Feature Optimization and Performance Evaluation of Machine Learning Algorithms for Identification of P2P Traffic

Sunil Agrawal
University Institute of Engineering & Technology, Panjab University, Chandigarh, India
Email: s.agrawal@hotmail.com

Balwinder S. Sohi
Senior Member, IEEE,
Campus Director, Surya World, Bapror, Patiala, Punjab, India
Email: bssohi@yahoo.com

***Abstract* - P2P applications supposedly constitute a substantial proportion of today's Internet traffic. The ability to accurately identify different P2P applications in internet traffic is important to a broad range of network operations including application-specific traffic engineering, capacity planning, resource provisioning, service differentiation, etc. However, current P2P applications use several obfuscation techniques, including dynamic port numbers, port hopping, and encrypted payloads. As P2P applications continue to evolve, robust and effective methods are needed for identification of P2P applications. It is general practice to reduce the cost of classification by reducing the number of features, utilizing some feature selection algorithm. But such algorithms are highly data-dependent and do not yield good result when tried upon other data set. In this paper, we propose an optimized set of features and compare five supervised ML algorithms for identification of the P2P traffic. It is found that NBTree outperforms other ML algorithms with 96.6% precision and 99.7% recall, when they are trained and tested on the same data set. As far as training time is concerned, BayesNet is the best with precision and recall very close to that of NBTree.**

***Index Terms* – Flow features, Feature selection, Machine learning (ML) algorithms, Traffic classification.**

I. INTRODUCTION

Over the last few years, peer-to-peer (P2P) file-sharing has dramatically grown to represent a significant component of Internet traffic. P2P volume is sufficiently dominant on some links, thereby causing an increased local peering among Internet Service Providers. This is observable, but its effect could not be quantified on the global Internet topology and routing system not to mention competitive market dynamics. Despite this dramatic growth, reliable profiling of P2P traffic remains elusive, as the newer generation P2P applications are incorporating various strategies to avoid detection. We no longer enjoy the benefit of first generation P2P traffic, which was relatively easily classified due to its use of well-defined port numbers. Current P2P applications tend to intentionally disguise their generated traffic to circumvent both filtering firewalls as well as legal issues. Not only do most P2P applications now operate on top of nonstandard, custom designed proprietary protocols, but also current P2P clients can easily operate on any port number. Internet service providers as well as enterprise networks require the ability to accurately identify the different P2P applications, for a range of uses, including network operations and management, application-specific traffic engineering, capacity planning, resource provisioning, service differentiation and cost reduction.

These circumstances lead to a conclusion that accurate identification of P2P traffic is only possible by examining user payload. Yet user payload capture and analysis is not possible because of issues like legal, privacy, technical, logistic, and financial. Further P2P applications tend to support payload encryption, obfuscating payload characterization attempts. Therefore, the frequency with which P2P protocols are being introduced and/or upgraded renders user payload analysis impractical as well as inefficient.

P2P application identification inside IP networks, in general, can be difficult. In an ideal situation, a network administrator wants to have precise information on the applications running inside the network, along with unambiguous mappings between each application and its associated network traffic. However, in general, such information is rarely available. If it is available, then it may not be up-to-date or complete, and identifying either the applications or their associated traffic is a challenging task.

At present, much of the existing research focuses on the achievable accuracy of different ML (Machine Learning) algorithms. Selecting few most relevant features out of a number of features is very important aspect to build ML classifier, as to improve its

© 2012 ACADEMY PUBLISHER
doi:10.4304/jait.3.2.107-114

accuracy as well as its computational performance. Currently, the researchers are using some feature selection algorithms, which produce different feature sets for different data sets, and are claiming good accuracy. But the effect of using same set of features on the different data sets and devising a common feature set has attracted almost no investigation. In this paper, we mainly focus on devising an optimized feature set applicable to every data set and its effect on the performance of five supervised ML algorithms for identification of P2P traffic in the internet.

