Performance Evaluation 67 (2010) 451–467

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison

Murat Soysal [a], Ece Guran Schmidt [b,*]

[a] Turkish Academic Network and Information Center, TÜBİTAK-ULAKBİM, Ankara, Turkey

[b] Department of Electrical and Electronics Engineering, Middle East Technical University, Ankara, Turkey

ARTICLE INFO

Article history:
Received 1 February 2009
Received in revised form 14 October 2009
Accepted 4 January 2010
Available online 11 January 2010

Keywords:
Traffic classification
Privacy-preserving classification
Supervised machine learning
Data set composition
Comparison

ABSTRACT

The task of network management and monitoring relies on an accurate characterization of network traffic generated by different applications and network protocols. We employ three supervised machine learning (ML) algorithms, Bayesian Networks, Decision Trees and Multilayer Perceptrons for the *flow-based* classification of six different types of Internet traffic including peer-to-peer (P2P) and content delivery (Akamai) traffic. The dependency of the traffic classification performance on the amount and composition of training data is investigated followed by experiments that show that ML algorithms such as Bayesian Networks and Decision Trees are suitable for Internet traffic flow classification at a high speed, and prove to be robust with respect to applications that dynamically change their source ports. Finally, the importance of correctly classified training instances is highlighted by an experiment that is conducted with wrongly labeled training data.

1. Introduction

The introduction of new services, and the great demand on bandwidth, increase the importance of network traffic engineering. Network traffic engineers' tasks include meeting bandwidth requirements of customers or services, managing the bandwidth consumption of the users if necessary, applying security rules and performing accurate accounting for billing issues. A broad understanding of network traffic properties is thus essential to both address the current needs and to develop new architectures that help improving the network performance. An important tool to accomplish these tasks is *traffic classification*. The approaches for traffic classification proposed in both the academic literature and in the practical field include *port-based methods*, *payload inspection-based methods* and *flow-based methods*.

Transport protocol ports were the initial traffic classification criteria. However, for example, P2P applications [1] hide themselves by issuing dynamic ports including the port numbers reserved for well-known protocols by IANA (Internet Assigned Numbers Authority [2]). The significant amount of bandwidth consumed by P2P applications, combined with their particular traffic characteristics and issues such as distribution of the copyrighted material, attracted attention to the detection of P2P traffic. Similarly, the FTP protocol can dynamically assign ports according to the traffic load. Hence, the success rate of port based identification is limited.

Currently, a very popular traffic classification method is payload inspection-based classification, due to its relatively high accuracy. Since it employs deep packet inspection, violation of user data privacy, and the amount of data processed are the major disadvantages of this method. The success rate of payload inspection-based traffic classification depends on the specific tool that is used. In addition, payload inspection-based classification of traffic with encrypted data is impossible.

* Corresponding author. Tel.: +90 312 210 4405; fax: +90 312 210 2304.
E-mail addresses: msoysal@ulakbim.gov.tr (M. Soysal), eguran@metu.edu.tr (E.G. Schmidt).

0166-5316/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.peva.2010.01.001

Traffic classification by using *flow traces* is proposed as an alternative to payload inspection-based methods. In this approach, flow traces that summarize the characteristic *features* of network flows, are collected only from the packet headers [3]. Hence, neither the amount of data to be analyzed nor potentially encrypted data constitute a problem, while user data privacy is preserved. As a potential disadvantage of this method, the limited information that can be used for classification can lead to inaccuracies.

A powerful tool for generating general hypotheses from available data (such as network flow traces) in order to classify unknown data is *supervised machine learning* [4]. This method is employed for the classification of network flows by a number of published works [5–9]. The *classification accuracies* of supervised ML algorithms are evaluated by applying them to test data sets.

There are two main issues related to such evaluation approaches for ML algorithms. First, the flows in the test data set must be correctly *labeled* before the evaluation, since the results of the evaluations can only be as correct as the labeling of the test data flows. That is, traffic classes of the flows in the test data must be previously known. In most of the previous studies this test data are either compiled from publicly available repositories or labeled using payload inspection-based or port-based tools. Considering that the identification techniques for publicly available data sets are either not declared or based on port-based or payload inspection-based classification, the resulting inaccuracies directly affect the evaluation results. The second issue is due to the fact that supervised ML require *training data* to characterize the different traffic classes. Particularly, the previous work on ML, which is mainly based on supervised learning, employs training data. The training data set is composed similarly to the test data set, which brings up the same issues related to the accuracy of the training data set. There is no detailed study on the impact of this incorrect labeling on the accuracy of the ML algorithms. Furthermore the possible correlations among the characteristic features of the traffic classes in the training data set can affect the correctness of the classification. The previous studies that evaluate ML algorithms for flow-based traffic classification consider the total number of flows in the training data sets. However, the number of flows from each traffic type is mostly based on the availability of the flow traces.

