Active Build-Model Random Forest Method for Network Traffic Classification

Alhamza Munther [#1], Rozmie Razif [#2], Shahrul Nizam [#3], Naseer Sabri [#4], Mohammed Anbar [*5]
[#1, 2, 3, 4] School of Computer and Communication Engineering, Universiti Malaysia Perlis, Perlis, Malaysia
[1] alhamza80@yahoo.com
[2] rozmie@unimap.edu.my
[3] shahrulnizam@unimap.edu.my
[4] naseersabri@yahoo.com
[*5] National Advanced IPv6 Centre of Excellence, Universiti Sains Malaysia, Penang, Malaysia
[5] anbar@nav6.org

Abstract— Network traffic classification continues to be an interesting subject among numerous networking communities. This method introduces multi-beneficial solutions in different avenues, such as network security, network management, anomaly detection, and quality-of-service. In this paper, we propose a supervised machine learning method that efficiently classifies different types of applications using the Active Build-Model Random Forest (ABRF) method. This method constructs a new build model for the original Random Forest (RF) method to decrease processing time. This build model includes only the active trees (i.e., trees with high accuracy), whereas the passive trees are excluded from the forest. The passive trees were excluded without any negative effect on classification accuracy. Results show that the ABRF method decreases the processing time by up to 37.5% compared with the original RF method. Our model has an overall accuracy of 98.66% based on the benchmark dataset considered in this paper.

Keyword-Network Traffic Classification, Machine learning, Supervised Learning, Random Forests Algorithm

I. INTRODUCTION

Network traffic identification and classification have recently gained considerable interest as an important network engineering tool for network security, network design, as well as network monitoring and management. The importance of traffic classification is based on the increasing size of transferred data and the variety of applications facilitated to understand the structure and the dynamics of networks, flow prioritization, and diagnostic monitoring. Most network administrators and ISPs also classify the applications to assist in diagnosing high resource consumption (i.e., CPU, memory, and bandwidth) and avoid exploits by attackers. For example, a network engineer may want to identify and control (e.g. blocking) peer-to-peer (P2P) application traffic to free bandwidth and make it available for other applications. The good performance of business-critical applications is also ensured because P2P applications occupy a high fraction of network bandwidth [1]. Likewise, network management tasks benefit from the accurate identification of network traffic to manipulate different types of network engineering problems, such as workload characterization and modeling, and route provisioning. The identification and classification of applications also enable both network operators and end users to detect anomalies in terms of data security and service availability. Anomalies are unusual traffic behavior that can cause significant changes in a network's traffic levels, such as those produced by worms or Denial of Service attacks.

Numerous approaches in application identification and classification have been proposed in literature, but some have legal problems. Well-known ports are no longer an answer because many applications, especially those with a high network volume (e.g., P2P file sharing and multimedia streaming), bypass the rules and use the known ports of other services. Another well-researched approach is inspecting packet payloads [2, 3]. In this approach, the packet payloads are analyzed to see whether they contain characteristics signatures of known applications. Many of studies moved toward machine learning algorithms to automate the predication of an application. Some of these algorithms belong to supervised or unsupervised machine learning. The most popular methods proposed [4, 5] are Naive Bayes which is based on the Bayesian network, employed K-means clustering [6], used Support Vector Machine (SVM) [7], and suggested C4.5 decision tree [8]. Random Forest (RF) is constructed by a series of decision tree-based classifiers. Each base classifier takes a different training set of x inputs that are drawn independently with replacement of the original training set of x inputs. The two main motivations in adopting the RF method are the potential to deal and classify huge learning data, and the capability to construct a multi-classifier in a parallel manner. Once we adopted the original RF method, some trees do not occupy a significant role in terms of classification accuracy (passive trees). In this paper, we address eliminating the passive trees from the forest. We propose the Active Build-Model Random Forest (ABRF) method to decrease the processing time in the original RF method in classifying network traffic with regard to accuracy classification. Optimization is achieved by decreasing the size of the forest, which is, in turn, achieved

by excluding low performance trees in terms of accuracy. This reduction is governed by specific conditionals, which do not negatively influence the overall classifier accuracy. Section 3 explores the details of this method.