II. PROBLEM STATEMENT AND OUR PROPOSAL

A. Machine learning (ML) for IP traffic classification

ML classification algorithms assume that a 'class' of traffic can be identified using statistical analysis of traffic features. They can utilize either unsupervised (clustering) or supervised learning (classification) approaches. Supervised learning involves learning from a set of pre-classified examples to classify unseen examples. It consists of two stages - *training* the ML algorithm to associate sets of features with known traffic classes, and *testing* - applying the learning to classify unknown traffic.

In contrast, supervised ML techniques are provided with pre-defined classes; whereas, unsupervised ML techniques discover natural groups in the data using some internal heuristics and group instances with similar properties (may be defined by a specific distance measuring approach, e.g. Euclidean space) into clusters. We are going to use following metrics to evaluate five supervised ML algorithms for identification of P2P traffic in the inetrnet. If a classifier is trained to identify members of class X, then the performance metrics are defined as follows:

- *Recall*: Defined as percentage of members of class X correctly classified as belonging to class X.
- *Precision*: Defined as percentage of those instances that truly have class X, among all those classified as class X.

These metrics range from 0 (poor) to 100% (optimal). It is important to note here that high precision is meaningful only when the classifier attains good recall, because recall tells about the ability of the classifier to correctly identify the instances of the target class.

B. Related work

Most protocols contain a protocol-specific string (called application signature) in the payload that can be used for application identification. These strings are often public information and can also be obtained by examining a number of network traffic traces.

Sen et al. [1] proposed an approach for detection of P2P traffic through application-layer signatures. They examined available documentation and packet-level traces to identify application layer signatures, and then utilized these signatures to develop filters that could track P2P traffic. They analyzed TCP packets in the download phase of file transfer. They decomposed P2P signatures into fixed pattern matches with fixed offsets and variable pattern matches with variable offset within a TCP payload. Those authors evaluated the accuracy and scalability of the application-layer signature technique. Their results showed that the proposed technique had <5% for both false positives and false negatives, indicating that the technique was accurate most of the time.

Application signature is obtained by analyzing the user payload. Though payload-based classifiers show good results but not suitable for newer generation of P2P traffic, because of two major limitations: (a) they cannot be used if payload information is not available and (b) they cannot identify unknown classes of traffic.

The research community has come up with an idea of using only transport layer information for traffic identification, in order to overcome the limitations of port and payload based classification.

Karagiannis et al. [2] proposed an approach to identify P2P flows at the transport layer. This approach was based on connection patterns, without relying on user's packet payloads. Their transport-layer approach relies primarily on two heuristics. The first one identifies source-destination IP pairs that concurrently use both TCP and UDP. If such IP pairs are found, not using specific well-known ports, then these flows are considered P2P flows. The second one considers the structural pattern of transport-layer connections between hosts.

Hu et al [3] proposed a profile-based approach to identify traffic flows belonging to the P2P application. Depending on the patterns dominant in the application, they built behavioral profiles of the target application. Based on this behavioral profile, they used a two-level matching method to identify new traffic. At first level matching, they determined whether a host participates in the target application by comparing its behavior with the profiles. At the second level matching, they compared each flow of the host with those patterns in the application profiles to determine which flows belong to this application. The results showed that popular P2P applications could be identified with high accuracy.

Li et al. [4] compared the effective and efficient classification of network-based applications using behavioral observations of network-traffic and those using deep-packet inspections. They demonstrated the accurate training of models from data with a high-confidence ground-truth and the use of an efficient and small feature set derived from the reduction of a large flow feature list using a sophisticated feature selection mechanism.

The preceding techniques are highly dependent on the information gathered to build behavioral profile of the host. But the requirement of large data-base, more computation, and other implementation issues limit their usefulness. Newer techniques rely on traffic's

© 2012 ACADEMY PUBLISHER

statistical characteristics to identify the target application. Such techniques assume that traffic at the network layer has statistical properties (such as the distribution of flow duration, flow idle time, packet inter-arrival time and packet lengths) that are unique for certain classes of applications and enable different applications to be distinguished from each other.