In this paper, a systematic approach for investigating and evaluating the classification performance of three supervised Machine Learning (ML) algorithms, *Bayesian Networks* (BNs) [10], *Decision Trees* (DTs) [11] and *Multilayer Perceptrons* (MLPs) [4], using flow traces is presented. The accuracy of our evaluation is achieved by featuring a large amount of correctly labeled, recent flow traces from the National Academic Network of Turkey. These flow traces are then used to assemble a large number of custom-made test and training data flows that constitute the basis for our studies.

In a first set of experiments, we employ the collected flow data to systematically investigate the impact of the amount and the ratio of flows from each traffic type in the training data. As a result, we identify the *correlations* between the different protocols that affect their respective classification performance, which allows us to construct an *ideal* training data set which yields the best classification results for each of the ML algorithms.

In a second set of experiments, we investigate three variations of the traffic classification problem using BNs and DTs as ML algorithms. First, we study the robustness of the ML algorithms with respect to scenarios where applications generate flows that have ports different from their default ports. Second, we highlight the significance of supplying correctly labeled training data by carrying out experiments with (knowingly) incorrectly labeled training data, and observing the degradation of the resulting classification performance. Finally, we demonstrate the impact of incorrectly labeled training and testing data on the reported accuracy of the ML algorithms.

The organization of the paper is as follows. Section 2 summarizes techniques for traffic flow data collection and traffic classification. Our experimental methodology along with our reliable data acquisition approach is presented in Section 3, followed by a detailed investigation of BNs, DTs and MLPs for traffic classification in Section 4. Section 5 gives conclusions.

2. Techniques for traffic classification

Different techniques have been proposed for the classification of Internet traffic. Particularly, P2P traffic detection is an active area of research, since this kind of traffic constitutes a significant fraction of the overall Internet traffic, and there are specific issues such as legal problems related to the data shared over P2P applications.

From the technical perspective, traffic classification techniques proceed in two steps. First, a traffic analysis is performed to extract certain traffic *features* and their respective values that characterize the different traffic classes, or in other words form a *signature* (or *fingerprint* [12]) for each traffic class. Then, unknown traffic can be classified by evaluating its feature values against the characterizing features obtained in the analysis step.

The features that are used in the classification process can be extracted from the *packet payload*, *packet headers* or *traffic flows*, where a traffic flow denotes a unidirectional stream of packets between a given IP address pair.

In the following sections, we first define the performance metrics for traffic classification and then investigate the traffic classification techniques that are grouped according to their source of feature extraction.

2.1. Performance metrics for traffic classification techniques

In principle, there are two types of *performance metrics* that measure the quality of traffic classification algorithms. They capture both the *correctness* and the *computational cost* of the classification.

Concerning the correctness of the classification, previous work mostly studies the *classification recall* and the *classification accuracy*. Assuming that a set of network flows with n different traffic types has to be classified, let F_i $(1 \leq i \leq n)$ be the

number of flows that belong to traffic type i, and denote TP_i as the number of flows that are correctly classified with type i.[1] Then, the classification recall R_i for the traffic type i measures the fraction of the correctly classified traffic of type i, and the classification accuracy A represents the overall ratio of the correctly classified traffic [13]:

$$R_i := TP_i/F_i \quad \text{and} \quad A := \sum_i^n TP_i \Big/ \sum_i^n F_i.$$

In addition to the classification correctness, several variables capture the *computational cost* of a classification approach. In this context, the amount of traffic classified per time unit (*classification rate*), and the *build time* for the training phase of supervised ML algorithms has to be considered.

2.2. Traffic classification by inspection of packet payload data

This approach to Internet traffic classification is based on investigating the traffic at multiple network layers including the application layer [14,15]. In this respect, the packet payloads are examined bit by bit to locate specific bit streams which are signatures for a certain network protocol. If such bit stream is found, the packet can be accurately labeled. In practice, payload inspection techniques are commonly employed for P2P traffic detection [1,16,17] and network intrusion detection [18,19].

Despite a relatively high accuracy, traffic classification by payload inspection has numerous disadvantages. First of all, it is a complex operation which requires high computation and storage capacities. The search for the signatures in the payloads requires mirroring (passive) or intercepting (inline) the traffic. Hence, especially for high-speed links, specific hardware is necessary, as software tools that employ payload inspection do not scale [20]. Second, these techniques can fail to classify the flows if new applications with new signatures are encountered, if the traffic is tunneled, and become completely ineffective if end-to-end encryption is used. Finally a very important problem is the violation of user privacy and the related legal issues.