This paper is organized as follows. Section 2 discusses some of the existing studies that review the most popular studies in classifying network traffic. Section 3 discusses the proposed ABRF as a classification method in detail, followed by the experiment platform. Section 5 highlights the evaluation of the proposed technique. Finally, Section 6 concludes this paper.

II. RELATED WORKS

Many approaches that have been proposed to address the accuracy problem of identifying and classifying network traffic. The commonly used approaches are reviewed as follows.

A. Port-Based Approach

The traditional approach to traffic classification relies on mapping applications to well-known port numbers (assigned by IANA [1]). This approach has been very successful because many traditional applications use fixed port numbers in the past. The main advantage of the port-based method is being fast as it does not require any complex computation. This method matches the application port with a well-known port number. For example, email applications commonly use port 25, HTTP traffic uses port 80, FTP uses port 21, and DNS uses port 53. Although port-based traffic classification is the fastest and simplest method, several studies show it to be ineffective [2, 3],[4]. The accuracy of the port-based method deteriorates abruptly with the emergence of some applications such as P2P. These applications attempt to evade or hide its behaviour from firewalls and network security tools using dynamic port numbers as HTTP or FTP applications. Several traffic applications are encrypted with used proxies, VPN, and tunneling. Some other applications use the same well-known ports that are unassigned, such as VoIP telephony, chat messenger systems, and web-page browsing. In the end, some studies [5, 6] report that only 50% to 70% of the Internet traffic is classifiable. Thus, the port-based method is no longer reliable.

B. Payload-Based Approach

The most accurate payload packet inspection is proposed to address the aforementioned drawbacks of port-based classification. In this approach, the complete packet payloads are analyzed. However, parsing all packets for all users separately is computationally complex for this method. Some countries also have privacy laws prohibiting network operators from analyzing the entire payload of network packets to avoid hidden sniffer programs. One way to decrease resource consumption for traffic classification is to search for specific byte patterns (signatures) in all or part of the packets. Predefined byte signatures are used to identify particular traffic types, such as for BitTorrent traffic (‘.\BitTorrent’), Gnutella protocol (.\‘GNUTELLA’), web traffic ('\GET'), and eDonkey P2P traffic (‘\xe3\x38’). Some researchers have proposed automated ways to identify signatures [7, 8]. They have implemented the automated schemes only on popular applications, such as FTP, SMTP, HTTP, HTTPS, SSH, DNS, and NTP, and not on newer applications such as P2P, games, and streaming. In general, payload-based classification achieves good results in terms of accuracy, but it is resources exhaustive, expensive, computationally complex and unscalable, does not work on encrypted traffic, and causes tremendous privacy and legal concerns.

C. Behaviour-Based Approach

Network traffic is an important carrier that records and reflects Internet and user activities. This parameter is also an important composition of network behavior. Through the analysis of traffic indicators, we can master network statistical behavior directly. This approach studies the behavior of each application flow independently and attempts to find discriminate by extracting the common behavioral information of the hosts (e.g., BLINC) [9].

D. Machine Learning Algorithms

Machine learning is a branch of artificial intelligence that learns systems that possess a dynamic environment; it is concerned with the design and development of algorithms that allow computers to evolve behavior based on empirical data [10, 11]. The best aspects of machine learning that has led many researchers to choose it for network traffic classification are its powerful capability to automate classifying traffic, deal with a diversity of applications, and to deal with large amounts of data. The main structure for machine learning algorithms consists of two phases. These phases are the building model and the classification. A building model is first constructed using training data, and then later used to predict the class membership of new instances by examining the feature values of unknown flows. This model is then entered into a classifier that classifies the entire dataset. Generally, machine learning algorithms are categorized into supervised and unsupervised learning Fig. 1. Supervised learning requires training data to be labeled in advance (i.e., the class of each traffic flow must be known before learning) and yields a model that fits the training data. The advantage of these algorithms is that they can be tuned to detect subtle differences and can clearly label the flows upon termination, unlike the unsupervised ones. An example for supervised learning is Naive Bayes [12, 13], the simplest probabilistic

