood, and A. N. Zincir-Heywood. Classifying ssh encrypted traffic with minimum packet header features using genetic programming. In *GECCO '09: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, pages 2539–2546, New York, NY, USA, 2009. ACM.

[19] Skype. http://www.skype.com/useskype/.

[20] R. Alshammari and A. Zincir-Heywood. A preliminary performance comparison of two feature sets for encrypted traffic classification. *Proceedings of the International Workshop on Computational Intelligence in Security for Information Systems CISIS'08*, pages 203–210, 2009.

[21] E. Alpaydin. *Introduction to Machine Learning*. MIT Press, 2004.

[22] P. Lichodzijewski and M. I. Heywood. Coevolutionary bid-based Genetic Programming for problem decomposition in classification. *Genetic Programming and Evolvable Machines*, 9(4):331–365, 2008.

[23] E. de Jong. A monotonic archive for pareto-coevolution. *Evolutionary Computation*, 15(1):61–93, 2007.

[24] J. Doucette and M. Heywood. Gp Classification under Imbalanced Data Sets: Active Sub-sampling and AUC Approximation. In *European Conference on Genetic Programming*, volume 4971 of *Lecture Notes in Computer Science*, pages 266–277, 2008.

[25] C. D. Rosin and R. K. Belew. New methods for competitive coevolution. *Evol. Comput.*, 5(1):1–29, 1997.

[26] PacketShaper, last accessed March, 2008. http://www.packeteer.com/products/packetshaper/.

[27] l7-filter, last accessed March, 2008. http://l7-filter.sourceforge.net/.

[28] Skype traces, last accessed August, 2009. http://tstat.tlc.polito.it/traces-skype.shtml.

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

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

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

[32] WEKA software. http://www.cs.waikato.ac.nz/ml/weka/.