2.3. Traffic classification without inspection of packet payload data

In contrast to payload inspection, there are several approaches that do not rely on payload data. Early traffic classification techniques consider the information in the transport layer and network layer headers, while further approaches evaluate the characteristics of traffic flows.

2.3.1. Port-based classification

Port-based classification is based on the 16-bit transport layer port numbers that are used by servers to determine the specific process that receives a certain traffic flow. This assignment of port numbers to processes is regulated by the Internet Assigned Numbers Authority (IANA) [2].

Early studies of traffic classification rely on the fact that many traffic services run on "well-known," standard transport ports in the IANA list of registered ports [21]. Identification by transport layer ports was employed, especially for P2P traffic detection [22]. This method is losing its effectiveness as the definitive mapping between services and transport ports is not valid anymore. Some protocols are not registered to IANA, and it is possible that IANA does not define ports for all applications, or that some ports are defined ambiguously. Furthermore, standard services can potentially run on non-standard ports to circumvent policy or operating system access control restrictions [12,23]. Moreover, some applications, such as P2P applications, intentionally do not use a predefined set of ports to avoid being detected [1] or applications, such as FTP in passive mode, change their ports dynamically [24]. Thus, port-based classification can produce false results if an application uses another known port or fails to classify the traffic if no matching port is found. The classification accuracy for port-based approaches is reported to be between 50% and 70% in [7,15].

2.3.2. Flow-based classification

The inadequacy of only using transport ports motivated the consideration of additional features that can be identified on a per flow basis. Note that although transport ports can still serve as features, they are no longer exclusively used to determine the packet traffic class. Now, each traffic flow is characterized by a well-defined set of features that will assume different values depending on the respective network traffic class.

Mostly, these features rely on information that can be directly extracted from IP headers. Additionally, per-flow features can include TCP-specific data such as the total amount of data transmitted or the TCP segment size [7]. Practically, flow-traces can be gathered by built-in router facilities. Since 100 MBs of traffic lead to flow data in the order of kBs, flow-based techniques are more scalable to high-speed links than payload inspection. It has to be noted, that flow-based classification requires the transmission of all the packets in a flow before classification can be performed.

A number of flow-based classification approaches are proposed. [17,25] investigate the detection of P2P traffic by combining transport layer port information with the connectivity of the IP host pairs. [12] perform flow-based classification for web, mail (SMTP, POP3) and secure remote access.

[1] This number is also called the number of *true positives* in the literature.

Table 1
Features used by the selected previous ML work on traffic classification.

Ref.	Features employed for traffic classification
[28]	Number of packets, packet length and inter-arrival statistics, flow duration
[23]	Flow duration statistics, data volume, number of packets per flow
[5,7,8]	249 features including port, flow duration and inter-arrival statistics, packet length
[9,27]	Protocol, duration, volume in bytes and packets, packet length and inter-arrival statistics
[29]	Percentage of IP packets with certain sizes, percentage of flows with certain packet sizes and durations, protocol layer ports
[6]	Protocol, source and destination port, number of packets and octets, flow duration, type of service, flags
[30]	Packet length statistics, window size, byte ratio of sent and received packets, number of packets per flow, transport protocol

Table 2
Accuracy values of the selected previous ML work on traffic classification. NB: Naive Bayes, NN: Nearest Neighbor.

Ref.	Algorithms	Applications	% accuracy
[28]	Unsupervised ML, NB	B, I, M, P, S, W	81.9–91.2
[23]	Unsupervised ML	B, I, P, S, T, W	87.4–97.5
[5]	Supervised-NB	A, B, D, G, I, M, P, S, T, W	65–94
[8]	Supervised-NB	B, G, I, M, P, S, W	65–91
[7]	NB, Neural Net	A, B, D, G, I, M, P, S, T, W	74.8 (NB)–99.8 (Neural Net)
[9]	BN, NB, C4.5, NBT	B, G, I, M, P, S, W	$\geq$80 (NB)–$\leq$99 (C4.5)
[27]	BN, NBT, C4.5 NN, MLP, SVM	B, G, I, M, S, W	81 (SVM)–97.8 (C4.5)
[29]	BN	P	62
[6]	BN, C4.5, MLP	P	97.5 (MLP)–99.9 (C4.5)
[30]	SVM	B, I, M, P, S, W	97.17–99.42

2.3.3. Machine learning approaches for flow-based traffic classification

With an increasing number of features, it becomes more difficult to specify a mapping between the features and the respective traffic classes. Hence, Machine Learning (ML) algorithms [26] are employed for traffic classification, where different algorithmic procedures can be applied to construct a *classifier* that groups data instances into different classes based on their feature values. *Supervised* ML algorithms require a pre-classified data set called a *training data set*, while *unsupervised* ML algorithms cluster the data to be classified and associate these clusters to traffic classes.