classifier based on Bayes' theorem [14], which analyzes the relationship between each feature and the application class for each instance. This approach derives a conditional probability for the relationships between the feature values and the class. The Bayesian network is used a directed acyclic graph model that represents a set of features (or classes) as its nodes and their probabilistic relationship as edges. If the conditional independence assumption is not valid, the Bayesian Network learning may outperform Naive Bayes. Nigel et al. [12] and Fu et al. [15] suggested the C4.5 decision tree its built the model based on tree structure, in which each internal node represents a test on features, each branch represents an outcome of the test, and each leaf node represents a class label. Many researchers are used Support Vector Machines (SVM [6, 16, 17]. The basic principle of SVM is to construct an optimal separating hyperplane using binary labeled training data, which maximizes the margin between two classes on either side. SVM has been proven to decrease an upper bound on the expected generaliation error. Unsupervised algorithms are able to cluster traffic flows into different classes according to similarities in their statistical properties (feature values). Unlike supervised learning, the model is not provided with the correct results during the training phase; the algorithm itself determines the group numbers for each instance based on statistical information, such as K-means (unsupervised) [18]. This approach takes statistical information as an input vector to build the classification models (or classifiers). The K-means clustering algorithm is a simple and popular analysis method. The basic idea is that it starts with a training set and an assigned number of clusters (k) to be found. The items within a training set are assigned to a cluster according to a similarity measurement distance, for example, using Euclidean distance to estimate similarity. Jeffrey et al. [4] applied the Expectation Maximization (EM) clustering technique called AutoClass to determine the most probable set of clusters from the training data. Authors calculate the probability of an object being a member of each discrete cluster using afinite mix ture model of the attribute values for the objects belonging to the cluster. This approach 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.

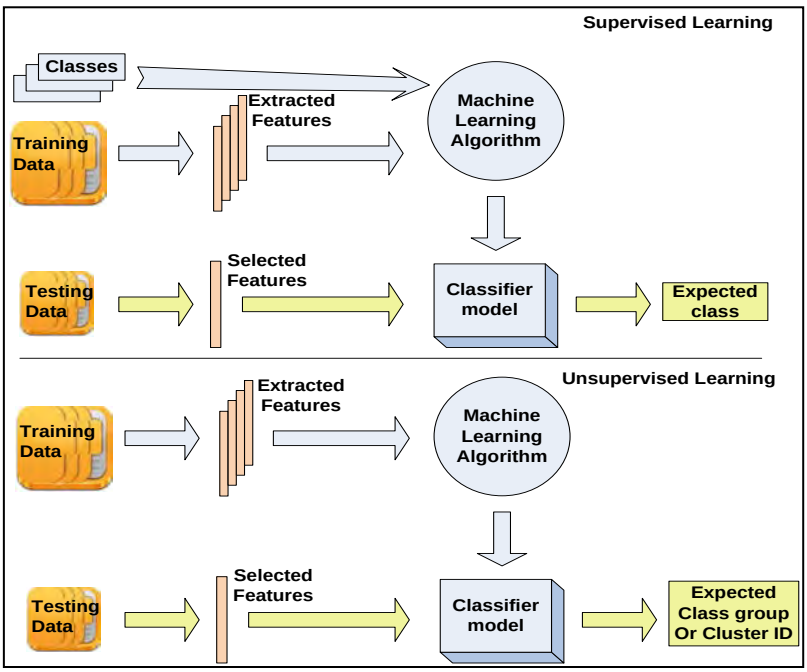

Fig.1. Supervised and Unsupervised learning paradigm

III. Classification Approach Using ABRF

The RF algorithm is a classification method developed by Leo Breiman [22]. The forest contains a crew of classifiers (binary trees). Some of the trees are active or passive in terms of identification and classification the accuracy of network traffic. In this section, we elaborate the potential of removing the passive trees aiming to diminish the forest size and decrease the processing time of classification accuracy. Fig. 2 shows the major phases of our proposed ABRF method.

