# Internet Traffic Identification using Machine Learning

Jeffrey Erman, Anirban Mahanti, and Martin Arlitt
Department of Computer Science
University of Calgary
Calgary, AB, Canada T2N 1N4
Email: {erman, mahanti, arlitt}@cpsc.ucalgary.ca

*Abstract*— **We apply an unsupervised machine learning approach for Internet traffic identification and compare the results with that of a previously applied supervised machine learning approach. Our unsupervised approach uses an Expectation Maximization (EM) based clustering algorithm and the supervised approach uses the Naïve Bayes classifier. We find the unsupervised clustering technique has an accuracy up to 91% and outperform the supervised technique by up to 9%. We also find that the unsupervised technique can be used to discover traffic from previously unknown applications and has the potential to become an excellent tool for exploring Internet traffic.**

## I. Introduction

Accurate classification of Internet traffic is important in many areas such as network design, network management, and network security. One key challenge in this area is to adapt to the dynamic nature of Internet traffic. Increasingly, new applications are being deployed on the Internet; some new applications such as peer-to-peer (P2P) file sharing and online gaming are becoming popular. With the evolution of Internet traffic, both in terms of number and type of applications, however, traditional classification techniques such as those based on well-known port numbers or packet payload analysis are either no longer effective for all types of network traffic or are otherwise unable to deploy because of privacy or security concerns for the data.

A promising approach that has recently received some attention is traffic classification using *machine learning* techniques [1]–[4]. These approaches assume that the applications typically send data in some sort of pattern; these patterns can be used as a means of identification which would allow the connections to be classified by traffic class. To find these patterns, flow statistics (such as mean packet size, flow length, and total number of packets) available using only TCP/IP headers are needed. This allows the classification technique to avoid the use of port numbers and packet payload information in the classification process.

In this paper, we apply an *unsupervised* learning technique (EM clustering) for the Internet traffic classification problem and compare the results with that of a previously applied *supervised* machine learning approach. The unsupervised clustering approach uses an Expectation Maximization (EM) algorithm [5] that is different in that it classifies unlabeled training data into groups called "clusters" based on similarity.

The Naïve Bayes classifier has been previously shown to have high accuracy for Internet traffic classification [2]. In parallel work, Zander *et al.* focus on using the EM clustering approach to build the classification model [4]. We complement their work by using the EM clustering approach to build a classifier and show that this classifier outperforms the Naïve Bayes classifier in terms of classification accuracy. We also analyze the time required to build the classification models for both approaches as a function of the size of the training data set. We also explore the clusters found by the EM approach and find that the majority of the connections are in a subset of the total clusters.

The rest of this paper is organized as follows. Section II presents related work. In Section III, the background on the algorithms used in the Naïve Bayes and EM clustering approaches are covered. In Section IV, we introduce the data sets used in our work and present our experimental results. Section V discusses the advantages and disadvantages of the approaches. Section VI presents our conclusions and describes future work avenues.

## II. Background and Related Work

There has been much recent work in the field of traffic classification. This section will survey the different techniques presented in the literature.

### A. Port Number Analysis

Historically, traffic classification techniques used well-known port numbers to identify Internet traffic. This was successful because many traditional applications use fixed port numbers assigned by IANA [6]. For example, email applications commonly use port 25. This technique has been shown to be ineffective by Karagiannis *et al.* in [7] for some applications such as the current generation of P2P applications which intentionally tries to disguise their traffic by using dynamic port numbers or masquerade as well-known applications. In addition, only those applications whose port numbers are known in advance can be identified.

### B. Payload-based Analysis

Another well researched approach is analysis of packet payloads [7]–[10]. In this approach, the packet payloads are analyzed to see whether or not they contain characteristics signatures of known applications. These approaches have been shown to work very well for Internet traffic including P2P traffic. However, these techniques also have drawbacks. First, payload analysis poses privacy and security concerns. Second, these techniques typically require increased processing and storage capacity. Third, these approaches are unable to cope with encrypted transmissions. Finally, these techniques only identify traffic for which signatures are available and are unable to classify previously unknown traffic.

