A Preliminary Investigation On The Identification of Peer to Peer Network Applications

Can Bozdodan
Megamation Systems
Halifax, NS, Canada
canbozdogan@gmail.com

Yasemin Gokcen
IBM
Halifax, NS, Canada
gokcenyaseminn@gmail.com

Ibrahim Zincir
Department of Computer Engineering
Yasar University
Izmir, Turkey
ibrahim.zincir@yasar.edu.tr

ABSTRACT
Identification of P2P (peer to peer) applications inside network traffic plays an important role for route provisioning, traffic policing, flow prioritization, network service pricing, network capacity planning and network resource management. Inspecting and identifying the P2P applications is one of the most important tasks to have a network that runs efficiently. In this paper, we focus on identification of different P2P applications. To this end, we explore four commonly used supervised machine learning algorithms as C4.5, Ripper, SVM(Support Vector Machines), Naïve Bayesian and well known unsupervised machine learning algorithm K-Means on four different datasets. We evaluate their performances to identify the P2P applications that each traffic flow belongs to. Evaluations show that, Ripper algorithm gives better results than the others.

Categories and Subject Descriptors
C.2.3 [**Computer-Communication Networks**]: Network Operations – *Network monitoring*

Keywords
Peer to Peer; P2P applications; Supervised and unsupervised machine learning; Network traffic classification.

1. INTRODUCTION
Identification and classification of network traffic is one of the most important tasks of network management today and it is still being developed. For example, to ensure that the qualities of service objectives are met for defined applications in a company or organization, bandwidth usage should be under control of a network administrator. In a case such as lack of resources, the network administrator may need to block the traffic created by some applications to keep other applications work efficiently which are used for important business processes. And the administrator may remove the restrictions for blocked applications when the resources are available or block them permanently. Moreover, identification of applications accurately in a network is a supportive task for network problems such as traffic policing, flow prioritization, network service pricing, network capacity planning and resource occupation management.

GECCO'15 Companion, July 11 - 15, 2015, Madrid, Spain

DOI: http://dx.doi.org/10.1145/2739482.2768432

There are many P2P applications exist online for different purposes such as file sharing and communication today. Distribution of copyrighted files via P2P applications is considered as illegal in many countries because of distribution without any permission. On the other hand, legally file sharing with a P2P application has advantages such as having fast and cheap service. This shows us depending on the usage objective, P2P applications can be really useful or can be harmful even illegal. So, directly blocking P2P application traffic for easy network management can restrict useful sides of the P2P traffic with its troublemaker sides. And directly allowing P2P traffic can create the problems we mentioned before. Identification and applying restrictions on P2P applications is absolutely necessary task for security, quality and management of network, but restricting all P2P traffic can result with reducing availability of useful applications. It is believed that identification of applications separately and specifically in a network traffic can support network managers to block traffic of P2P applications, which are reducing the quality of network or can be used for illegal activities, and to allow traffic of P2P applications which are useful and effective for users. Thus, this paper focuses on the supervised and unsupervised learning algorithms to maximize efficiency in identifying P2P applications. To this end, we explore the usage of different supervised machine learning algorithms namely C4.5 [1], Ripper [1], SVM [2], Naïve Bayesian [3] and well known unsupervised learning algorithm K-Means [1] to classify network traffic flows [6]. To be able to evaluate these different algorithms, four different datasets (generated by network traffic of Research and Development Laboratory 5 of Yasar University) are employed.

The remaining of the paper is organized as follows: Section 2 briefly describes previous works done before to analyze peer-to-peer network traffic flow. The description of the machine learning algorithms and data sets are given in Section 3. Experiments and results are discussed in Section4. Finally Section 5 discusses conclusions and future work.

2. LITERATURE SURVEY
There are many different works done in the literature, which are based on identification of P2P network traffic. Different algorithms are applied; different features of the network traffic flows are used to receive more accurate classification results for identification of P2P traffic. But this work specifically focuses on classifying different torrent protocols in a network traffic file. Instead of identifying a flow only as P2P or non-P2P, we believe that being able to define specific torrent protocols can be really useful for researchers who work on managing traffic flow tasks with permissions and blockings for specific network packages. In this section, first it is briefly described few of the works that are done to classify P2P traffic inside network flow and this gives us

an overview on the machine learning methods which gave successful results on network traffic flow datasets.