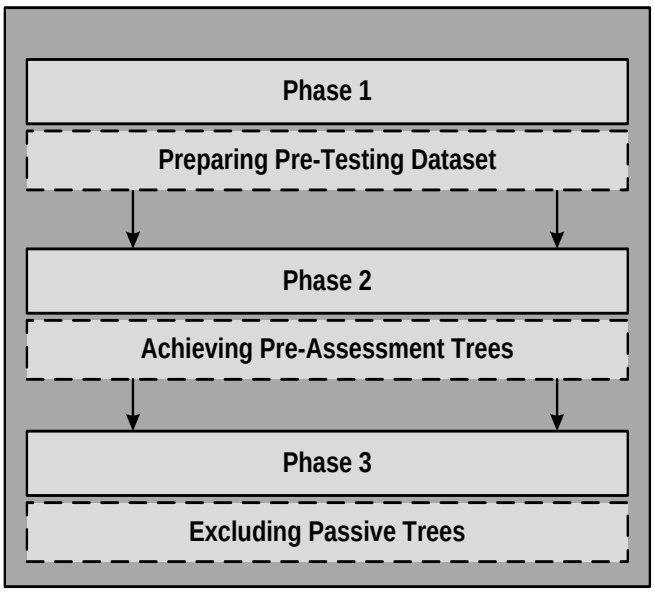

Fig. 2. Phases of the proposed method

A. Phase 1: Preparing the Pre-Testing Dataset

In this phase, we select a data subset from the original dataset, which includes different instances with dissimilar types of traffic application. The instances are selected randomly to construct the pre-testing dataset. The number of instances ranges from 2 to 5 per class depending on the number of classes available in the original dataset. We try to keep the size of this data as small as possible to avoid extra processing time. These instances carry the same features that are subsequently used to construct the trees, whereas the other features are ignored. The set of features are selected randomly. Pre-testing dataset will be used to select the set of omitting trees as elaborated in the next steps.

B. Phase 2: Achieving the Pre-Assessment Tree

After the pre-testing dataset is determined, the same procedure for the original RF is employed to build the model algorithm constructed with a group of trees. The pre-testing dataset is then passed to the RF building model for voting and testing for each instance. The entire pre-testing dataset is classified by each single tree. The accuracy of the instances is measured using the standard formula for margin function $mg(X,Y)$. The values are then passed to the next step to make decision for eliminating trees. The margin function measures the distance between the average numbers of correct votes for a specific instance and the average number of votes of any other class for the same instance. The larger the margin is, the more accurate the classification becomes. Mathematically, we can express the margin function by assuming that the forest consists of Kth trees (classifiers) ($k = 1, 2, 3 \ldots k$). Thus:

$$mg(X,Y) = avg_c I(tree_k(X) = Y) - max_{S \neq Y} avg_c I(tree_k(X) = J)$$

where $I(.)$ is an indicator function, X is a labeled instance (input) having a class Y, and $tree_k(X)$ is the vote class of $tree_k$ for the instance X.

C. Phase 3: Excluding Passive Trees

After voting for the pre-testing dataset occurs and the accuracy for each instance is determined, we select the set of trees with higher accuracy (active trees), whereas trees with low accuracy (passive trees) are excluded from the forest. In this way, yield a new build model for RF algorithm (Active-Build model (ABRF)) which only involves active trees. The Active-Build model will be used to train and test the entire original dataset instead of the previous one. Fig. 3 summarizes the major steps in constructing the ABRF model.

