A Statistical-Feature-Based Approach to Internet Traffic Classification Using Machine Learning

Shijun Huang Kai Chen Chao Liu
School of Information Security Engineering
Shanghai Jiao Tong University
Shanghai, China
sj_yeye@sjtu.edu.cn kchen@sjtu.edu.cn

Alei Liang
School of Software
Shanghai Jiao Tong University
Shanghai, China
liangalei@cs.sjtu.edu.cn

Haibing Guan
Department of Computer Science and Engineering
Shanghai Jiao Tong University
Shanghai, China
hbguan@sjtu.edu.cn

Abstract—**This Internet traffic classification using Machine Learning is an emerging research field since 1990's, and now it is widely used in numerous network activities. The classification technique focuses on modeling attributes and features of data flows to accomplish the identification of applications. In the paper we design and implement the classification model based on header-derived flow statistical features. Compared with the traditional methods, the model designed here, which is totally insensitive to port numbers and contents of payload on application level, overcomes difficulty in operation caused by unreliable port numbers and complexity of payload interpretation. Rather than relatively complex ML algorithms or even in mixture, supervised k-Nearest Neighbor estimator is adopted for the sake of computational efficiency, along with the effective and easy-to-calculate statistical features selected according to the operational background. Our results indicate that about 90% accuracy on per-flow classification can be achieved, which is a vast improvement over traditional techniques that achieve 50-70%.**

Keywords-traffic classification; Machine Learning; flow features; k-Nearest Neighbor

I. INTRODUCTION

Internet traffic classification is of primary importance to various network topics, from fields like security monitoring to accounting, and from Intrusion Detection Systems (IDS) components [1, 2] to Quality of Service (QoS) architectures [3]. However, along with the growing popularity of services like peer-2-peer (P2P) applications and emerging encryption techniques, traditional techniques of Internet traffic classification that relied on well-known TCP or UDP port numbers, or interpreting the contents of packet payloads, cannot meet the needs of classification accuracy [4], and are restricted faced with encrypted contents (including TCP or UDP port numbers), or privacy regulation as well.

Our work uses supervised Machine Learning (ML) to categorize Internet traffic. We hand-assign common network applications that we are interested in to several pre-defined categories. Well-selected statistical features are extracted and calculated upon flows (e.g., flow data bytes, initial window size) to constitute the unique signature (feature set) of each flow. Numbers of signatures for known categories are used to train the classifier, which is the progress mapping feature sets to specific category of applications. Given signatures for unknown categories when testing, we use k-Nearest Neighbor (KNN) estimator to accomplish the classification.

The whole process of classification in our design aims at higher computational efficiency in both periods of calculating statistical features and applying classification algorithm, with high accuracy of classification being maintained all the same, which is worth mentioning.

The rest of this paper is organized as follows. Section II reviews the related work in the field of traffic classification. Section III outlines some setups for operational deployment. Section IV expounds the core techniques in the classification process, from feature selection to classification algorithm. Section V illustrates and discusses the classification results. Section VI concludes the paper with some final remarks and possible future work.

II. RELATED WORK

The most common technique for identification of network applications relies on the use of well-know TCP or UDP ports: port numbers are obtained through inspection of the headers of transport layer, and the certain application is then inferred by looking them up in the Internet Assigned Numbers Authority (IANA)'s list of registered ports [5]. Such a simple method, that using no more information than port numbers, achieves relatively low accuracy in classification because emerging services (e.g., peer-2-peer classification) avoid using well-known ports. (Madhukar and Williamson [6] showed 30-70%

9781-4244-3941-6/09/$25.00 ©2009 IEEE

of Internet traffic flows they investigated cannot be identified using port-based analysis.) Also, server ports are dynamically allocated as needed in some special cases (e.g., FTP service connected in passive mode).

For the purpose of getting rid of reliance on the semantics of port numbers, payload based technique utilizing session reconstruction and payload interpretation is adopted for traffic classification to achieve higher accuracy. Yet, analysis of contents of packet payloads brings in such problems as complexity and inefficiency. Moreover, the technique becomes useless when faced with payload encryption or privacy regulation.

With the underlying assumption that traffic generated by different categories of applications owns unique flow-based properties, statistical-feature-based technique for traffic classification overcomes the limitations mentioned above.

Roughan et al. [7] proposed to use the nearest neighbors (NN), Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA) ML algorithms to map different network applications to pre-determined QoS traffic classes. Five levels of flow-based features were used to perform the classification between seven categories of applications. Although around 10% error rate is acceptable, the fact that too many features are used (several times larger in number than that of feature set adopted in our approach) more or less restricts the computational efficiency.