Yu et al [5] used ML algorithms for spam email classification and found out that RVM is most suitable for spam classification in terms of the applications that require low complexity. They also found out that owing to dynamism of internet traffic, Machine Learning techniques are far better than any other technique for identification of target application in the internet traffic.

In this paper, we use the supervised Machine learning (ML) techniques, which provide a promising alternative in classifying flows, based on payload independent statistical features derived from packet streams consisting of one or more packet headers. Each traffic flow is characterized by the same set of features but with different feature values. An ML classifier is built by training on a preclassified set of flow instances where the network applications are known. Then the built classifier can be used to determine the class of an unknown flow.

C. Data-sets

We illustrate our method with preclassified data described originally in [6]. This data consists of description of the internet traffics that have been classified manually, to provide the input sets for the training and testing phases. To construct the sets of data, the day trace was split into ten blocks (approximately 28 minutes each) of transport control protocol (TCP) traffic flows, with each instance described by its membership class and a set of 248 features. This feature set includes flow duration statistics, TCP Port information, payload size statistics, Fourier transform of the packet inter-arrival time and more. In order to provide a wider sample of mixing across the day, the start of each sample was selected randomly (uniformly distributed over the whole day trace).

While each of the data sets represent approximately the same period of time, the number of instances per data set fluctuates as a result of the variation in the activity throughout the course of the day. Further details of the original hand-classification are given in [7], and the data sets themselves are described at length in [6].

Our central object for classification is the traffic flow and for the work presented, we have limited our definition of a traffic flow to being a complete TCP flow, those that start and end validly, e.g., with the first SYN, and the last FIN ACK.

D. Feature Selection

Feature selection process removes irrelevant and redundant [8] features, i.e., those that can be excluded from the feature set without loss of classification accuracy, thus reducing computational complexity and increasing classification speed. It is a preprocessing step to machine learning, and is the process of choosing a subset of original features that will optimize for higher learning accuracy with lower computational complexity and higher classification speed.

We use the Correlation-based Filter (CFS), which is computationally practical and outperforms the other filter method (e.g. Consistency based Filter) in terms of classification accuracy and efficiency [9, 10]. The Correlation-based Filter examines the relevance [11] of each feature, i.e., those highly correlated to specific class but with minimal correlation to each other [10]. We use a Best First search algorithm to generate candidate subsets of features from the full feature set, since it provides higher classification accuracy than Greedy search algorithm.

But these feature selection algorithms are highly data-dependent, i.e. for every data-set, these algorithms result in different sets of features, which are unique to each data-set. It is found that for a given data-set, the classification algorithm gives the highest classification accuracy, only when its own unique feature set is used; and classification accuracy degrades for other feature sets. We want a common feature set, on the basis of which, all of the data-sets could be classified by our classification algorithms, without degradation in their performance. For this purpose, we have taken ten data-sets to devise a common reduced feature set.

We use the WEKA machine learning software suite [12], often used in traffic classification efforts; to evaluate five supervised machine learning algorithms.

III. MACHINE LEARNING ALGORITHMS

We use a range of supervised ML algorithms. A supervised or ‘inductive learning’ algorithm forms a model based on training data and uses this model to classify new data. We use the following five supervised ML algorithms, which are reported by different research papers to be performing well for most of the applications:

- Bayesian Networks
- Multilayer Perceptron Network
- Naive Bayes Tree
- C4.5 Decision Tree
- Naive Bayes

In the following subsections we briefly describe these ML algorithms.

A. C4.5 Decision Tree

The C4.5 algorithm [13] creates a model based on a tree structure. Nodes in the tree represent features, with branches representing possible values connecting features. A leaf representing the class terminates a series of nodes and branches. Determining the class of an instance is a matter of tracing the path of nodes and branches to the terminating leaf. C4.5 uses the ‘divide and conquer’ method to construct a tree from a set S of

© 2012 ACADEMY PUBLISHER

training instances. If all instances in S belong to the same class, the decision tree is a leaf labeled with that class. Otherwise the algorithm uses a test to divide S into several non-trivial partitions. Each of the partitions becomes a child node of the current node and the tests separating S is assigned to the branches.