Hyunchul et al. investigates the classification approaches as port based approach, payload based approach, host behavior based approach and flow features based approach to identify the network traffic flow [4]. Naive Bayes, Bayesian Network, Naive Bayes Kernel Estimation, C4.5, neural Networks and Support Vector Machines (SVM) machine learning methods are applied to analyze the traffic. Depending on the authors' results, it is shown that the most effective approach is port based classification and SVM achieved the highest accuracy on datasets. Also authors mention that each approach has its own strengths and weaknesses, and different combinations of techniques can provide synergy.

Feng et al. developed a new method that is based on Support Vector Machine for identification of P2P network traffic [5]. Their method identifies network traffic by analyzing packet length, remote hosts' discreteness, connection responded success rate and the ratio of IP and port at the host level without relying on the port numbers and packet payload, which are used as classification metrics. Authors analyze P2P traffic at host level and they research on the flows and the interaction between the flows. The focus is identifying a P2P host by calculating the classification metrics every 30 seconds, and labeling the class of applications for each IP. Authors mention that they retrieved good results by this approach with high and stable success rates.

On the other hand, in other work of Alshammari et al. used machine learning methods to classify encrypted network traffic to identify SSH and Skype [6]. SVM, AdaBoost, Naïve Bayesian, C4.5 and Ripper algorithms are applied to data sets to evaluate their efficiency in traffic identification. The results show that while using flow based features approach, C4.5 algorithm generated better model than other classifiers with highest detection rate and lowest false positive rate values.

Furthermore, Jingyu et al. reviewed main technologies of P2P traffic identification based on network traffic characteristics [7]. P2P traffic characteristics are introduced and studied by focusing on the false negative and negative detection reasons. In addition, the authors present an integrated solution algorithm which is able to detect P2P traffics with dynamic ports and encrypted transmission. Results show that the proposed method is able to prevent non-P2P flow to be identified as P2P flow with zero false alarm rates for all used P2P applications in tests.

Tiexing et al. proposed a traffic identification methodology that is named as PeerIdentifier [8]. The Authors describe that PeerIdentifier can separate the P2P file sharing traffic from traditional application traffic and identify hosts which participate in P2P activities and the listening ports of the P2P clients running on these hosts. Creating large number of connections to remote hosts feature of P2P applications is used by the authors to identify the traffic of P2P sharing applications. PeerIdentifier has two major modules as flow statistics tool, which gets information from flow statistics provided by routers, and traffic identification module, which uses series of identification algorithms to be used for connection patterns. Authors mention that experimental tests of PeerIdentifier shows, it is able to identify majority of P2P traffic with low false negative and false positive.

Soysal et al. evaluated and compared several flow based P2P traffic detection methods, which employ machine learning techniques [9]. Bayesian Networks, Multi-Layer Perceptron and C4.5 machine learning algorithms are employed on custom-made data where P2P flows can be uniquely identified. The authors mention that depending on the results, decision tree based techniques such as C4.5 have a very fast run time and are suitable for real time accuracy evaluation.

Kun et al. presented the concept of feedback by getting help from electronics as transistor's current and voltage [10]. And the authors proposed feedback model based on traffic control. Their system aims to determine the selection of upload nodes according to traffic generating request of terminal nodes in P2P network traffic. The authors mention that according to their experimental results, P2P traffic feedback model based on the amplifier principle is available for controlling traffic in P2P networks with its good precision and efficiency.

He et al. proposed a detailed analysis and modeling of P2P traffic of different applications [11]. The authors gathered and identified P2P traffic on link between Japan and USA. Then they analyzed the distributions of the traffic volume, connection duration, connection interval times of P2P connections. As P2P applications, Napster, BitTorrent, EDonkey, Gnutella and Fastrack are analyzed. By their research, the authors mention that the traffic generated by the different types of applications has to be modeled with different probability distributions.

Chunzhi et al. put forward a method to identify P2P traffic based on multi dimension characteristics of P2P applications [12]. As multi-dimension characteristics, ratio of UDP/TCP protocol packet, average data transmitting size and up/down traffic ratio are chosen. The authors mention that test results show them that multidimensional characteristics can effectively identify P2P traffic in network with low false alarm rate and missing report rate.