### *C. Transport-layer heuristics*

Transport-layer heuristic information has been used to address the drawbacks of payload-based analysis and the diminishing effectiveness of port-based identification. Karagiannis *et al.* propose a novel approach that uses the unique behaviors of P2P applications when they are transferring data or making connections to identify this traffic [7]. This approach is shown to perform better than port-based classification and equivalent to payload-based analysis. In addition, Karagiannis *et al.* created another method that uses the social, functional, and application behaviors to identify all types of traffic [11].

### *D. Machine Learning Approaches*

Machine learning techniques generally consists of two parts: model building and then classification. A model is first built using training data. This model is then inputted into a classifier that then classifies a data set.

Machine learning techniques can be divided into the categories of unsupervised and supervised. McGregor *et al.* hypothesize the ability of using an unsupervised approach to group flows based on connection-level (i.e., transport layer) statistics to classify traffic [1]. In this method, an EM algorithm [5] is used and McGregor *et al.* draw the conclusion that this approach is promising. In [3] and [4], Zander *et al.* extend this work by using an EM algorithm called AutoClass [12] and find the optimal set of attributes to use for building the classification model.

Some supervised machine learning techniques, such as [13] and [2], also use connection-level statistics to classify traffic. In [13], Roughan *et al.* use nearest neighbor and linear discriminate analysis. This approach is limited because it does not classify HTTP traffic and uses a limited number of connection-level statistics. In [2], Moore *et al.* suggests using Naïve Bayes as a classifier and shows that the Naïve Bayes approach has a high accuracy classifying traffic.

## III. Machine Learned Classification

Both approaches studied in this paper classify Internet traffic using flow statistics. This connection information is used to build the classification models (called classifiers) for both approaches. This section presents an overview of the machine learning techniques used in this work.

### *A. Supervised Machine Learning Approach*

The Naïve Bayes classifier is the supervised machine learning approach used in this paper. Assuming that flow attributes are independent and identically distributed, Moore *et al.* applied the Naïve Bayes classifier and found that this approach has good accuracy for classifying Internet traffic [2]. Here we provide an overview of this method and point the interested reader to [2] for details.

The Naïve Bayes method estimates the Gaussian distribution of the attributes for each class based on labeled training data. A new connection is classified based on the conditional probability of the connection belonging to a class given its attribute values. The probability of belonging to the class is calculated for each attribute using the Bayes rule:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \tag{1}$$

where A is a given class and B is a fixed attribute value. These conditional probabilities are multiplied together to obtain the probability of an object belonging to a given class A. In this paper, we used the Naïve Bayes implementation in the WEKA software suite version 3.4 [14]. This software suite was also used by Moore *et al.* for their analysis [2].

### *B. Unsupervised Machine Learning Approach*

The unsupervised machine learning approach is based on a classifier built from clusters that are found and labeled in a training set of data. Once the classifier has been built, the classification process consists of the classifier calculating which cluster a connection is closest to, and using the label from that cluster to identify that connection.

*1) Clustering Process:* The clustering process finds the clusters in a training set. This is an *unsupervised* task that places objects into groupings based on similarity; this approach is unsupervised because the algorithm does not have a priori knowledge of the true classes. A good set of clusters should exhibit high intra-cluster similarity and high inter-cluster dissimilarity.

We use an implementation of the EM clustering technique called AutoClass [12] to determine the most probable set of clusters from the training data. AutoClass calculates the probability of an object being a member of each discrete cluster using a finite mixture model of the attribute values for the objects belonging to the cluster. This assumes that all attribute values are conditionally independent and that any similarity of the attribute values between two objects is because of the class they belong to.

When this algorithm is initially run, the parameters of the finite mixture model for each cluster are not known in advance. The EM algorithm has two steps: an expectation step and a maximization step. The initial expectation step guesses what the parameters are using pseudo-random numbers. Then in the maximization step, the mean and variance are used to reestimate the parameters continually until they converge to a local maximum. These local maxima are recorded and the EM process is repeated. This process continues until enough samples of the parameters have been found (we use 200 cycles in our experimental results). A best set of parameters is selected based on the intra-cluster similarity and inter-cluster dissimilarity.