The divide and conquer approach partitions until every leaf contains instances from only one class or further partition is not possible e.g. because two instances have the same features but different class. If there are no conflicting cases the tree will correctly classify all training instances. However, this over-fitting decreases the prediction accuracy on unseen instances.

C4.5 attempts to avoid over-fitting by removing some structure from the tree after it has been built. Pruning is based on estimated true error rates. C4.5 uses subtree replacement or subtree raising to prune the tree as long as the estimated error can be decreased.

B. NaiveBayes

A Bayes classifier is a simple probabilistic classifier based on applying Bayes' theorem (from Bayesian statistics) with strong (naive) independence assumptions. A more descriptive term for the underlying probability model would be "independent feature model". In simple terms, a naive Bayes classifier assumes that the presence (or absence) of a particular feature of a class is unrelated to the presence (or absence) of any other feature.

Depending on the precise nature of the probability model, naive Bayes classifiers can be trained very efficiently in a supervised learning setting. In many practical applications, parameter estimation for naive Bayes models uses the method of maximum likelihood; in other words, one can work with the naive Bayes model without believing in Bayesian probability or using any Bayesian methods.

C. Bayesian Networks (BayesNet)

A Bayesian network, belief network or directed acyclic graphical model is a probabilistic graphical model that represents a set of random variables and their conditional independencies via a directed acyclic graph (DAG). Formally, Bayesian networks are directed acyclic graphs whose nodes represent random variables in the Bayesian sense: they may be observable quantities, latent variables, unknown parameters or hypotheses. Edges represent conditional dependencies; nodes which are not connected represent variables which are conditionally independent of each other. Each node is associated with a probability function that takes as input a particular set of values for the node's parent variables and gives the probability of the variable represented by the node. For example, if the parents are m Boolean variables then the probability function could be represented by a table of 2^m entries, one entry for each of the 2^m possible combinations of its parents being true or false. Efficient algorithms exist that perform inference and learning in Bayesian networks.

D. Naive Bayes Tree (NBTree)

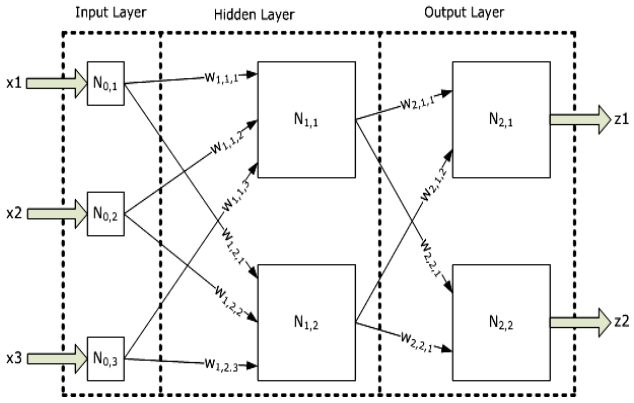
The NBTree [14] is a hybrid of a decision tree classifier and a NaiveBayes classifier. Designed to allow accuracy to scale up with increasingly large training datasets, the NBTree algorithm has been found to have higher accuracy than C4.5 or Naive Bayes on certain datasets. The NBTree model is best described as a decision tree of nodes and branches with Bayes classifiers on the leaf nodes.

As with other tree-based classifiers, NBTree spans out with branches and nodes. Given a node with a set of instances the algorithm evaluates the 'utility' of a split for each attribute. If the highest utility among all attributes is significantly better than the utility of the current node the instances will be divided based on that attribute. Threshold splits using entropy minimization are used for continuous attributes while discrete attributes are split into all possible values. If there is no split that provides a significantly better utility a Naive Bayes classifier will be created for the current node.

The utility of a node is computed by discretising the data and performing cross validation to estimate the accuracy using Naive Bayes. The utility of a split is the weighted sum of the utility of the nodes, where the weights are proportional to the number of instances in each node. A split is considered to be significant if the relative (not the absolute) error reduction is greater than 5% and there are at least 30 instances in the node.