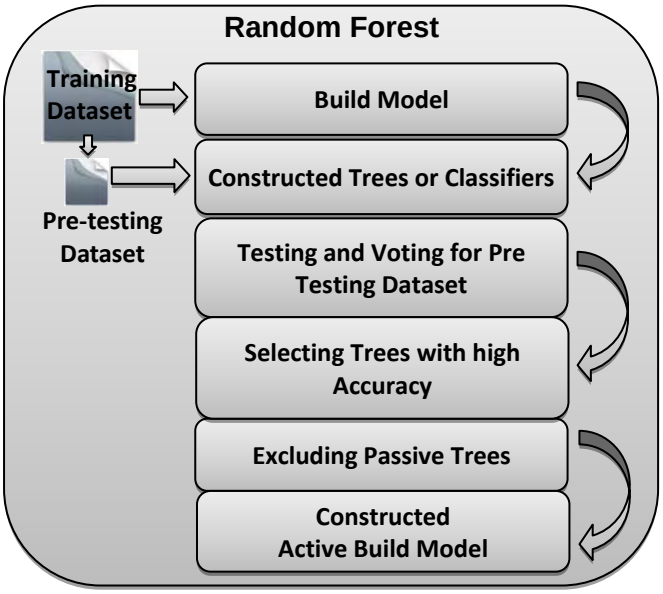

Fig. 3. Steps for constructing the Active-Build model

IV. EXPERIMENT PLATFORM

In this paper, the experiments were conducted using WEKA version 3.7.10 , which is an open source software for machine learning implemented using the Java language and developed at the University of Waikato, New Zealand. We also adopted the same dataset of [3] consisting of 24863 instances, 248 attributes, and 11 classes. This dataset includes WWW, MAIL, FTP-CONTROL, FTP-PASV, ATTACK, P2P, DATABASE, FTP-DATA, MULTIMEDIA SERVICES, and INTERACTIVE. These data were collected from the edge of a network. It allows access to all packets associated with a TCP connection in both directions (from sender to receiver and vice-versa). As a result, we can obtain more features for each packet Fig. 4.

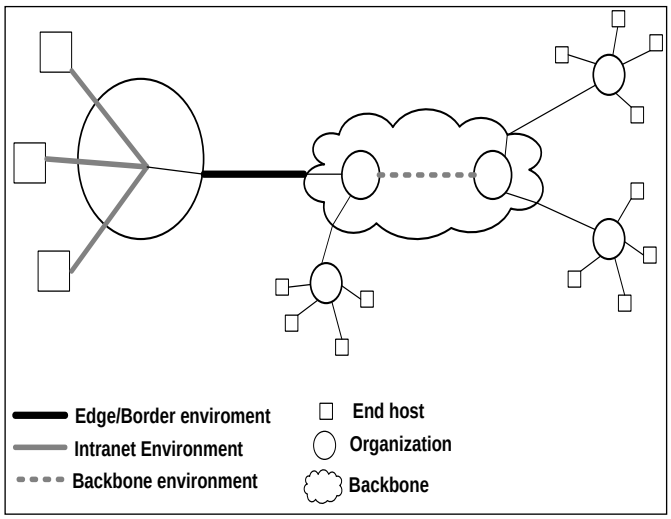

Fig. 4. Dataset environment

V. EVALUATION AND RESULTS ANALYSIS

This section evaluates two important parameters, namely, accuracy of classification and processing time for the proposed method. We elaborate the impact of each parameter individually as follows:

A. Classification Accuracy

We measure the standard metrics to assess the performance of the constructed classifier using the RF algorithm. These metrics are true positives, false positives, precision recall, and F-Measure. We can define each metric as follows:

True Positives (TP): The percentage of the Radom Forest algorithm correctly classified as belonging to class C.

False Positives (FP): The percentage of members of other classes incorrectly classified as belonging to class C.

Precision: The percentage of the objects that truly have class C among all those classified as class C.

Recall*:* The percentage of class C members correctly classified as belonging to class C.

F-Measure: The combination of precision and recall.

Table I and its bar plotting in Fig. 5 show the experiment results for the abovementioned metrics. The strength of the RF algorithm to classify some types of applications is shown. The TP percentages were between 84% and 99% for WWW, MAIL, FTP-CONTROL, DATABASE, FTP-DATA, and SERVICES. The results show a form of weakness in classifying another application, such as FTP-PASV, MULTIMEDIA, and INTERACTIVE, where the TP decreased by 56% to 80% due to some applications attempt to hide its identification by changing some statistical features. The worst TP ratio was 58.2% for ATTACK. We noted that FP percentages for most classes have low values (0.016% and 0.001%). The high percentage appears in the web application because of the variety of web browsing applications, and then fades out with the rest of the classes.