As can be seen from all of the above, in the previous works, algorithms and approaches are tested to evaluate their efficiency for identifying P2P network traffic. However our work is focusing on specifically defining four P2P applications of network traffic. Brief information about datasets and applied algorithms are discussed in following section.

3. METHODOLOGY

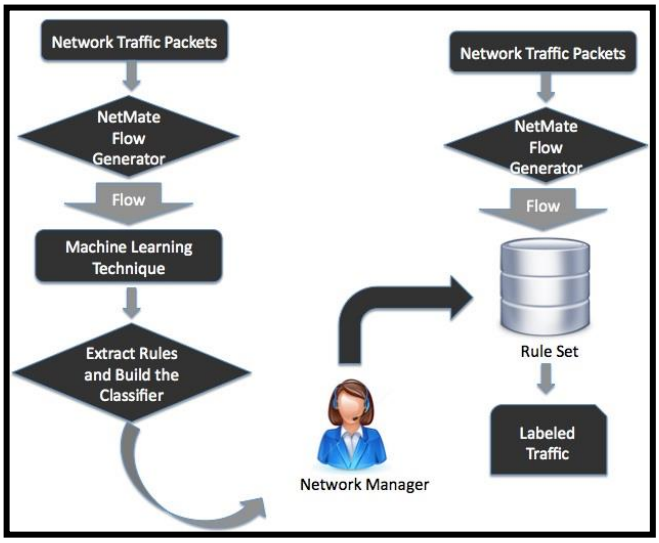

Figure 1. Representative system

Identification of P2P network traffic is one of the essential and difficult tasks. And we believe that specifically identifying applications on a network is more helpful for security and availability tasks of a network. For better quality of service and better network management, systems which give support for network traffic classification has critical role for organizations and companies. Our aim is to evaluate machine-learning algorithms by their efficiency on classifying the network flows to detect P2P application that creates the network traffic. And we believe that an efficient algorithm can be useful for a system that aims to support network managers. The system that our approach is employed can be represented as in the Figure 1.

3.1 Data Sets

Table 1. Features used for flow representation

1) Protocol
2) Duration of the flow in microseconds
3) Smallest packet length in the forward direction in bytes
4) Mean packet length in the forward direction in bytes
5) Biggest packet length in the forward direction in bytes
6) Standard deviation of the mean packet length in forward direction in bytes
7) Smallest packet length in the backward direction in bytes
8) Mean packet length in the backward direction in bytes
9) Biggest packet length in the backward direction in bytes
10) Standard deviation of the mean packet length in backward direction in bytes
11) Smallest time between two packets sent in forward direction in microseconds
12) Mean time between two packets sent in forward direction in microseconds
13) Biggest time between two packets sent in forward direction in microseconds
14) Standard deviation of mean time between two packets sent in forward direction in microseconds
15) Smallest time between two packets sent in backward direction in microseconds
16) Mean time between two packets sent in backward direction in microseconds
17) Biggest time between two packets sent in backward direction in microseconds
18) Standard deviation of mean time between two packets sent in backward direction in microseconds
19) Number of packets in the forward direction
20) Number of bytes in the forward direction
21) Number of packets in the backward direction
22) Number of bytes in the backward direction

In our experiments we used four datasets to evaluate effectiveness of the algorithms. Each data set is used to classify specific traffic created by an application in a network. Yasar University R&D Laboratory 5 computers are used to collect network traffic. 3 file sharing applications, BitCommet [13], BitTorrent [14] and UTorrent [15] are chosen to generate P2P network traffic. They were installed into the computers at the lab and then movies, TV series and music albums popular at the torrent sites were downloaded to generate traffic. From this generated traffic, two different datasets are created for training and testing purposes. Hereafter, we will call dataset used for BitCommet network traffic classification as BitCommetDS, dataset used for BitTorrent network traffic classification as BitTorrentDS, UTorrent network traffic classification as UTorrentDS and data set used for non-P2P traffic classification NonDS. Each dataset contains network flows of each applications and web surfing network traffic created by different web browser applications. This means that all datasets contain network flow of all applications employed and web surfing network traffic. To be sure about package labels, one desktop machine for each application is provided to record the traffic generated.