E. Multilayer Perceptron (MLP)

The basic building block of a neural network [15] such as a multilayer perceptron is a processing unit called a neuron (or simply node). The output of a neuron is a combination of the multiple inputs from other neurons. Each input is weighted by a weight factor. A neuron outputs if the sum of the inputs exceeds a threshold function of the neuron. The output from a multiplayer perceptron is purely predictive. As there is no descriptive component, the resulting classification can be hard to understand. The architecture of the multilayer perceptron consists of a single input layer of neurons; one or multiple hidden layers and a single output layer of neurons (see Fig. 1).



© 2012 ACADEMY PUBLISHER

Figure 1. Architecture of MLP Neural Network

In order to learn the perceptron must adjust its weights. The learning algorithm compares the actual output to the desired output to determine the new weights repetitively for all training instances. The network trains with the standard backpropagation algorithm, which is a two-step procedure. The activity from the input pattern flows forward through the network, and the error signal flows backward to adjust the weights. The generalized delta rule adjusts the weights leading out of the hidden layer neurons and the weights leading into the output layer neurons. Using the generalized delta rule to adjust the weights leading to the hidden units is backpropagating the error adjustment.

Our multiplayer perceptron uses sigmoid threshold functions. The number of input nodes is equal to the number of attributes and the number of output nodes is equal to the number of classes. There is only one hidden layer, which has as many nodes as the sum of the number of attributes and the number of classes divided by two. As we use different feature selection techniques that produce different feature subsets the number of input and hidden nodes differs depending on the number of features used. By default the algorithm we use performs normalization of all attributes including the class attribute (all values are between -1 and +1 after the normalization). The learning rate (weight change according to network error) was set to 0.3, the momentum (proportion of weight change from the last training step used in the next step) to 0.2 and we ran the training for 500 epochs (an epoch is the number of times training data is shown to the network).

IV. RESULTS AND DISCUSSION

Our ultimate goal is to show the impact of the proposed feature set on the performance of our chosen ML algorithms, for detection of P2P applications in internet traffic. First, Best First search algorithm is used to generate candidate subsets of features from the full feature set. Further using correlation-based feature subset evaluation, we identify the best subset of features. This procedure of generating best subset of features is carried out for each data-set. These feature sets are given the name 'CFS-BF' feature set, and these feature sets are different for different data-sets. Then, we devise a common feature set, using the following procedure:

- Run the above-mentioned feature selection algorithm on the ten data-sets, and get the ten different sets of features, referred as 'CFS-BF' sets.
- Then we examine all ten feature sets, and construct a new feature set, which includes all those features which occur in more than five out of ten feature sets.
- By selecting features in this way, we could be able to keep the number of feature in our new feature set to a minimum, so as to ensure lower training and classification time.
- We refer this feature set as 'optimized' feature set, the description of which is given in Table 1.

TABLE III.
DESCRIPTION OF 'OPTIMIZED' FEATURE SET

Feature Number	Feature Description
1	Port Number at server
2	Time Stamp requested (Client to Server)
3	The missed Data, calculated as the difference between the ttl stream length and unique bytes sent (Client to Server)
4	The total number of full-size RTT samples (Client to Server)

Unlike CFS-BF feature sets, the optimized feature set is common to all data-sets, thereby facilitating the trained classifier to be used for any data-set, with reasonably good performance.

We evaluate the performance metrics for identification of P2P applications, for each ML algorithm using proposed feature set. Each ML algorithm is trained using first data-set and then the trained ML algorithm is tested using all of the ten data sets. A comparison of their performances using precision and recall as performance metrics is shown in the Fig. 2.