Moore and Zuev [8] proposed to apply the supervised ML Naïve Bayes technique to identify Internet traffic by eight categories of applications. Fast Correlation-Based Filter (FCBF) method was used to select the most important ones from 248 full-flow-based features. Yet, the complexity of the classification algorithm is not that ideal.

Crotti et al. [9] proposed a classification method based on normalized thresholds, using only three properties: packet length, inter-arrival time and packet arrival order, which were expressed in a compact structure called protocol fingerprint. As a result, fewer applications can be identified well since the features are not adequate after all. (Only three applications: HTTP, SMTP and POP3 are tested for identification in their experiment, and around 91% flow accuracy can be achieved.)

There're also numbers of classification techniques using clustering approaches (unsupervised ML). For detail, please refer to [10].

Compared with the approaches mentioned above, our method of classification to be discussed later uses fewer statistical features (only ten features, with the space complexity when calculating reduced to the level of O(1)) and a relatively simple classifier model (KNN estimator), which means computational efficiency and less memory space required, with high classification accuracy at the mean time.

III. Setup for Operational Deployment

Throughout the study, we have used data sets collected by a network sniffer known as Wireshark [11], which measures time to the accuracy of nanosecond. Also, its powerful functions of filtering help a lot when picking up pure traffic of certain applications whenever training or testing the classifier.

Considering that different link environment varies in packet loss ratio and other parameters, we've collected data sets for several different periods in time from one site on the Internet. We took part in various network activities involving different network applications from the site that was a host placed in the campus network in SJTU with Wireshark running on it. Traffic was monitored for each traffic-set consists of a random ten minutes' period and for both link directions.

A. Objects for Classification

The main idea of traffic classification is the progress mapping the objects of traffic to certain categories of applications. Actually, the definition of objects for classification matters very much, especially when selecting statistical features that describe the objects in terms of numeric parameters.

The objects for classification in our design are bi-directional full-flows. A bi-directional flow is a pair of uni-directional flows (a series of packets sharing the same five-tuple) going in the opposite directions between the same source and destination IP addresses and ports. A full-flow is defined as a bi-directional flow captured over its entire lifetime, from the establishment to the end of the communication connection. Hence, we use flow hash to differentiate flows which is calculated as the exclusive-OR value of the five-tuple:

$$\begin{aligned} flow\_hash = protocol \oplus src\_addr \oplus \\ src\_port \oplus dest\_addr \oplus dest\_port \end{aligned} \tag{1}$$

B. Definition of Accuracy

When evaluating different classification techniques, we need some kind of metric to show the accuracy results. Either byte or flow accuracy is all right for this, which is known as percentages of bytes or flows relative to the total number of bytes or flows of certain applications among the traffic being classified.

Erman et al. in [12] argue that byte accuracy is crucial when evaluating the accuracy of traffic classification algorithm. Cases are that, although just a few "large flows" (accounting for a big portion of total bytes and packets) are misclassified, byte accuracy drops a lot while flow accuracy still remains high.

In spite of this, since our work is just classifying traffic flows but not so sensitive to byte accuracy, flow accuracy seems more direct-viewing when discussing the classification results. The term accuracy in our study is known as flow-recall (percentages of flows of class X correctly classified as belonging to class X).

IV. Classification Using Machine Learning

Supervised ML technique we use in our work can be divided into two major phases: training and testing.

Fig. 1 captures the overall training process. First, traffic traces known for categories of applications are collected as PCAP files. The "flow statistical features processing" step involves calculating the statistical properties of these flows such as average packet inter-arrival time and flow duration.

As is noted in Fig. 1, a feature selection step is needed for the purpose of feature refinement (by reducing the number of needed features). Within the training process, flow-based feature sets are marked with known categories of applications and then combined together to form a feature sets holder as the output of the training phase.