The previous work on traffic classification using ML algorithms employs different features and algorithms as listed in Tables 1 and 2. The applications to be classified in these works are chosen as prominent applications with well known IANA defined ports and are abbreviated as A: Attack, B: Bulk, D: Database, G: Games, I: Interactive, M: Mail, P: P2P, S: Services, T: Streaming, W: Web. [6,7,9,27] provide a performance evaluation of different algorithms. [9,27] evaluate different ML algorithms with a fixed feature set and different numbers of flows in the training data. Some of the ML algorithms are supported with boosting algorithms, which increases both the accuracy of the algorithm and the computational cost. Excluding such boosted algorithms, Bayesian Networks (BN), C4.5 Decision Trees (DT) and Naive Bayesian Trees (NBT) are reported to give the highest accuracy, where NBTs exhibit a significantly longer build time and a lower classification rate compared to the other algorithms. A significant result of the work in [27] is the particularly low accuracy of Support Vector Machine (SVM) and Multi-layer Perceptron (MLP) algorithms, their high build time and their low classification rate. The performance evaluation study in [7] records build times of up to 39 h for the largest training set with 188 763 flows using a single 2.2 GHz processor. As the classification considers 246 features, larger sets would require several GB of RAM in their approach. In [6], the build time of a BN increases from 0.3 s for 10 000 flows to 3.2 s for 500 000 flows. In contrast, the increase in the build time of a MLP model is from 105.9 s to 4371.7 s. In all the experiments conducted in [6], the classification rate of different models does not go below 12 500 flows/s. Furthermore, [25] states that on a Pentium IV with 512 MB of RAM a classification rate of over 200 000 packets per second could be achieved.

The use of learning algorithms in computer networks is not limited to traffic classification. The cognitive packet networks (CPN) proposed by Gelenbe et al. [31–35] in the context of self-aware networks and autonomic networks employ special control packets which learn about the network state as they traverse the network. This information is later used for QoS-aware routing in backbone networks [36], power-aware routing in wireless ad-hoc networks [37] and making connection admission decisions based on the QoS requirements [38]. In another application, Bayesian classifiers and random neural networks (RNN) are used for the detection of denial of service attacks [39,40].

2.4. Data acquisition for traffic classification

The performance of the traffic classification techniques is evaluated by applying the resulting classifier on a set of *test data* which is previously classified by some other means. Comparing the classification results and the known classes of the test data, the correctness of the respective approach is assessed. A significant issue is that the accuracy of the performance evaluation results is limited to the accuracy of the test data classification.

Table 3
Data acquisition of the selected previous work on traffic classification. RN: Real Network, PT: Publicly available trace, PB: Port-based classification, PI: Payload inspection, MC: Manual classification, TS: Trusted server, NA: Not available.

Ref.	Date	Acquisition, pre-classification	Flow information
[28]	2001	PT [43], PB	3.5 000 000 flows
[23]	2001, 2003	RN, PT [43], PB	NA
[5]	2005	RN, MC	10 sets of flows, 25 000 to 65 000 flows per set
[8]	2005	RN, MC	25 000–65 000 flows
[7]	2005	RN, MC	337 000 flows, training: 400 to 189 000 flows, test: 168 000 and 90 000 flows
[9,27]	2001–2003	PT [41], PB	24 000 flows, 4000 flows per traffic class
[29]	2006	RN, PI	1000 000 flows
[6]	2007	RN, TS	4000 000 flows
[30]	NA	RN, PI	8-h data on a Gbps Ethernet link, 70% identified and used
[1,17]	2002–2004	RN, PT [42], PB, PI	2–4 billion packets
[12]	NA	RN, TS	Training: 15 000 flows, test: 6000 flows

Similarly, most traffic classification techniques need previously classified data to characterize the traffic classes. In particular, the supervised ML techniques require *training data* to build the model. The training data set contains flows that are represented by their feature values (input) and the corresponding traffic type (output). In the training process, a function is generated to predict the traffic type for the given feature values by using the flows in the training data. Therefore, the quality of the training data is one of the key elements for the performance of the supervised learner, since a large number of inconsistent entries in the input/output pairs severely affect the learner. In addition to the correctness of the test and training data, the amount and age of this data is significant, as the composition and properties of network traffic changes continuously with new applications, such as different generations of P2P communication. The importance of the accuracy and the amount of training data for supervised ML algorithms is discussed in [7].