Table II and its bar plotting in Figure 5 highlights two important metrics, namely, precision and recall for both the original RF and ABRF models. The results reveal a substantial convergence between the RF and ABRF models for WWW, MAIL, FTP-CONTROL, DATABASE, FTP-DATA SERVICES, and INTERACTIVE, whereas a slight difference was observed for the other applications. Obviously, original RF model clearly shows a weakness in classifying some applications, such as ATTACK, MULTIMEDIA, and DATABASE. The overall accuracy of the original RF is 98.89%, whereas that of ABRF reached 98.66%.

TABLE I.
Accuracy parameters for ABRF

Class	TP	FP	Precision	Recall	F-Measure
WWW	99.4	0.016	99.4	99.4	99.4
MAIL	98.8	0.003	98.7	98.8	98.7
FTP-CONTROL	89.3	0	95	89.3	92
FTP-PASV	72.1	0.001	64.6	72.1	68.1
ATTACK	58.2	0.002	54.2	58.2	56.1
P2P	82.6	0.002	86.4	82.6	84.3
DATABASE	97.9	0	95.9	97.9	96.9
FTP-DATA	99.7	0	99.8	99.7	99.7
MULTIMEDIA	78.2	0.001	75.6	78.2	76.8
SERVICES	100	0	97.2	100	98.6
INTERACTIVE	66.7	0	100	66.7	80

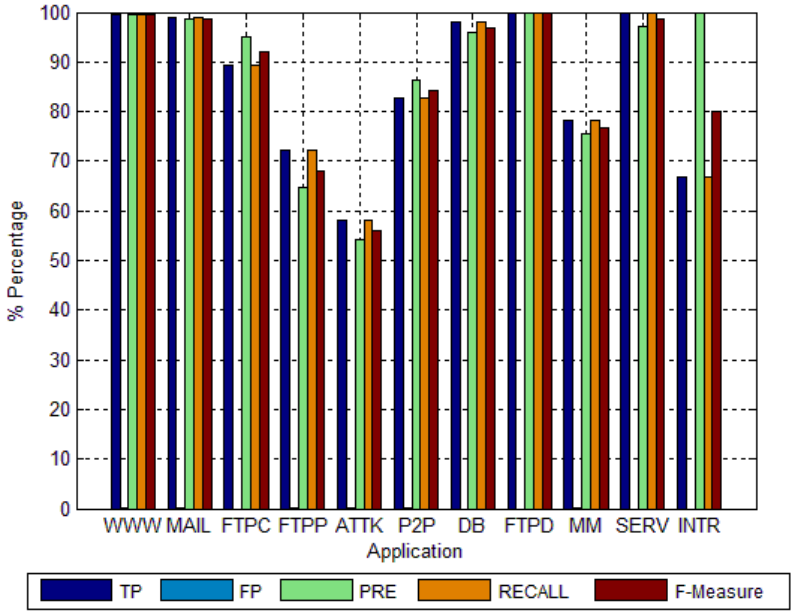

Fig. 5. Different parameters for ABRF accuracy classification

TABLE 2
Recall and Precision for both RF and ABRF

CLASS	ABRF PRECISION	RF PRECISION	ABRF-RECALL	RF-RECALL
WWW	99.4	99.2	99.4	99.9
MAIL	98.7	98.7	98.8	99.1
FTP-CONTROL	95	95.6	89.3	87.2
FTP-PASV	64.6	67.6	72.1	53.5
ATTACK	54.2	81.9	58.2	55.7
P2P	86.4	92.5	82.6	80.2
DATABASE	95.9	99.6	97.9	95
FTP-DATA	99.8	99.5	99.7	99
MULTIMEDIA	75.6	84.8	78.2	64.4
SERVICES	97.2	100	100	99.5
INTERACTIVE	100	100	66.7	70.1