*2) Using Clustering Results as a Classifier:* Once an acceptable clustering has been found using the connections in a training data set, the clustering is transformed into a classifier by using a transductive classifier [15]. In this approach, the clusters are labeled and a new object is classified with the label of the cluster which it is most similar to.

We labeled a cluster with the most common traffic category of the connections in it. If two or more categories are tied, then a label is chosen randomly amongst the tied category labels. A new connection is then classified with the traffic class label of the cluster it is most similar to.

## IV. Experimental Results

This section evaluates the effectiveness of both the Naïve Bayes and AutoClass algorithms. First, the data sets used in this study are outlined. Second, the criteria measuring the effectiveness of the techniques is introduced. Finally, the experimental results are shown.

### *A. Data Sets*

Data from two publicly available traces is used in this work. Both traces present a snapshot of the traffic going through the Internet infrastructure at the University of Auckland. Due to the large size of the traces, only a subset of each trace is used (Auckland IV and Auckland VI [16]). The Auck-IVsub data set consists of all traffic measured during the 72 hour period on March 16, 2001 at 06:00:00 to March 19, 2001 at 05:59:59 from the Auckland IV trace. The Auck-VIsub data set used in this work from the Auckland VI trace is a subset from June 8, 2001 at 06:00:00 to June 9, 2001 at 05:59:59.

*1) Connection Identification:* To collect the statistical flow information necessary for the tests, the flows must be identified within the traces. These flows, also known as connections, are a bidirectional exchange of packets between two nodes. These two nodes can be identified based on their IP addresses and transport layer port numbers which stay constant during the connection.

In both traces, the data is not exclusively from connection-oriented transport layer protocols (e.g., TCP). Some of the traffic originates from UDP and ICMP which are not connection-oriented. While some connection related statistics could be collected for these, we removed connection-less traffic from our data sets because our primary interest was in applications that used TCP.

The TCP/IP header data recorded for the packets in both traces allow identification of connections by SYN/FIN packets. A connection is started when a SYN packet is sent and is closed when FIN packets are sent. A connection that did not have a packet sent between the nodes for over 60 seconds and no FIN packet was received was also closed. Once a connection has been identified, the following statistical flow characteristics are calculated: Total Number of Packets, Mean Packet Size (in each direction and combined), Mean Data Packet Size, Flow Duration, and Mean Inter-Arrival Time of Packets. Our decision to use these characteristics was based primarily on the previous work done by Zander *et al.* [3]. Due to the heavy-tail distribution of many of the characteristics, we found that the logarithms of the characteristics gives much better results using both approaches [14], [17].

TABLE I
Traffic class breakdown for Auck-IVsub data set

| Traffic Class | Port Numbers | # of Connections | % of Total |
|---|---|---|---|
| http | 80, 8080, 443 | 3,092,009 | 81.2% |
| smtp | 25 | 118,211 | 3.1% |
| dns | 53 | 75,513 | 2.0% |
| socks | 1080 | 69,161 | 1.8% |
| irc | 113 | 53,446 | 1.4% |
| ftp (control) | 21 | 50,474 | 1.3% |
| pop3 | 110 | 37,091 | 1.0% |
| limewire | 6346 | 10,784 | 0.3% |
| ftp (data) | 20 | 5,018 | 0.1% |
| other | - | 295,732 | 7.8% |

*2) Classification of the Test Data Sets:* The test data needs to be pre-classified so it can validate the results from the algorithms (i.e., a *true classification* is needed). Since the traces are publicly available, and therefore, only contain TCP/IP header information, no payload-based identification method can be used to determine the *true* classes. Therefore, port-based identification is used. While port-based identification is becoming increasingly ineffective we feel this should still provide accurate results for the traces used in this paper. This is because the emergence of dynamic port numbers in P2P traffic did not happened until late 2002 [18]; in 2001 the Auckland traces were collected.

Table I presents summary statistics of the traffic classes (along with the identifying port numbers) for the Auck-IVsub data set. For HTTP data, all connections that have a destination port of 80, 8080 and 443 are included. The reason for the inclusion of port 443 containing encrypted HTTP data is that at the connection-level, it behaves the same as unencrypted HTTP. This allows both encrypted and unencrypted packets that originate from the same applications to be identified from the same class or application.