3.2 Feature Selection

The network traffic is represented as flow based features and these features are used to create input vectors for machine learning algorithms. In this case each network flow is described by statistical features. NetMate [16] tool is used to preprocess network traffic packages to create our datasets, which consist of network traffic flows. The features that can result in biasing as IP addresses and port numbers of source and destination are removed. The same features of traffic flows as used in [6], [17] and shown in [18] are extracted to form the input vector and to have comparison environment. For each flow, packet length, inter arrival time statistics, packet headers, duration, mean payload length, mean packet length, payload volume, payload rate, source IP, destination IP, total file length, flow byte, packets in forward direction, packets in backward direction, backward arrival time and forward arrival time are used for converting TCP,UDP and Ethernet into vectors to analyze network traffic. Table 1 shows the features that are used for traffic representation.

3.3 Algorithms

Please leave 3.81 cm (1.5") of blank text box at the bottom of the left column of the first page for the copyright notice.

3.3.1 Supervised Machine Learning Algorithms Employed

3.3.1.1 C4.5

C4.5 is an algorithm used to generate a decision tree developed by Ross Quinlan. It is an extension of earlier ID3 algorithm. The decision tree generated by C4.5 is used for classification. It builds decision trees from a set of training data using the concept of information entropy. The data is a set $A = a_1, a_2, ...$ of already classified samples. Each sample $a_i = x_1, x_2, ...$ represents attributes or features of the sample. The training data is a augmented with a vector $C = c_1, c_2,...$ where $c_1, c_2,...$ represent the class to which each sample belongs. At each node of the tree, C4.5 chooses one attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. Its criterion is the normalized information gain that results from choosing an attribute for splitting the data. The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurs on the smaller sub lists. A more detailed explanation can be found in [19].

3.3.1.2 RIPPER

Repeated Incremental Pruning to Produce Error Reduction is a rule based machine learning algorithm. The algorithm learns rules

directly from the data and it does a depth-first search, then it generates one rule at a time[1]. Each rule is created by conjunction of conditions on numeric or discrete attributes. And these conditions are added to optimize some criterion and to maximize an information gain measure. Minimum description length is used to measure the quality of a rule[1]. Growing of a rule is stopped when its description is at least 64bits larger than the best description length. All the training examples that satisfy a rule are removed from the training set after the growing and pruning is done for that rule. This process does not stop until the enough rules are added. More information about the algorithm can be found in [1].

3.3.1.3 SVM (Support Vector Machines)

Support vector machines (SVMs) are a set of related supervised learning methods used for classification regression. In simple words, given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. Intuitively, an SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on. In data classification, the objective of SVMs is to determine a relevant subset of data which is sufficient to separate input examples. There are two kinds o problems commonly faced in classification domain: when the data can be separated using linear function (hyper plane) and when the data must be separated using nonlinear hyper plane. Unfortunately, the bulk of real data sets are very complex due to large number of features and input patterns. Thus, it is difficult to determine whether the input data can be separated using linear hyper plane. Therefore, nonlinear classificatory are the most often used models in data separation. Data classifiers based on SVM methodology belong to this group of models and are usually called nonlinear SVMs. Naturally, these classifiers can also be applied to linear classification problems but nonlinear SVMs are the general case in the classification process performed by means of this algorithm. Further information on the SVM algorithm can be found in [2].

3.3.1.4 Naïve Bayesian

Naïve Bayesian classifier is a simple statistical classifier based on applying Bayes' theorem that uses conditional probability. The classifier assumes the values of input features are independent and it investigates relationship between instance of each class and each attributes to create a conditional probability which depends on the relationship between the class and the feature values. Because of conditional independence, computations are simple and it is considered as naïve. The algorithm computes the probability of occurrence of each class and computes the probability of occurrence of instance in a given class to create relationships. Naïve Bayesian algorithm can be trained very efficiently in a supervised learning setting because of the precise nature of the probability model. A more detailed explanation of the algorithm can be found in [3].

3.3.2 Unsupervised Machine Learning Algorithm Employed

3.3.2.1 K-Means

K-means is known as a nonhierarchical and partitional clustering technique [1]. In K-Means clustering, the number of clusters, which is denoted as *k*, is assumed to be fixed. The algorithm consists of the following basic steps:

- initialize k prototypes ($w_1, \ldots, w_k$)
  where $w_y = x_l, y \in \{1, \ldots, k\}, l \in \{1, \ldots, n\}$
- associate each cluster C_i with prototype w_j
- repeat until the cluster membership no longer changes or there is no significant change in the error quantity. A more detailed explanation of the algorithm can be found in [1].