There are different possibilities to obtain the training and testing flows for flow-based traffic classification. They can be collected from a real network, acquired from publicly available repositories such as [41–43] or generated by a synthetic data generation tool [44–46]. Since the traffic type information for each flow is not available for the listed methods, the quality of the obtained data relies on the correct classification of the respective flows using payload inspection or port-based methods with their disadvantages, as discussed in Sections 2.2 and 2.3.1. A method that circumvents these problems and inherently classifies flow traces correctly is the collection of flow data from trusted servers [6,12]. The data acquisition methods of the classification techniques discussed above are summarized in Table 3.

2.5. Summary and conclusions for previous work on traffic classification

ML is a very promising approach for traffic classification. On the one hand ML does not rely on packet payload data which has disadvantages with respect to both correctness and computational cost. On the other hand, ML makes use of a number of features, which can be collected from traffic flows rather than depending on a single feature such as the transport layer port, and provides more correct results. A number of previous researches study ML algorithms for flow-based traffic classification for different types of traffic, with different ML algorithms and flow features and using data collected by different means, as listed in Tables 1–3. In Section 3.1, we select BN, DT and MLP as the ML techniques for our study.

It can further be observed from Table 2 that there is no consensus on the traffic classes studied in the previous literature. This is mainly caused by two reasons. First, several studies [1,9,23,28] are restricted to applications that are available in previously prepared and served (publicly or with authorization) data packages [41–43]. Second, some studies choose a subset of the listed applications on purpose [6,12,17,25,29], according to the motivation of the respective study, such as popularity of the protocol, bandwidth consumption or type of data carried. In Section 3.1, we choose 6 traffic types that share a very high volume of the overall Internet traffic, including traffic types such as P2P and content delivery traffic [47] that are particularly difficult to classify.

Supervised ML algorithms require large amounts of correctly pre-classified and labeled data for training and testing the performance. The data acquired for these purposes in the previous work are pre-classified by port-based or payload inspection methods as in Table 3. In addition they include data that date back to 2001. In Section 3.2, we elaborate our method to acquire a very large amount of correctly pre-classified and labeled training and testing data that have recently been collected from trusted servers.

A number of ML algorithms are investigated and high accuracies and feasible training times for BNs and C4.5 DTs are reported, as listed in Table 2. In all these studies different sets of training and testing data are used without investigating the impact of the data composition and correctness of the available training and testing data on the results. In Section 3.3, we propose an approach for the detailed systematic evaluation of BNs, C4.5 DTs and MLPs with respect to different training data set sizes and compositions, and apply this approach in Section 4. Furthermore, we consider issues such as dynamic and masquerading port use and erroneous training data in Section 4.3.

Some of the features that are employed in the previous work can be readily extracted from the flows whereas some of them require expensive computations for statistical analysis, as listed in Table 1. Similarly, a large number of features can

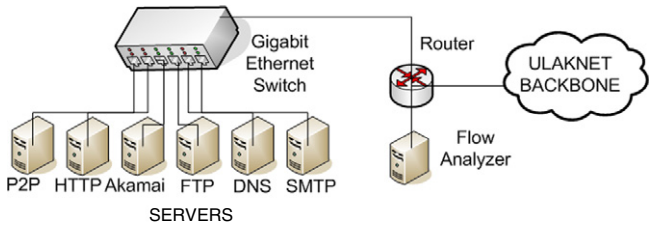

Fig. 1. Network topology for reliable data acquisition.

significantly slow down the classification without any considerable improvement. In Section 3.3 we use a small number of features that can be readily extracted without any statistical processing.

3. Our experimental methodology

3.1. Selected ML algorithms and traffic types

In this paper we investigate BNs and C4.5 DTs due to their reported high accuracies in [9,27], feasible training times and classification rates. Furthermore it is stated that the DTs perform better when dealing with discrete/categorical features and the BNs are found to be successful if the number of features is not very high [4]. Hence, BNs fit to our framework as we perform traffic classification using only flow data with a small number of features. In a further study, we choose MLPs in order to analyze a representative of an algorithm that is reported to exhibit a comparatively low accuracy for general traffic classification [27].

We focus on six different applications: P2P (P), content delivery (C) [47], web (W) (HTTP), bulk (B) (FTP), service (S) (DNS), and mail (M) (SMTP). According to a previous comparative study that was performed on the National Academic Network of Turkey (ULAKNET) [48], the collective share of these six applications was more than 90% in all tests, while there was no other application with a share that exceeded 1%.

The well-known protocols web, bulk, mail and service traffic have also been studied in the context of traffic classification in [5,7–9,12,14,15,23,28]. P2P currently has the largest share among the overall Internet traffic, varying from 35% to 70% according to different researches in this area [1,49]. Identifying P2P communication is a very significant issue addressed by several Internet traffic classification approaches [1,6,16,17,22,25,29]. Content delivery in a transparent and distributed manner is becoming more and more popular. Akamai Technologies [47] provides a distributed computing platform for global Internet content and application delivery. Although there are studies on the traffic characteristics of Akamai, such as the request rate, the bandwidth consumption and redirection dynamics [50,51], flow-based traffic classification has not been investigated yet.