When calculating the number of connections belonging to each type of class, the results showed that the majority of the connections in the two data sets were HTTP traffic. This large amount of HTTP traffic in the data sets does not test the approaches well for identifying any traffic class with the exception of HTTP. Therefore, to enable a fair analysis, the data sets used for the training and testing have equal samples of 1000 random connections of each traffic class. This allows the accuracy achieved in the test results to fairly judge the ability of both machine learning techniques to classify all types of traffic classes and not just HTTP.

### *B. Effectiveness Criteria*

To measure the effectiveness of the algorithms three metrics were used: *precision*, *recall*, and *overall accuracy*. These measures have been widely used in the data mining literature to evaluate data clustering algorithms [14]. For a given class, the number of correctly classified objects is referred to as the True Positives. The number of objects falsely identified as a class are referred to as the False Positives. The number of objects from a class that are falsely labeled as another class is referred to as the False Negatives.

*Precision* is the ratio of True Positives to True and False Positives. This determines how many identified objects were correct.

$$precision = \frac{TP}{TP + FP}. \qquad (2)$$

*Recall* is the ratio of True Positives to the number of True Positives and False Negatives. This determines how many objects in a class are misclassified as something else.

$$recall = \frac{TP}{TP + FN}. \qquad (3)$$

*Overall accuracy* is defined as the sum of all True Positives to the sum of all the True and False Positives for all classes. This measures the overall accuracy of the classifier. Note that precision and recall are per-class measures.

$$overall\ accuracy = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n}(TP_i + FP_i)}, \qquad (4)$$

where *n* is the number of classes. Precision and recall are related to each other. If the Recall for one class is lower, this will cause the precision for other classes also to be lower because the algorithms used always classify the objects into a class. In addition, the overall accuracy is related to precision in that it measures the average precision of all classes.

### *C. Naïve Bayes Classifier Results*

For each data set, the Naïve Bayes classifier is first trained with a training set containing 1000 random samples of each traffic class. Once this training is complete the classifier is then tested to see how well it classifies 10 different test sets containing 1000 (different) random samples of each traffic class. The classification of the test set is what is used to calculate the effectiveness criteria. The minimum, maximum, and average precision and recall results for the Auck-IVsub data set are shown in Figure 1. The results for Naïve Bayes using the Auck-VIsub data set are qualitatively similar to the Auck-VIsub data set (These results are not shown due to space limitations.).

An analysis of the results from the Auck-IVsub data set shows that, on average, the precision and recall for six of the nine classes were above 80%. It performed best for IRC connections with 95.0% precision and 94.5% recall, followed by 87.2% precision and 88.6% recall for POP3 connections. Conversely, it performed worst for SOCKS and LIMEWIRE connections with precisions of 69.7% and 73.4%, respectively. The poor performance is owing to 10% of both the LIMEWIRE and FTP-data transfers being falsely classified as SOCKS and consequently contributing to their lower recall values. For LIMEWIRE, HTTP and SOCKS were the main traffic classes being falsely classified.

Overall, the Naïve Bayes classifier performs well for these test data sets with the majority of the traffic classes being classified with average precision and recall values above 80%.

### *D. AutoClass Results*

This section presents results for the unsupervised machine learning approach using AutoClass. In this approach, for each of the data sets, the training set of data is first clustered using AutoClass to produce clusters of objects that are similar to each other. Then a transductive classifier is built using these clusters using the method previously described. The resulting classifier is then used to predict which traffic class a new connection belongs to from the 10 test sets of data. Figures 2 presents the minimum, maximum, and average precision and recall results for the Auck-IVsub data set.

Figure 2 shows that the values for precision and recall are, on average, much higher than those obtained using the Naïve Bayes approach. In Figure 2, all classes have precision and recall values above 80%. Note that six out of the nine classes have average precision values above 90%, and seven have average recall values above 90%. The two worst classified classes, HTTP and LIMEWIRE, still have precisions and recalls over 80%. The reason HTTP had this lower precision was that approximately 10% of the SOCKS connections were being incorrectly classified as HTTP. The LIMEWIRE classification accuracy was low primarily because HTTP was being incorrectly classified as LIMEWIRE.