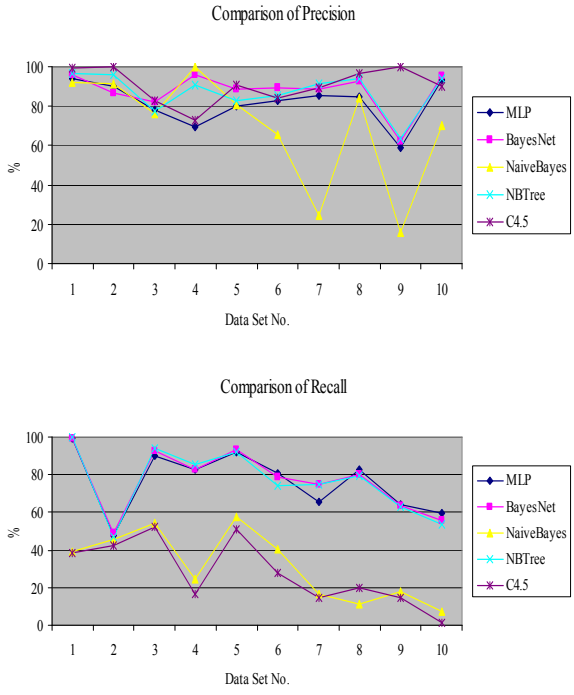


Figure 2. Performance comparison of five ML algorithms

We observe from Fig. 2, that the precision for C4.5 is consistently highest, but the recall value is very low. So the overall performance of this classifier is poor since classifier with high precision is useless until

© 2012 ACADEMY PUBLISHER

unless it achieves high value of recall also. The NaïveBayes algorithm achieves very low values for both precision and recall, rendering it worst performing algorithm. To compare the rest of the three ML algorithms in better way, we take the average of their performance metrics over the ten data sets as shown in the Fig. 3.

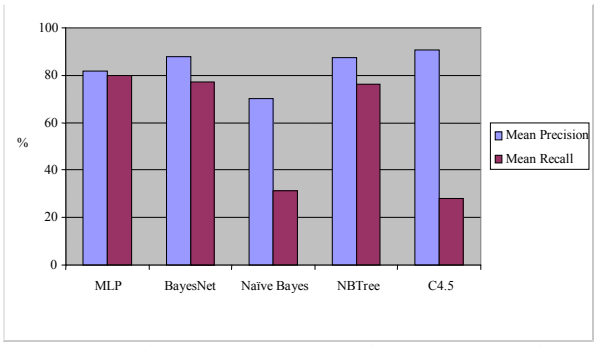


Figure 3. Performance comparison on the basis of mean values

From this figure, it can be seen that the BayesNet is able to show very high value of mean precision with reasonably good recall, and NBTree also performs almost equally. We observe highest value of mean recall for MLP algorithm, but with less precision as compared with BayesNet and NBTree. Depending on the nature of the task the ISP wants to carry out, one out of these three classifiers can be selected for identification of P2P traffic in the internet.

We expect the ML classifier to perform well when it is trained and tested on the same data set, and its performance degrades when the test data set is different from the training data set. This is because of the dynamism of the network traffic (specifically P2P traffic) in the internet. A further comparison of the performances of the chosen ML algorithms is shown in the Fig. 4, when the training and testing of ML algorithms is done on the same data set.

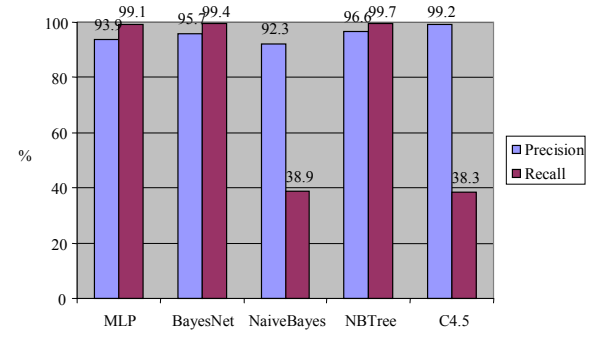


Figure 4. Performance comparison when trained and tested on the same data set

We can see from this figure that NaïveBayes and C4.5 perform very poorly because of their very low recall values. The rest of the three ML algorithms provide good performance where NBTree is the best with 96.6% precision and 99.7% recall values. Here one can suggest that the ML algorithm should be trained frequently in order to get optimum performance, but this is feasible only when we are interested in off-line analysis of the P2P traffic in the internet. To get an idea of the re-training interval and to test the portability of the ML classifier, we use a test data set (taken 1 year later) [16] from different site, and evaluate the performance of the above-mentioned ML classifiers, a comparison of which is shown in the Fig. 5.