3.2. Data acquisition technique: Data collected from trusted servers

The training and test data that we use in this work are acquired from the National Academic Network of Turkey (ULAKNET) maintained by the Turkish Academic Network and Information Center (ULAKBIM) [52]. ULAKBIM provides a service to 90 000 academic personnel and more than 2 million university students. There are approximately 200 000 active devices connected to ULAKNET that is itself connected to the global Internet via two 1 Gbps links with a utilization of 95%. Moreover, an STM-4 connection exists between the European Academic Network GEANT and ULAKNET.

3.2.1. Network topology

Our data acquisition network topology is shown in Fig. 1. For traffic *generation*, quarantined servers with unique IP addresses deliver data packets which belong to one distinct protocol per server to the Internet. Hence, the traffic types of the flows that originate from these servers are known without ambiguity, and all flows can be labeled correctly. In order to *collect* several traffic flow characteristics, we enable the NetFlow feature [3] on the router to export flow information about the routed traffic gathered from the IP headers of the transmitted packets. The collected NetFlow data is directed to a host machine that we call the Flow Analyzer, which is connected to the router on a 100 Mbps link. On this machine, the General Public License program Flow-Tools [53] captures and stores the directed data for further processing. Our Flow Analyzer is a PC with a Pentium 4 processor, 2 GB of RAM and 4×400 GB hard disk. The operating system of the Flow Analyzer is FreeBSD [54]. Together, the combination of correctly labeled traffic flows generated by our quarantined servers and the collection of the flow characteristics with our Flow Analyzer allow the acquisition of accurately labeled and characterized traffic flows.

3.2.2. Generality of the data acquisition approach

In this study we aim to collect of a large amount of accurately labeled traffic flows in a heterogeneous environment. To this end, on the one hand, we use the registered servers of ULAKNET that have been running for several years as the

Table 4
Average traffic measurements for the quarantined servers (March 2008).

Server	Running applications	Traffic measurements
P	Azureus [55], DC++ [56], LimeWire [57], eMule [58], Kazaa [59]	70 Mbps upload
W	Apache [60]	1 Mbps, 4 million hits
C	Akamai [47]	100 Mbps upload
B	WS_FTP [61]	40 Mbps upload
S	Bind [62]	5 Mbps, 45 million queries
M	PostFix [63]	1 Mbps, 300 000 m delivered, 450 000 m sent

Table 5
The training data set combinations for P2P and web (HTTP) traffic.

E	P2P	Web	E	P2P	Web	E	P2P	Web
1	5000	5 000	4	20 000	5 000	7	80 000	5 000
2	5000	20 000	5	20 000	20 000	8	80 000	20 000
3	5000	80 000	6	20 000	80 000	9	80 000	80 000

quarantined servers that offer HTTP, Akamai, FTP, DNS, and SMTP services. On the other hand, the P2P server was installed for this study at the beginning of February 2008. We started collecting data after one month of operation such that our installed P2P server is recognized by super nodes in P2P networks.

The generation of heterogeneous traffic is supported by the following additional features of our quarantined servers. The P2P server runs 5 different applications as described in Table 4. In order to achieve representative P2P traffic characteristics, data with different sizes and types are supplied to these applications and shared with clients in the global Internet. The HTTP server hosts the classical web content of university and company websites. Owing to the features of content delivery traffic, our Akamai server supplies data not only to ULAKNET but to the Internet community in Turkey and neighboring countries. Our FTP server hosts the mirror of most of the existing Open Source projects. Hence, its traffic is highly correlated to the traffic of FTP servers around the world. The DNS server of ULAKNET that is used in our experiments is the primary/secondary DNS server for around 70 nodes and their subdomains, while our SMTP server serves the overall ULAKNET.

Our experiments were conducted during two weeks in March 2008, where the training data were acquired in the first week and the test data were collected in the second week. We collected a total of 116 million flow traces. Out of these, we selected 5 million traces collected on weekdays between 08:00 and 18:00 to be able to gather the characteristics of the investigated protocols at the peak hour of our network utilization. The resulting traffic statistics of the 6 quarantined servers are depicted in Table 4.

3.3. Evaluation methodology

The effect of increasing the number of flows in the training data set on the accuracy of the traffic classification is investigated in [7] for BNs and neural networks. In [6], we perform a preliminary study of the classification performance of BNs, DTs and MLPs for P2P detection with respect to both different numbers of flows and different fractions of flows from each traffic class in the training data set. In this paper, we conduct a rigorous analysis by means of a systematically constructed set of experiments that incorporates large data sets. With this study, we aim at demonstrating the relationship between the training data set sizes of different traffic classes and their respective correct classification. Additionally we evaluate the impact of different training data set sizes on the computational cost.