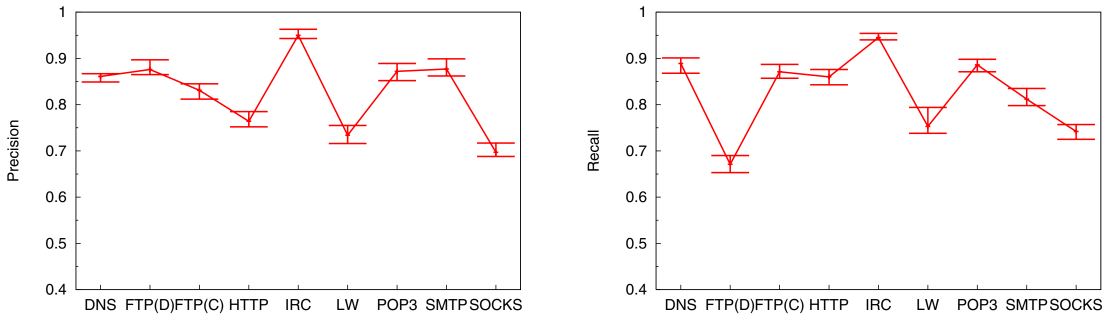
The clusters produced by AutoClass were individually analyzed. This gives further insight as to why some connections are being falsely classified. For example, we examined one of the clusters where HTTP was being falsely classified as LIMEWIRE. In this cluster of 111 connections, 37 were HTTP (33%) and 66 were LIMEWIRE (59%). The number of packets sent for all the connections in this cluster was 12. The average packet size for all the connection was 106 bytes with the HTTP connections having an average of 118 bytes and LIMEWIRE 101 bytes. The average duration was 0.5 seconds with HTTP and LIMEWIRE have 0.7 and 0.3 seconds, respectively.

The results for AutoClass using the Auck-VIsub data set are qualitatively similar to the Auck-IVsub data set.

Overall, the AutoClass approach performs quite well for the data sets with precision and recall values averaging around 91% for both data sets.

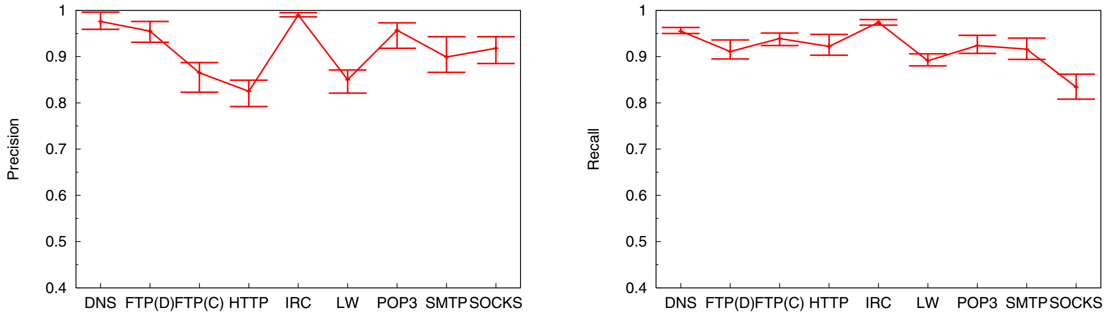
### *E. Overall Accuracy of Algorithms*

The examination of the overall accuracy between the Naïve Bayes classifier and the AutoClass approach can be seen in Table II. In the Auck-IVsub data set, AutoClass has an average overall accuracy of 91.2% whereas in comparison, the Naïve Bayes classifier has an overall accuracy of 82.5%. Thus, for this data set, we find that AutoClass outperforms the Naïve Bayes classifier by 9%. This shows that the unsupervised machine learning approach is at least as good as the supervised learning approach without requiring the training data to be labeled beforehand.