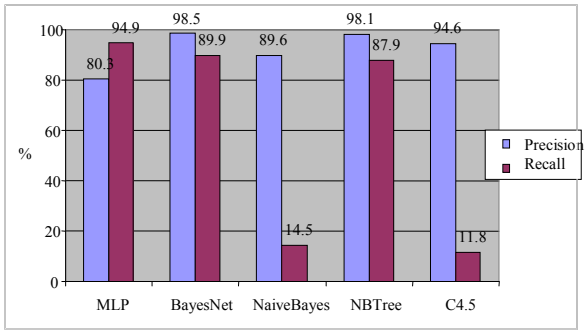


Figure 5. Performance comparison on test data taken one year later

It can be seen form the figure that the precisions of BayesNet and NBTree algorithms are still very high; with more decrease in the recall value for NBTree. The results here indicate that the NBTree is the best choice when it is possible to re-train frequently, preferably once in a few months, otherwise BayesNet is the best option, which requires less frequent re-training. At this point it is necessary to compare the training time of these ML algorithms, as shown in the Table 2.

TABLE II.
COMPARISON OF TRAINING TIME

ML algorithm	Training Time (in Seconds)
MLP	188.02
BayesNet	0.13
NaiveBayes	0.07
NBTree	7.89
C4.5	0.33

It is very clear now that the BayesNet is the only choice when the performance of ML algorithm is considered along with their training time. Overall we can conclude that there is a trade-off between performance (in terms of precision and recall), training time and frequency of re-training while selecting a particular ML algorithm for identification of P2P traffic in the internet.

Simulation and performance evaluation were carried out on a 3 GHz, Intel Core 2 Duo CPU workstation with 3GB of RAM.

V. CONCLUSION AND FUTURE WORK

P2P application is becoming more prominent and consuming as more as 70% bandwidth of IP network. Accurate identification of P2P applications facilitates in a broad range of network operations like capacity

© 2012 ACADEMY PUBLISHER

planning, network expansion, resource provisioning and lawful interception. To overcome the limitations of traditional approaches like port, payload and host behavior based analysis; researchers have shown an increased interest in the machine learning techniques for IP traffic classification.

This paper has demonstrated the selection of features and its successful application on five supervised ML algorithms for P2P traffic identification. Our main findings are as follows.

- It is found that the BayesNet and NBTree perform excellently when we use our proposed feature set.
- These ML algorithms show excellent prediction accuracies when they are trained and tested on the same data set. And also their training time is very less as compared with that of MLP algorithm.
- Testing of the trained ML algorithms on the data set taken one year later from the other site gives an indication of re-training interval and portability of the ML classifier.
- Our proposed feature set consists of only 4 features, which are to be calculated in the one direction only i.e. from Client to Server, thereby simplifying the feature extraction process. This feature set results in significant reduction in training time and an improvement in classification speed without any degradation in the performance, making this approach a low-overhead method with potential for real-time implementation for identification of P2P traffic in the internet.

There are a number of areas in which future work would further confirm the suitability of our proposed feature set. An evaluation on further sources of classified data from other sites will give insight into the stability of this approach. We know that new applications are coming up every day, causing a decline in the classification accuracy of the trained ML algorithms over time. For satisfactory performance of the ML algorithm with time, it is required that the ML algorithm is retrained after certain interval. Testing of ML algorithms with data set from later time will give an exact indication of the retraining interval. Specific implementation issues and algorithmic optimization need to be explored further.

REFERENCES

[1] S. Sen, O. Spatscheck, and D. Wang, "Accurate, Scalable In-Network Identification of P2P Traffic using Application Signatures," *Proceedings of the 13th International World Wide Web Conference*, pp. 512-521, NY, USA, May 2004.

[2] T. Karagiannis, A. Broido, M. Faloutsos, and K. Klaffy, "Transport Layer Identification of P2P Traffic," *Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement (IMC 2004)*, pp. 121-134, Italy, October 2004.