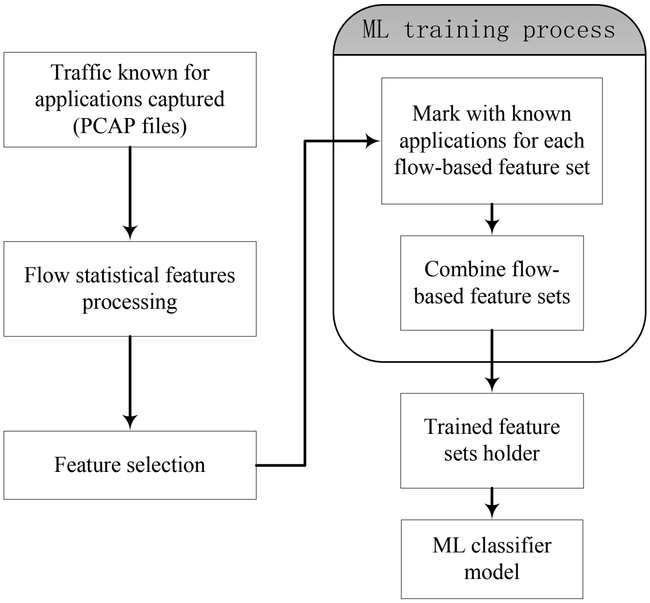
ML training process
Traffic known for applications captured (PCAP files)
Flow statistical features processing
Feature selection
Mark with known applications for each flow-based feature set
Combine flow-based feature sets
Trained feature sets holder
ML classifier model

Figure 1. Data flow within the training phase

Fig. 2 illustrates the data flow within the classification phase. Traffic traces unknown for applications are captured as PCAP files. After calculating the selected flow statistics, flow-based feature sets are inputted into the classifier model to be categorized.

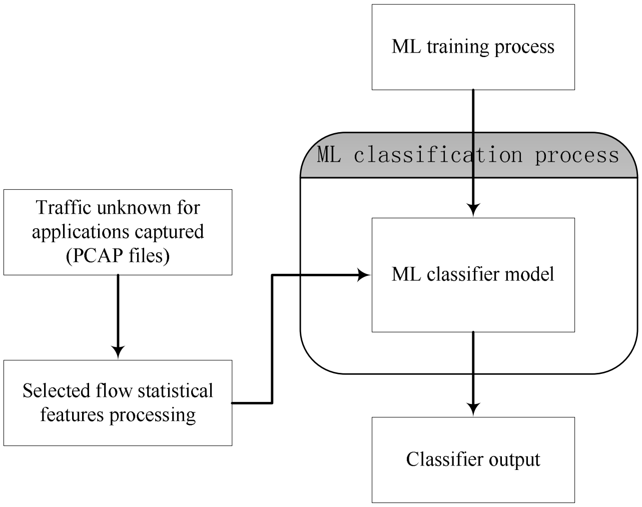
ML training process
ML classification process
Traffic unknown for applications captured (PCAP files)
ML classifier model
Selected flow statistical features processing
Classifier output

Figure 2. Data flow within the classification phase

A. *Traffic Categories & Feature Selection*

As is noted in section II, classifying traffic with flow-based statistical features has the underlying assumption that traffic generated by different categories of applications owns unique flow-based properties. This assumption seems too ideal to remain true when distinguishing some applications similar in flow statistics (e.g., the two objects for classification being both peer-2-peer applications like BitTorrent [13] and eMule [14]). As a result, certain granularity is needed when classify applications into categories.

In our work, the categories of applications which we're interested in are illustrated in Table I. Examples of applications are also listed for better understanding of certain categories.

As is noted in section II, Moore and Zuev [8] have used FCBF method [15] to refine the feature space in size and in quality. (In FCBF, a feature becomes good if it is highly correlated with the class, yet is not correlated with any other good features.) Since their interested categories of applications were WWW, MAIL, BULK, SERV, DB, P2P, ATT and MMEDIA, which is almost the superset of those of ours except Instant Messaging (IM, mainly for testing in our work), the results of FCBF methods will be used for reference here in our study.

TABLE I. CATEGORIES OF NETWORK TRAFFIC

Categories	Example Applications
MAIL	pop3, smtp
WWW	http, https
BULK	ftp
IM	MSN, QQ
P2P	BitTorrent, eMule
STREAM	youtube, youku

11 most important features that were identified by them are listed in Table II where s represents server and c represents client. (The full version of 248 features is given in [16].)

TABLE II. 11 MOST IMPORTANT FEATURES AFTER FCBF

Index	Statistical Features
01	Port server
02	No. of pushed data packets s→c
03	Initial window bytes c→s
04	Initial window bytes s→c
05	Average segment size s→c
06	IP data bytes median c→s
07	Actual data packets c→s
08	Data bytes in the wire variance s→c
09	Minimum segment size c→s
10	RTT samples c→s
11	No. of pushed data packets c→s

The reason why just 11 features are chosen is that the number of features for the best classification results varies from time to time with different data samples. And in Moor and

Zuev's study, up to 11 features are needed for the best results of classification.