(a) Precision (b) Recall

Fig. 1. Naïve Bayes classifier results for Auck-IVsub data set



(a) Precision (b) Recall

Fig. 2. AutoClass results for Auck-IVsub data set

TABLE II

OVERALL ACCURACY OF EACH ALGORITHM (AUCK-IVSUB DATA SET)

| Algorithm | Average | Minimum | Maximum |
|---|---|---|---|
| Naïve Bayes | 82.53% | 81.92% | 83.31% |
| AutoClass | 91.19% | 90.51% | 91.70% |

## V. DISCUSSION

In the previous section we showed that while the unsupervised cluster approach has better accuracy than the Naïve Bayes classifier, both performed fairly well at classifying the connections. Both algorithms offer some distinct benefits over the payload-based approaches. As mentioned in Karagiannis *et al.* [7], non-payload based approaches have less privacy issues to consider because the private information inside packets are not examined. Less storage and processing overhead is incurred because less information is needed to be processed when only dealing with packet headers. Finally, these approaches will not be inhibited by payloads being encrypted. However, one disadvantage for both algorithms is that they rely on the training data being representative of the overall network traffic. If the training data no longer remains representative then the classifiers must be retrained.

The unsupervised cluster approach offers some additional benefits because it does not require the training data to be labeled. For example, new applications can be identified by examining the connections that are grouped to form a cluster. Typically, clusters created correspond to a single application. Therefore, only a small subset of the connections in each cluster must be identified in order to have a high confidence as to what each cluster contains. This could result in significant time savings for the operator of this approach because the hand classification of the training set could take a significant amount of time.

### A. Runtime Analysis

The runtime of both approaches is an important consideration because the model building phase is computationally time consuming. For the analysis, all operations are performed on a Dell Optiplex GX620 with an Intel Pentium IV 3.4 GHz processor and 1GB of RAM. The number of data objects in the training set was varied between 1000 and 128000. In general, the runtime for the Naïve Bayes classifier was significantly less than AutoClass when building the classification models. For example, with 8000 objects Naïve Bayes took 0.06 seconds whereas AutoClass took 2070 seconds to build the classification model. Both approaches did exhibit a linear growth pattern as the number of objects increased. Although the Naïve Bayes classifier was faster, the size the the training set is ultimately limited by the amount of memory because both approaches must load the entire training set into memory before building the model.

### B. Weight of each AutoClass Cluster

As may be expected, some clusters AutoClass identified contain considerably more connections than other clusters. In this section, the number of connections in each of the clusters is analyzed from the clusters produced by five of the Auck-IVsub training data sets. This is a useful analysis because it

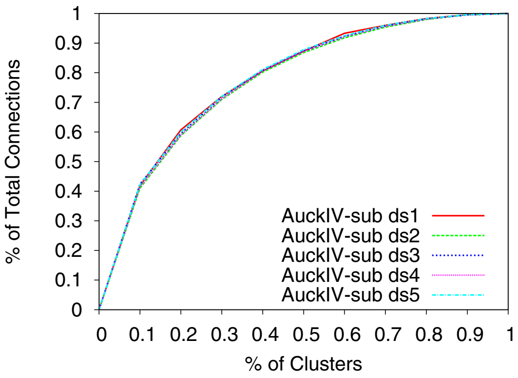


Fig. 3. CDF of the weight of each cluster created by AutoClass

demonstrates that if the AutoClass approach was used, the traffic class that each cluster corresponds to would not need to be identified for all clusters and still produce good results.

The CDF graphs in Figure 3 show the total number of connections as a function of the number of clusters for five of the Auck-IVsub data sets. In the Auck-IVsub data set there were 123 clusters found on average. This graph indicates that the last 20% of the clusters produced represent only 2% of the total connections. Identifying these clusters will not change the overall accuracy of the clustering significantly. These graphs further show that 80% of the connections can be represented with 50% of the clusters. This means that to identify 80% of the connections only half of the clusters need to be analyzed so as to determine which traffic class it belongs, in order to generate the transductive classifier.

## VI. Conclusions

This paper presented an unsupervised machine learning approach (AutoClass) for Internet traffic classification. We used qualitative and quantitative results to compare this approach to a supervised machine learning approach (Naïve Bayes classifier). Our results show that AutoClass can achieve an average accuracy greater than 90%. For the data sets considered in this paper, we find that AutoClass outperforms Naïve Bayes by up to 9%.

We also determined that the time required to classify connections can be reduced with the unsupervised clustering technique. The time savings can be achieved because only a portion of the connections in each cluster must be manually identified. Not all clusters are necessarily needed to have fairly accurate results.