[3] Y. Hu, D. Chiu, and J.C.S. Lui, "Profiling and Identification of P2P Traffic," *Computer Networks, Volume 53, issue 6,* pp. 849-863, April 2009.

[4] W. Li, M. Canini, A.W. Moore and R. Bolla, "Efficient Application Identification and the Temporal and Spatial Stability of Classification Schema", *Computer Networks, Volume 53, issue 6,* pp. 790-809, April 2009.

[5] B. Yu, Z. Xu, "A Comparative study for Content-based dynamic spam classification using four machine learning algorithms", Knowledge-based Systems, vol. 21, pp. 355-362, 2008.

[6] A. Moore, D. Zuev, "Discriminators for use in flow-based classification", *Technical report, Intel Research*, Cambridge (2005).

[7] A. W. Moore and D. Papagiannaki, "Toward the accurate identification of network applications," *Proc. 6th Passive Active Meas. Workshop (PAM)*, vol. 3431, pp. 41–54, Mar. 2005.

[8] A. Appice, M. Ceci, S. Rawles, and P. Flach, "Redundant feature elimination for multi-class problems", *Proceedings of the 21st International Conference on Machine Learning*, Canada, July 2004.

[9] N. Williams, S. Zander, and G. Armitage, "Evaluating machine learning algorithms for automated network application identification", *Technical Report 060401B, CAIA, Swinburne Univ.*, April 2006.

[10] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification", *ACM SIGCOMM CCR*, 36(5):7–15, October 2006.

[11] A. Blum and P. Langley, "Selection of relevant features and examples in machine learning", *Artificial Intelligence*, 97(1-2):245–271, 1997.

[12] WEKA: Data Mining Software in Java. http://www.cs.waikato.ac.nz/ml/weka/.

[13] R. Kohavi and J. R. Quinlan, Will Klosgen and Jan M. Zytkow, editors, "Decision-tree discovery", In *Handbook of Data Mining and Knowledge Discovery*, chapter 16.1.3, pages 267-276, Oxford University Press, 2002.

[14] Ron Kohavi, "Scaling Up the Accuracy of Naive-Bayes Classifiers: a Decision-Tree Hybrid", *Proceedings of 2nd International Conference on Knowledge Discovery and Data Mining* (KDD), pp. 202-207, August 1996.

[15] Simon Haykin, "Neural Networks", Pearson Education, 2006.

Sunil Agrawal received his B.E. degree in Electronics & Communication in 1990 from Jodhpur University in Rajasthan, India and M.E. degree in Electronics & Communication in 2001 from Thapar University in Patiala, India.

He is Assistant Professor at the University Institute of Engineering & Technology in Panjab University, Chandigarh, India. He has 19 years of teaching experience (undergraduate and postgraduate classes of engineering) and has supervised several research works at masters level. He has several research papers to his credit in national and international conferences and journals.

The author's main interests include applications of artificial intelligence, QoS issues in Mobile IP, and mobile ad hoc networks.

© 2012 ACADEMY PUBLISHER

Balwinder S. Sohi is currently working as Campus Director at Surya World – a group of schools in Engineering & Technology. He has a long administrative experience as Director, University Institute of Engineering & Technology – a premier engineering institute of Panjab University at Chandigarh, India. He has served at various faculty positions as Professor, Assistant Professor and Lecturer, during his long professional carrier of 38½ years (including 4 years in research organizations). Graduated from Panjab Engineering College, Chandigarh, he has attained his masters and PhD degrees from Panjab University, Chandigarh.

He has guided the research works at masters and PhD levels and has more than 70 national and international research papers to his credit. He has been responsible in setting up various facilities in the field of Electronics & Communication, through sponsored projects from agencies like MHRD, AICTE, DIT etc. He has been Dean of Engineering & Technology at Panjab University, Chandigarh. He has contributed to technical education in various capacities at different fora like AICTE etc. He has been honored twice by Institute of Engineers, Kolkata, India, for best research work.

© 2012 ACADEMY PUBLISHER