4. EXPERIMENTS AND RESULTS

In this work, WEKA [20] the open source data mining tool is employed to run C4.5, Ripper, SVM, Naïve Bayesian and K-Means machine learning algorithms. Four datasets are used to evaluate performance of the algorithms on identification of P2P applications inside a network. In the evaluation phase, four metrics as detection rate, false positive rate, F-Measure and correctly classified rate are used to compare effectiveness of the all algorithms.

4.1 Performance Metrics

The definitions of the performance metrics are as below. The used term in the definitions "positive" means label of the network flow which is created by the specific P2P application that we are looking for. And the used term "negative" means label of the network traffic, which is not created by the specific application that we are looking for.

Detection Rate (DR) [6] represents rate of the correctly identified traffic flows. We are looking for high DR value for better effectiveness. It is calculated as in Eq.1:

$$1-\left(\#FN\_Classifications \Big/ \#TotalSpecific\_Application\_Flows\right) \quad \text{(Eq.1)}$$

where #FN_Classifications means flows which are identified as negative but which are positive in real.

False Positive Rate (FPR) [6] represents rate of the flows which are defined as positive but which are negative in real. We are looking for low FPR for better effectiveness. It is calculated as in Eq.2:

$$\left(\#FP\_Classifications \Big/ \#TotalSpecific\_Application\_Flows\right) \quad \text{(Eq.2)}$$

where #FP_Classifications means flows which are identified as positive but which are negative in real.

F-Measure (F1) [4] is a metric that takes harmonic mean of precision and recall to compare and rank the per-application performance of machine learning algorithms. We are looking for high F1 value for better effectiveness. It is calculated as in Eq.3:

$$F1 = 2 \cdot \left(\mathrm{Precision} \cdot \mathrm{Recall} \Big/ \mathrm{Precision} + \mathrm{Recall}\right) \quad \text{(Eq.3)}$$

Correctly Classified Rate (CCR) represents rate of the flows, which are classified correctly. We are looking for high CCR value for better effectiveness. It is calculated as in Eq.4:

$$CCR = \left(\frac{\#TP\_Classifications + \#TN\_Classifications}{\#Total\_Flows}\right) \quad \text{(Eq.4)}$$

where #TP_Classifications means flows which are identified as positive and positive in real, #TN_Classifications means flows which are identified as negative and negative in real.

4.2 Algorithms' Results

In this section, we evaluate the performance of the each algorithm we mentioned before on BitCommetDS, BitTorrentDS, UTorrentDS and NonDS. We investigated each algorithm independently by creating different training and testing datasets as we mentioned before. The results of the algorithms are shown on the tables as following.

Table 2 shows the results of C4.5 algorithm on datasets. As we see on the table, C4.5 performs better on BitcommetDS, BitTorrentDS and NonDS than UTorrentDS if we consider DR metric.

Table 2. C4.5 Results

	C4.5			
	BitCommetDS		UTorrentDS	
	Training	Test	Training	Test
DR	1	0.824	0.942	0.589
FPR	0.01	0.127	0.315	0.674
F1	0.995	0.707	0.835	0.275
CCR	99.53%	86.35%	81.38%	37.85%
	BitTorrentDS		NormalDS	
	Training	Test	Training	Test
DR	0.996	0.991	1	0.93
FPR	0.202	0.401	0.002	0.204
F1	0.906	0.552	0.999	0.677
CCR	89.68%	67.78%	99.90%	82.25%

Table 3 shows the results of Naïve Bayesian Algorithm on datasets. As we see on the table, Naïve Bayesian performs better on BitCommetDS than other data sets if we consider all metrics.

Table 3. Naïve Bayesian Results

	Naïve Bayesian			
	BitCommetDS		UTorrentDS	
	Training	Test	Training	Test
DR	0.998	0.748	0.951	0.845
FPR	0.016	0.04	0.643	0.824
F1	0.991	0.784	0.733	0.329
CCR	99.10%	91.75%	65.40%	30.95%
	BitTorrentDS		NormalDS	
	Training	Test	Training	Test
DR	0.961	0.959	0.6	0.154
FPR	0.457	0.703	0.181	0.187
F1	0.795	0.402	0.674	0.162
CCR	75.20%	42.95%	70.95%	68.13%

Table 4 shows the results of Ripper Algorithm on datasets. As we see on the table, Ripper performs better on BitcommetDS, BitTorrentDS and NonDS than UTorrentDS if we consider all metrics.