Among these features, port server seems a little different with other ones since it is not possible to show quantitative difference between two different port numbers. (It cannot be said that port number 100 is 900 less than port number 1000.) Moreover, our study aims at an approach that is totally insensitive to port numbers. That is to say, port number is just used for calculating flow_hash (mentioned in section III-A) and has no more semantic meaning. So, here in our work, port server will not be used as one of the statistical features.

Besides, IP data bytes median client→server is so costing in memory space since all records need to be reserved to calculate the median. In our work, average IP data bytes client→server is used for substitution, which reduce the space complexity to the level of O(1). Honestly speaking, the rationality of this substitution remains to be studied further, although the two statistical parameters are almost the same.

Table III lists the final version of flow-based statistical features in our design. (Notice that features of indexes 01, 02, 03, 04, 05, 09 and 10 are TCP-specific.)

TABLE III. STATISTICAL FEATURES USED IN OUR WORK

Index	Statistical Features
01	No. of pushed data packets s→c
02	No. of Pushed data packets c→s
03	Initial window bytes c→s
04	Initial window bytes s→c
05	Average segment size s→c
06	Average IP data bytes c→s
07	Actual data packets c→s
08	Data bytes in the wire variance s→c
09	Minimum segment size c→s
10	RTT samples c→s

B. K-Nearest Neighbor Estimator

K-Nearest Neighbor Algorithm [17] was invented by Cover and Hart in 1968, which is a mature approach in theory. The training samples in k-Nearest Neighbor Algorithm are multi-dimensional vectors in N-dimension (in our case, N=10) vector space. The whole space is then covered densely by the training samples of different categories of applications.

In classification phase, an object is classified by a majority vote of its neighbors, with the object being assigned to the category most common amongst its k nearest neighbors.

The concrete algorithm is: assuming k1, k2, ..., kc each belongs to class ω1, ω2, ..., ωc, we define Decision function:

$$g_i(x) = k_i, i = 1,2,\ldots,c \tag{2}$$

And the decision would be:

$$g_j(x) = \max g_i(x) \rightarrow x \in \omega_j \tag{3}$$

Since the Euclidean distance is not applicable due to the fact that the selected features are different in measurement, the Mahalanobis distance [18] is used to measure the distance between two samples in the ten-dimension vector space.

Formally, the Mahalanobois distance from a group of values with mean $\mu = (\mu_1, \mu_2, \mu_3, \ldots, \mu_N)^T$ and covariance matrix S for a multivariate vector $x = (x_1, x_2, x_3, \ldots, x_N)^T$ is defined as:

$$D_M(x) = \sqrt{(x-\mu)^T S^{-1} (x-\mu)} \tag{4}$$

The Mahalanobois distance can also be defined as a dissimilarity measure between two random vectors $\vec{x}$ and $\vec{y}$ of the same distribution with the covariance matrix S:

$$d(\vec{x}, \vec{y}) = \sqrt{(\vec{x}-\vec{y})^T S^{-1} (\vec{x}-\vec{y})} \tag{5}$$

Fig. 3 gives a concrete example. For two-dimension feature space, there are two kinds of training data, let's say they are square and triangle. And there also exists an unknown class of object, let's say it is a ball. We all know that the ball would be classified as either square class or triangle class after applying KNN algorithm.

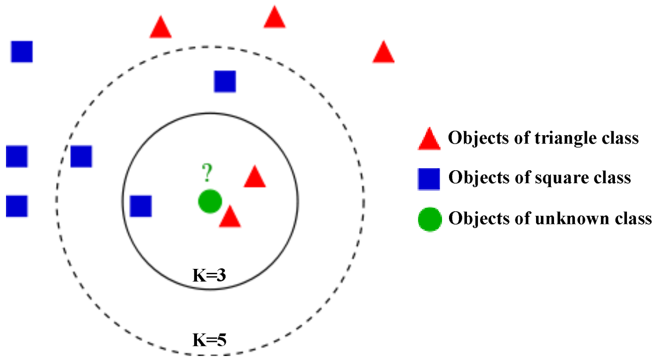

Figure 3. K-Nearest Neighbor in two-dimension feature space

As is displayed in Fig. 3, if we choose K=3, then in the three-neighbor field (as black solid line described), there exists two samples of triangles class and one sample of square class. So we categorize the ball to triangle class, since the number of samples of triangle class is in the majority. If we choose K=5, then in the five-neighbor field (as black dashed line described), we categorize the ball to square class. In the three-dimension space, the neighbor field is a spheroid, and in the four-dimension space, it is a hyper-sphere and so forth. We can extend the neighbor field to N-dimension.