Overall, the unsupervised machine learning approach achieved better results and can be concluded to be at least as good while greatly reducing the amount of manual configuration. This is a very promising result. In the future, this approach could become an excellent tool to explore the traffic on a network, separating connections into groups that can be easily used to identify the applications transmitting the data.

We are pursuing this work in several directions. Our immediate next step is to apply the unsupervised clustering approach to a more recent trace that may contain peer-to-peer and streaming media traffic. In this work, only AutoClass based on Bayesian classification theory was used as the clustering method. The data mining literature contains many other clustering algorithms based on different theories and approaches [19]. Currently, we are exploring some of these unique clustering algorithms; results from our preliminary investigation can be found in [20].

## Acknowledgment

This work was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada and Informatics Circle of Research Excellence (iCORE) of the province of Alberta. We thank Carey Williamson for his comments and suggestions which helped improve this paper.

## References

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

[2] A. Moore and D. Zuev, “Internet Traffic Classification Using Bayesian Analysis Techniques,” in *SIGMETRICS’05*, Banff, Canada, June 6-10, 2005.

[3] S. Zander, T. Nguyen, and G. Armitage, “Self-Learning IP Traffic Classification Based on Statistical Flow Characteristics,” in *PAM 2005*, Boston, USA, March 31-April 1, 2005.

[4] ——, “Automated Traffic Classification and Application Identification using Machine Learning,” in *LCN’05*, Sydney, Australia, November 15-17, 2005.

[5] A. Dempster, N. Paird, and D. Rubin, “Maximum likelihood from incomeplete data via the EM algorithm,” *Journal of the Royal Statistical Society*, vol. 39, no. 1, pp. 1–38, 1977.

[6] IANA. Internet Assigned Numbers Authority (IANA), “http://www.iana.org/assignments/port-numbers.”

[7] T. Karagiannis, A. Broido, M. Faloutsos, and K. claffy, “Transport Layer Identification of P2P Traffic,” in *IMC’04*, Taormina, Italy, October 25-27, 2004.

[8] P. Haffner, S. Sen, O. Spatscheck, and D. Wang, “ACAS: Automated Construction of Application Signatures,” in *SIGCOMM’05 Workshops*, Philadelphia, USA, August 22-26, 2005.

[9] A. Moore and K. Papagiannaki, “Toward the Accurate Identification of Network Applications,” in *PAM 2005*, Boston, USA, March 31-April 1, 2005.

[10] S. Sen, O. Spatscheck, and D. Wang, “Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures,” in *WWW2005*, New York, USA, May 17-22, 2004.

[11] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, “BLINK: Multilevel Traffic Classification in the Dark,” in *SIGCOMM’05*, Philadelphia, USA, August 21-26, 2005.

[12] P. Cheeseman and J. Strutz, “Bayesian Classification (AutoClass): Theory and Results.” *In Advances in Knowledge Discovery and Data Mining, AAI/MIT Press, USA*, 1996.

[13] 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 *IMC’04*, Taormina, Italy, October 25-27, 2004.

[14] I. Witten and E. Frank, *(2005) Data Mining: Pratical Machine Learning Tools and Techniques*, 2nd ed. San Francisco: Morgan Kaufmann, 2005.

[15] A. Banerjee and J. Langford, “An Objective Evaluation of Criterion for Clustering,” in *KDD’04*, Seattle, USA, August 22-25, 2004.

[16] Auckland Data Sets, “http://www.wand.net.nz/wand/wits/auck/.”

[17] V. Paxson, “Empirically-Derived Analytic Models of Wide-Area TCP Connections,” *IEEE/ACM Transactions on Networking*, vol. 2, no. 4, pp. 316–336, August 1998.

[18] C. Colman, “What to do about P2P?” *Network Computing Magazine*, vol. 12, no. 6, 2003.

[19] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Englewood Cliffs, USA: Prentice Hall, 1988.

[20] J. Erman, M. Arlitt, and A. Mahanti, “Traffic Classification using Clustering Algorithms,” in *SIGCOMM’06 MineNet Workshop*, Pisa, Italy, September 2006.