Training data: Our training data sets are composed of a *small* (5000), *medium* (20 000) or *large* (80 000) number of flows for each traffic class. The values of 5000 and 20 000 flows are chosen according to previous classification studies as described in Section 3.2, while the number of 80 000 flows for specific traffic types represents a value that goes beyond existing works.

We first compose large size sets with 80 000 flows for each traffic class from a continuous one week data set. Next, we extract the first 20 000 flows from each large set to compose the medium sets resulting in 6 new sets. Similarly the small size data sets are composed from the first 5000 instances from the medium sets. With this approach the possible difference in the classification performance for a certain class of traffic depends only on the size of the training data but not on the particular flows.

We compose training sets with different fractions of data from each traffic class to evaluate the correlation between protocol pairs. More complicated dependencies among more than two protocols are not investigated in this paper. For a given protocol pair, we compose $3^2 = 9$ experiments, where each combination of different training data set sizes constitutes an experiment. An example for training data compositions for P2P and web (HTTP) is depicted in Table 5. The experiments investigate the impact of an increasing number of web flows in the training data set, while the number of P2P flows is fixed to 5000, 20 000 and 80 000, respectively and vice versa. The remaining four traffic types have 20 000 flows each.

Testing data: For all experiments, the performance evaluation is carried out on a single verification data set composed of 100 000 flows from each protocol with a total of 600 000 flows. This total number of flows is more than the number of flows used for testing in most of the works discussed in Section 3.2, including [5,7–9]. We choose to compose our test data with an equal number of flows per traffic type such that the recall and accuracy cannot be biased toward the values of traffic

Table 6
Observations for different traffic types under BN classification.

Traffic type	Observations
P	–recall ≥99.6% with 80 000 flows –can be wrongly classified as all traffic types except for service traffic –negative impact: (C80), (M80)
W	–negative impact: (S80), (M80), with 5000 flows
C	–strongly depends on the number of flows –can be wrongly classified as service traffic –negative impact: (W80), (S80)
B	–recall = 99.8% with at least 20 000 flows –no significant dependency on the training data of other traffic types
S	–dependency on the amount of training data for content delivery traffic
M	–recall = 100% with 5000 flows

types with a larger share in the testing data, as observed in [27,30]. Also note that we use distinct flows for training and testing similar to the investigations in [7,12]. Although such arrangement yields lower accuracy values, as reported in [6], it demonstrates the practical case.

Feature selection: One goal of our work is the classification of Internet traffic that is not biased by our experimental setup. Hence, we use flow features of Protocol, Source Port, Destination Port, Packets and Octets that depend on the application but not on our network topology. As we do not want to rely on application dependent features only, we include Active (flow duration), Type of Service and Flags into our feature set. However, the contribution from the latter features is considerably low. Our selected features can be directly gathered from Flow-Tools with a single run. This is important for the scalability of our method, since gathering features after processing the flow traces for multiple times, and investigating complicated relations of flow data considerably slows down the classification process.

Evaluation software tool: The software tool WEKA [64] is used to implement and run the three ML algorithms. It runs on a dual core machine with Intel (R) Core (TM)2 Duo 2.200 GHz processors, and 1.5 GB of RAM allocated to WEKA. We use the default values for the parameters for building the models during the training and testing phases, since we do not focus on adapting the parameters of each algorithm to our specific data set, but we intend to illustrate the significance of an accurate data acquisition method for different ML algorithms. The parameter values of the WEKA tool are as follows: Estimator: simple estimator, search algorithm: K2, ADTree: disabled, confidence factor: 0.25, minimum number of instances for leaf generation: 2, hidden layers: 14, learning rate: 0.3, training time: 500 epochs.

4. Detailed comparative evaluation of ML algorithms with reliable training and test data

In this section, we discuss the results from our experiments that are carried out as described in Section 3.3 with the correctly labeled and characterized traffic flows according to Section 3.2 and for the three ML algorithms BN, DT and MLP chosen in Section 3.1. Note that all the experiments are based on traffic flows that are generated by applications which are configured to use their default source ports. It has to be taken into account that P2P applications frequently switch to other ports if their default ports are occupied, and that the port assignment of the proprietary Akamai service is unknown.