As is shown by the example above, the choice of value of K would lead to different classification results. In practice, value of K depends on the training data samples in terms of distribution, number of classes and so on. Generally, a big K would reduce the impact of "noise sample" (e.g., a data sample of class A happens to own statistical features more similar to that of class B, which can be regarded as a noise sample). But on the other hand, it would also blur the border between categories. A good value of K could be retrieved from many Heuristic techniques such as cross-validation. Particularly, if k equals 1, the object is simply categorized to the class of its nearest neighbor.

V. RESULTS

In our work, we carry out a relatively simple kind of testing: several data sets including 9 flows of MAIL, 100 flows of WWW, 34 flows of BULK, 100 flows of IM, 100 flows of P2P and 5 flows of STREAM (full-flow) are collected in the way mentioned in section III, all of which are used to train the k-Nearest Neighbor estimator. (We will see, later in this part, that a vast difference among the number of sample flows of different classes might affect the results.) Another part of the data sets collected (only used for classification, but not for training) remains to be classified by the classifier model.

The powerful filter functions of Wireshark help us to carry out classification tests of different granularity:

TABLE IV. FLOW RECALL OF 3-CLASS KNN, K=3

Class	MAIL	WWW	BULK
Recall	100%	99.3%	100%

TABLE V. FLOW RECALL OF 4-CLASS KNN, K=5

Class	MAIL	WWW	BULK	IM
Recall	50%	99.8%	100%	100%

TABLE VI. FLOW RECALL OF 6-CLASS KNN, K=7

Class	MAIL	WWW	BULK	IM
Recall	50%	88.9%	100%	100%
Class	P2P	STREAM		
Recall	46.1%	100%		

Table IV, V and VI give the classification results with three, four and six categories of applications classified. In fact, just as is mentioned in section IV-B, it is not necessary that the number of classes and the value of K remain the relationship of n<=K. A relatively big K is only for reducing the impact of "noise sample". The values of K we use here are all decided by experiments. In other words, they are just suitable for the training data sets in our experiment since value of K is much sensitive to the data samples.

An overall look on classification results is shown by Fig. 4. It can be inferred that the classifier model works perfectly when classifying only MAIL, WWW and BULK flows. But with IM flows added, classification results of MAIL flows drop greatly. The probable reason is that MAIL flows are so similar in selected statistical features with IM flows, and the number of MAIL flows for training is much less than that of other flows, which breaks the principle of fairness in KNN algorithm (the number of objects of each class is preferred to be the same). More problems are discovered when P2P flows are added. Although with relatively adequate samples for training, the low flow-recall of P2P flows (only 46.1%) results mainly from the confusion in selected statistical features between P2P flows and IM flows (among 53.9% misclassified P2P flows, around 80% are misclassified to IM class, with the small remaining portion misclassified to WWW class).

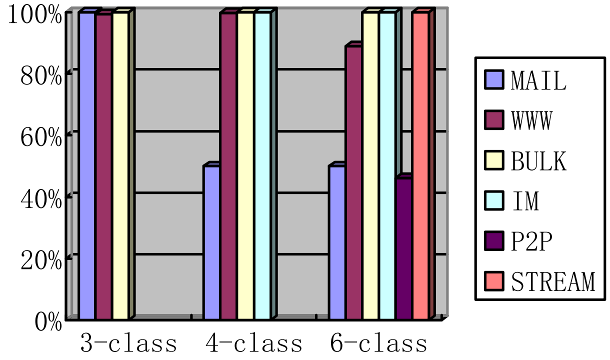

Figure 4. Flow recall of KNN algorithm

The results described above again show us the importance of feature selection. Actually, as is mentioned in section IV-A, data samples of IM were originally out of consideration when FCBF algorithm was first applied for feature selection in [8], which means the correlation between IM and other applications has not been take into account at all. Further work of feature selection is needed to distinguish IM flows from others.

VI. CONCLUSIONS & FUTURE WORK

In this paper we have demonstrated a statistical-feature-based approach to classify Internet traffic using supervised Machine Learning (ML). No more information than headers in IP and transport layers is need for classification. Taking no account of extra-added Instant Messaging (IM) flows (mainly for testing), the classifier model performs well in traffic classification with above 90% flow accuracy, which shows no inferiority compared with that of similar research work like Roughan et al. 's work [7]. Moreover, the simplified statistical features and the easy-to-use k-Nearest Neighbor (KNN) estimator result in lower space and time complexity, which is worth mentioning.