Table 4. Ripper Results

	RIPPER			
	BitCommetDS		UTorrentDS	
	Training	Test	Training	Test
DR	0.998	0.748	0.923	0.156
FPR	0.016	0.04	0.34	0.431
F1	0.991	0.784	0.815	0.109
CCR	99.10%	91.75%	79.13%	48.68%
	BitTorrentDS		NormalDS	
	Training	Test	Training	Test
DR	0.984	0.981	0.999	0.993
FPR	0.223	0.387	0.001	0.045
F1	0.891	0.556	0.999	0.914
CCR	88.03%	68.65%	99.9	96.28%

Table 5 shows the results of SVM Algorithm on datasets. As we see on the table, SVM performs better on BitcommetDS, BitTorrentDS and NonDS than UTorrentDS if we consider FPR, F1 and CCR metrics.

Table 5. SVM Results

	SVM			
	BitCommetDS		UTorrentDS	
	Training	Test	Training	Test
DR	0.761	0.761	0.925	0.835
FPR	0.135	0.135	0.527	0.847
F1	0.803	0.737	0.755	0.32
CCR	81.33%	88.58%	69.93%	28.95%
	BitTorrentDS		NormalDS	
	Training	Test	Training	Test
DR	0.965	0.97	0.92	0.986
FPR	0.35	0.711	0.12	0.321
F1	0.834	0.403	0.902	0.603
CCR	80.75%	42.50%	90.00%	74.08%

Table 6 shows the results of K-Means Algorithm on datasets. As we see on the table, K-Means performs better on BitcommetDS, BitTorrentDS and NonDS than UTorrentDS if we consider DR and F1 metrics.

Table 6. K-Means Results

	K-Means			
	BitCommetDS		UTorrentDS	
	Training	Test	Training	Test
DR	0.9245	0.27	1	0.165
FPR	0.6985	0.1253	0.7905	0.1516
F1	0.703	0.305	0.717	0.236
CCR	61.30%	75.38%	60.48%	71.17%
	BitTorrentDS		NormalDS	
	Training	Test	Training	Test
DR	0.9975	0.829	0.6295	0.154
FPR	0.669	0.85	0.8475	0.1544
F1	0.749	0.317	0.506	0.174
CCR	66.43%	71.43%	60.90%	70.72%

These results show that Ripper, SVM and Naïve Bayesian algorithms show close performance on BitCommetDS if we consider DR, F1 and CCR metrics. Then again, we can see that Ripper and Naïve Bayesian algorithms are better with their lower FPR and C4.5 has the best value of DR. UTorrentDS results, on the other hand, show that SVM and Naïve Bayesian algorithms have almost same performance with their close metric values. Both have the highest DR, whereas Ripper algorithm has the highest CCR. Alternatively, BitTorrentDS results show us that C4.5 and Ripper have almost same DR, FPR, F1 and CCR. Additionally, C4.5 and Ripper have highest DR values and CCR while they have lowest FPR values. By looking at the results we can say that C4.5 and Ripper have better performance on BitTorrentDS than other employed algorithms. Finally, NonDS results show us that which algorithm has better performance on identifying P2P and non-P2P network traffic.

All of these outcomes present that Ripper algorithm is performing better at least for one metric between employed algorithms for identifying specific P2P application inside a network. Moreover the features that are used to define network flows as vectors in [6][17] are also gave good results. And this can be a lead for defining specific P2P application inside a network for network management and security purposes.

5. Conclusion and Future Work

The objective of this paper is to maximize the efficiency of flow based classification approach on identification of specific P2P applications inside network traffic by using the machine learning algorithm which provides better performance. We evaluated four supervised machine learning algorithms (C4.5, Ripper, SVM and Naïve Bayesian) and one regularly used unsupervised machine learning algorithm (K-Means). Four different data sets as BitCommetDS, UTorrentDS, BitTorrentDS and NonDS are employed for the evaluation of used algorithms. Based on the DR(Detection Rate), FPR(False Positive Rate), F1(F-Measure) and CCR(Correctly Classification Rate) evaluation metrics, Ripper algorithm performs best in identifying P2P network traffic and it is performing better at least for one metric between employed algorithms for identifying specific P2P application inside a network. We believe that these results can give a support for specific P2P application identification inside a network for network management and security tasks.