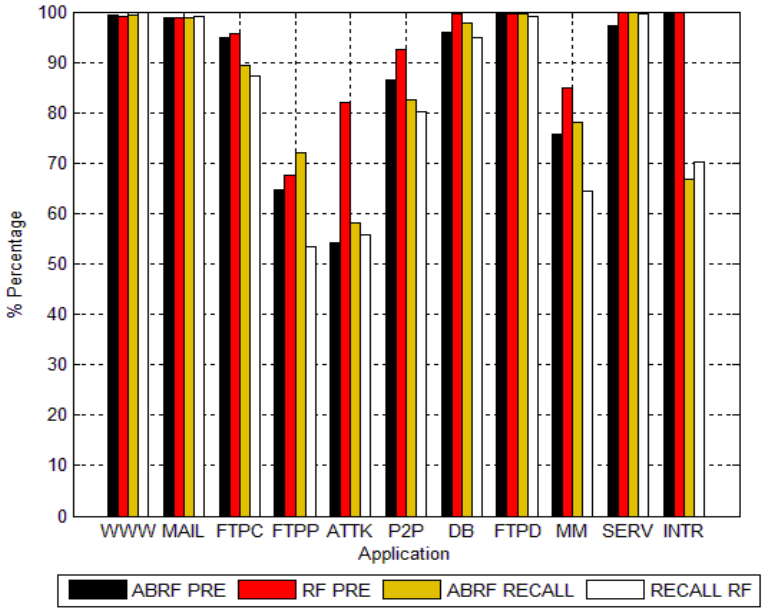

Fig. 6. Classification accuracy for both ABRF and the original RF models

B. Time of Processing

Most machine learning algorithms used for classifying network traffic focus on measuring the accuracy of algorithm classification. Besides accuracy, significant factors need to be considered to validate the powerful classification methods particularly for online classification (i.e., processing time memory consumption and CPU consumption). In this paper, we measure the processing time for the original RF and ABRF models, where eliminating passive trees from the forest influences the processing time. Fig. 7 shows the processing time results for the modified RF model compared with the original RF model. Processing time is measured for different sizes of the dataset. Notably, the required time to classify 1000 instances in the original RF was 0.848 sec, which then increased to 10.256 sec for 24000 instances. In contrast, the processing time for the ABRF model decreased to 0.53 sec for 1000 instances and 6.41 sec for 24000 instances. Excluding the number of trees from the entire forest clearly has a positive effect on the processing time. Our results show that the time decreases by 37.5% compared with the original RF.

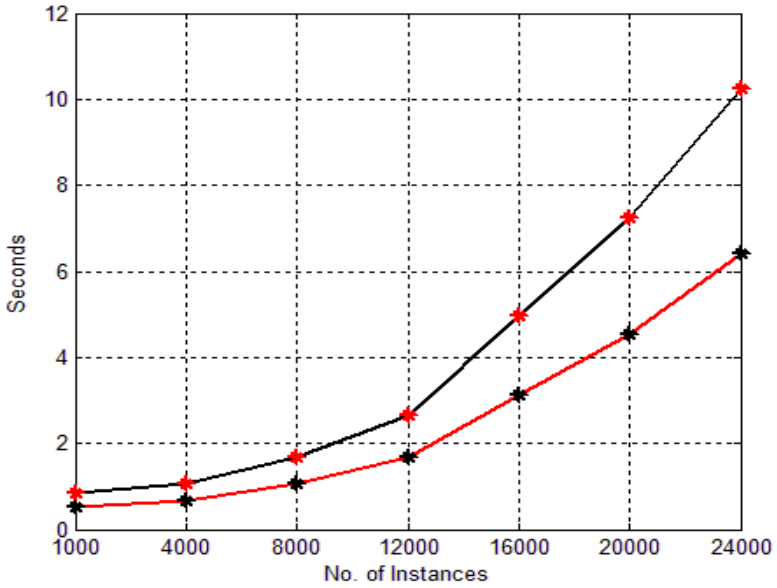

Fig. 7. Processing time for both the ABRF and original RF models

VI. CONCLUSION

The increasing data volume and the variety of traffic applications in networks were the motivations behind identifying applications for different networks types. In this paper, we optimized the supervised RF method to classify network traffic. Optimization is achieved by constructing a new building model that excludes passive trees from the forest in the building model stage in terms of classification accuracy. The more relevant trees to a certain class were then selected. The experiment results for the ABRF model reveal that the processing time decreased by 37.5% compared with the original RF model. The ABRF model had an overall accuracy of 98.66%.