In our future work, complete tests about the current approach are badly needed (e.g., using adequate data samples for training for the sake of the principle of fairness in KNN algorithm). Furthermore, we'll apply numbers of refinements to improve the classification results such as meticulous process of feature selection and optimization of the classifier model.

This paper is just one of those that have applied ML techniques in traffic classification. Further investigations like using unsupervised ML techniques to classify unknown applications (not marked when training phase at all) and classification techniques on the basis of multi-flow topology [19] are also on the schedule if needed.

ACKNOWLEDGMENT

This work was supported by Project supported by the National Natural Science Foundation of China (Grant No.60773093), the Key Program for Basic Research of Shanghai (Grant No.08JC1411800), the Ministry of Education, and Intel joint research foundation (Grant No.MOE-INTEL-08-11).

REFERENCES

[1] Bro intrusion detection system, http://bro-ids.org/, as of June 25, 2009.

[2] V. Paxson, "Bro: A system for detecting network intruders in real-time," *Computer Networks*, no. 31 (23-24), pp. 2435-2463, 1999.

[3] L. Stewart, G. Armitage, P. Branch, and S. Zander, "An architecture for automated network control of QoS over consumer broadband links," in *IEEE International Region 10 Conference (TENCON2005)*, Melbourne, Australia, November 2005.

[4] A. Moore and K. Papagiannaki, "Toward the accurate identification of network applications," in *Proc. Passive and Active Measurement Workshop (PAM2005)*, vol. 3431, Springer-Verlag LNCS, Boston, MA, U.S.A., March/April 2005.

[5] *Internet Assigned Numbers Authority (IANA)*, http://www.iana.org/assignments/port-numbers, as of June 19, 2009.

[6] A. Madhukar and C. Williamson, "A longitudinal study of p2p traffic classification," in *Proceedings of the 14th IEEE International Symposium on Modeling, Analysis, and Simulation (MASCOTS2006)*, pp. 179-188, Washington, D.C., U.S.A., September 2006.

[7] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield, "Class-of-service mapping for QoS: A statistical signature-based approach to IP traffic classification," in *Proceedings of ACM/SIGCOMM Internet Measurement Conference (IMC2004)*, pp. 135-148, Taormina, Sicily, Italy, October 2004.

[8] A. Moore and D. Zuev, "Internet traffic classification using Bayesian analysis techniques," in *ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS2005)*, pp. 50-60, Banff, Alberta, Canada, June 2005.

[9] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic classification through simple statistical fingerprinting," *SIGCOMM Computer Communication Review*, vol. 37, no. 1, pp. 5–16, 2007.

[10] T. T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys and Tutorials*, vol. 10, no. 4, pp. 56-76, 2008.

[11] *Wireshark*, http://www.wireshark.org/, as of June 25, 2009.

[12] J. Erman, A. Mahanti, and M. Arlitt, "Byte me: a case for byte accuracy in traffic classification," in *Proceedings of the 3rd annual ACM workshop on Mining network data (MineNet2007)*, pp. 35–38, New York, NY, USA, June 2007.

[13] *BitTorrent*, http://www.bittorrent.com/, as of June 25, 2009.

[14] *eMule*, http://www.emule.org/, as of June 25, 2009.

[15] Lei Yu and Huan Liu, "Feature selection for high-dimensional data: A fast correlation-based filter solution," in *Proceedings of the 20th International Conference on Machine Learning (ICML2003)*, pp. 856-863, 2003.

[16] A. W. Moore, D. Zuev, and M. Crogan, "Discriminators for use in flow-based classification," Technical Report RR-05-13, Department of Computer Science, Queen Mary, University of London, September 2005.

[17] J. Laaksonen and E. Oja, "Classification with learning k-nearest neighbors," in *Proceedings of International Conference on Neural Networks,* vol. 3, pp. 1480-1483, Washington, D.C., U.S.A., June 1996.

[18] R. De Maesschalck, D. Jouan-Rimbaud and D.L. Massart, "The Mahalanobis distance," *Chemometrics and Intelligent Laboratory*, vol. 50, pp. 1–18, 2000.

[19] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "Blinc: Multilevel traffic classification in the dark," in *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM2005)*, pp. 229-240, Philadelphia, Pennsylvania, USA, August 2005.