Future work should investigate optimization of P2P application identification results by using another algorithms such as SOM, Fuzzy C-Means, improving current algorithms or developing a new algorithm. In addition, the FPR was high for the training and testing for UTorrentDS and BitTorrentDS data sets for most the DM algorithms. Generating a bigger dataset and implementing these algorithms over this set may help us overcome these drawbacks. Finally, other traffic analysis approaches can be applied to see their effectiveness on this task.

6. REFERENCES

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

[2] Burges, C. J. C. 1998. A Tutorial on Support Vector Machines for Pattern Recognition, Data Mining and Knowledge Discovery, Volume 2, pp. 1-47.

[3] George, H. and Langley, P. 1995. Estimating Continuous Distributions in Bayesian Classifiers, Proceedings of the Eleventh Conference on Uncertainty in Artificial Intelligence. pp. 338-345, Morgan Kaufmann, San Mateo.

[4] Hyunchul, K., Claffy, K. C., Fomenkov, M., Barman, D., Faloutsos , M., and Lee, K. 2008. Internet traffic classification demystified: myths, caveats, and the best practices, Proceedings of the 2008 ACM CoNEXT Conference, p.1-12, Madrid, Spain.

[5] Liu, F., Li, Z., and Nie, Q., 2009. A New Method of P2P Traffic Identification Based on Support Vector Machine at the Host Level, International Conference on Information Technology and Computer Science, ITCS 2009, Volume 2, pp. 579-582, Kiev, Ukraine.

[6] Alshammari, R., and Zincir-Heywood, A.N., 2009. Machine learning based encrypted traffic classification: Identifying SSH and Skype, IEEE Symposium on Computational Intelligence for Security and Defense Applications, *CISDA 2009,* pp.1-8, Ottawa, Canada.

[7] Wang, J., Zhang, J., and Tan, Y, 2011. Research of P2P traffic identification based on traffic characteristics, International Conference on Multimedia Technology, ICMT 2011, pp.5032-5035, Hangzhou, China.

[8] Liu, T., and Chen, X., 2011. A novel approach to detect P2P traffic based on program behavior analysis, International Conference on Electrical and Control Engineering, ICECE 2011, pp.5677-5680, Yichang, China.

[9] Soysal, M., and Schmidt, E.G., 2007. An accurate evaluation of machine learning algorithms for flow-based P2P traffic detection, 22nd International Symposium on Computer and Information Sciences, ISCIS 2007, pp.1-6, Ankara, Turkey.

[10] Kun, L., Wei, G., 2010. Feedback model based on P2P traffic control, International Conference on Computational Problem Solving, ICCP 201, pp.35-38, Lijiang, China.

[11] Guanghui, H., Hou, J., Chen, W. P., Hamada, T., 2007, One Size Does Not Fit All: A Detailed Analysis and Modeling of P2P Traffic, Global Telecommunications Conference, IEEE GLOBECOM 2007, pp.393-398, Washington, US.

[12] Chunzhi, W., Wei., J., Hong, C., Luo, W., Fang, H., 2010. Research on a method of P2P traffic identification based on multi-dimension characteristics, 5th International Conference on Computer Science and Education, ICCSE 2010, pp.1010-1013, Hefei, China.

[13] BitCommet P2P file sharing software, http://www.bitcomet.com/

[14] BitTorrent P2P file sharing software, http://www.bittorrent.com/

[15] UTorrent P2P file sharing software, http://www.utorrent.com/

[16] Netmate network measurement tool, http://www.ip-measurement.org/tools/netmate

[17] Alshammari, R., Zincir-Heywood, A. N., 2010. An investigation on the identification of VoIP traffic: Case study on Gtalk and Skype, International Conference on Network and Service Management, CNSM 2010,pp.310-313, Niagara Falls, Canada.

[18] Calculating Flow Statistics using NetMate: http://dan.arndt.ca/nims/calculating-flow-statistics-using-netmate/

[19] Quinlan, J.R., 1993. C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, ISBN:1-55860-238-0.

[20] Weka software http://www.cs.waikato.ac.nz/ml/weka