Our immediate next step is to improve its classification accuracy for other applications. We also intend to apply the proposed method to classify UDP protocol applications and implement this method on huge datasets to recognize the effects of different factors in terms of quality-of-service.

VII.ACKNOWLEDGMENT

This research is partially funded by the generous fundamental grants – “Fundamental study on the effectiveness measurement of partially executed t-way test suite for software testing” from Ministry of Higher Education (MOHE), and the UniMAP short term grant.

REFERENCES

[1] A. Callado, C. Kamienski, S. n. Fernandes, and D. Sadok, "A Survey on Internet Traffic Identification and Classification," 2009.
[2] T. Karagiannis, A. Broido, and M. Faloutsos, "Transport layer identification of P2P traffic," in Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, 2004, pp. 121-134.
[3] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of p2p traffic using application signatures," in Proceedings of the 13th international conference on World Wide Web, 2004, pp. 512-521.
[4] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification," ACM SIGCOMM Computer Communication Review, vol. 36, pp. 5-16, 2006.
[5] A. W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," in ACM SIGMETRICS Performance Evaluation Review, 2005, pp. 50-60.
[6] L. Yingqiu, L. Wei, and L. Yunchun, "Network traffic classification using k-means clustering," in Computer and Computational Sciences, 2007. IMSCCS 2007. Second International Multi-Symposiums on, 2007, pp. 360-365.
[7] Z. Li, R. Yuan, and X. Guan, "Accurate classification of the internet traffic based on the svm method," in Communications, 2007. ICC'07. IEEE International Conference on, 2007, pp. 1373-1378.
[8] L. FU, B. TANG, and D. YUAN, "The Study of Traffic Classification Methods Based on C4. 5 Algorithm," 2012.
[9] IANA: Internet Assigned Numbers Authority.
[10] H. Dreger, A. Feldmann, M. Mai, V. Paxson, and R. Sommer, "Dynamic application-layer protocol analysis for network intrusion detection," in USENIX Security Symposium, 2006, pp. 257-272.
[11] A. W. Moore and K. Papagiannaki, "Toward the accurate identification of network applications," in Passive and Active Network Measurement: Springer, 2005, pp. 41-54.
[12] J. Erman, A. Mahanti, and M. Arlitt, "Qrp05-4: Internet traffic identification using machine learning," in Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE, 2006, pp. 1-6.
[13] W. Li and A. W. Moore, "A machine learning approach for efficient traffic classification," in Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, 2007. MASCOTS'07. 15th International Symposium on, 2007, pp. 310-317.
[14] H. Kim, K. C. Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee, "Internet traffic classification demystified: myths, caveats, and the best practices," in Proceedings of the 2008 ACM CoNEXT conference, 2008, p. 11.
[15] P. Haffner, S. Sen, O. Spatscheck, and D. Wang, "ACAS: automated construction of application signatures," in Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data, 2005, pp. 197-202.

[16] J. Ma, K. Levchenko, C. Kreibich, S. Savage, and G. M. Voelker, "Unexpected means of protocol inference," in Proceedings of the 6th ACM SIGCOMM conference on Internet measurement, 2006, pp. 313-326.
[17] T. Karagiannis, K. Papagiannaki, and M. Faloutsos, "BLINC: multilevel traffic classification in the dark," in ACM SIGCOMM Computer Communication Review, 2005, pp. 229-240.
[18] B. S. Clarke, E. FokouÃ©, and H. H. Zhang, Principles and theory for data mining and machine learning: Springer, 2009.
[19] T. M. Mitchell, "Machine learning. WCB," McGraw-Hill Boston, MA:, 1997.
[20] T. J. Fagan, "Letter: nomogram for Bayes theorem," The New England journal of medicine, vol. 293, pp. 257-257, 1975.
[21] K. P. Bennett and C. Campbell, "Support vector machines: hype or hallelujah?," ACM SIGKDD Explorations Newsletter, vol. 2, pp. 1-13, 2000.
[22] L. Breiman, "Random forests," Machine learning, vol. 45, pp. 5-32, 2001.