4.1. Classification results for Bayesian networks, decision trees and multilayer perceptron

We summarize our observations from the traffic classification experiments for each traffic type using the different training data sets, as described in Section 3.3 in Tables 6, 11 and 15, for BNs, DTs and MLPs respectively. Furthermore, Figs. 2–4 show the recall values of selected traffic types with respect to the training data set composition for these ML algorithms. For each training flow composition of each traffic type specified on the horizontal axis of these figures, the light gray, dark gray and black columns represent the recall of the respective traffic type when using 5000, 20 000 and 80 000 training flows. (e.g., for the P2P recall, the light grey column at W5 represents the experiment with 5000 P2P flows, 5000 web flows, and 20 000 flows for the remaining traffic types).

Tables 8, 12 and 16 summarize the correlation between the classification recall of certain traffic types and the amount of training instances for the other traffic types. Each row of such a table illustrates the dependency of a particular traffic type on an increasing number of training instances of the traffic types displayed in the columns of the table. Here, a "−" stands for a negative impact, "+" means a positive impact, and "○" reports a negligible impact. Accordingly, each column explains the influence of using more training data for a certain traffic type on the recall of the other traffic types.

BNs: Table 6 summarizes our general observations from experiments on BNs. In addition, we discuss several characteristics of the obtained results.

We observe that the combination of (M80) and (W5) causes 2.5% of the web flows to be labeled as bulk traffic. Therefore, it is possible that a correlation of more than two protocols causes the negative impact on the recall of web traffic. A similar complicated correlation is observed when (S80) is combined with (W5). This time 2.7% of the web flows are

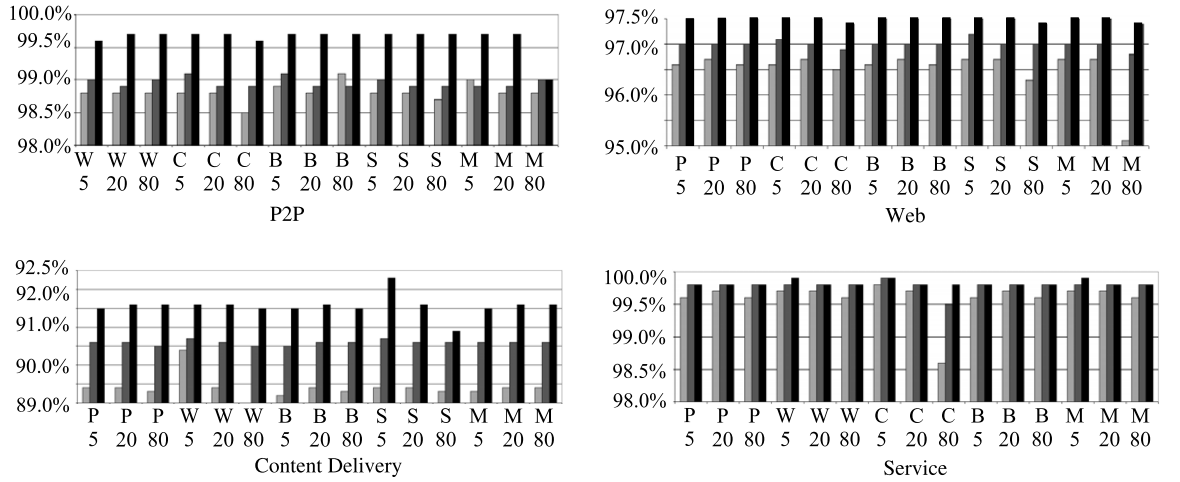

Fig. 2. BNs: Classification recall depending on the amount of training data: Vertical axis: Recall; horizontal axis: Amount of training data.

Table 7
BNs: Classification of content delivery traffic.

Training set	S5 (C80)			S20 (C80)			S80 (C80)		
C classified as	C (%)	B (%)	S (%)	C (%)	B (%)	S (%)	C (%)	B (%)	S (%)
	92.3	0.2	7.5	91.6	0.2	8.2	90.9	0.2	8.9

Table 8
BNs: Correlation between the training instances and recalls.

	P	W	C	B	S	M		P	W	C	B	S	M
P	+	○	−	○	○	−	B	○	○	○	+	○	○
W	○	+	○	○	−	−	S	○	○	−	○	+	○
C	○	−	+	○	−	○	M	○	○	○	○	○	○

labeled as content delivery traffic. One reason for the dependency between Akamai and service traffic could be due to the similarity of their operation. The content delivery servers might attempt to resolve the IP addresses of possible other Akamai servers. This claim is supported by Table 7 that shows how content delivery traffic is actually classified for different amounts of service traffic training data. In all cases, around 8% of the content delivery traffic is wrongly identified as service traffic.

In our experiments, the recall for classifying mail traffic is always 100%. Hence, it is sufficient to use 5000 flows of mail traffic for training. Spam email detection results of the ULAKNET server show that about 47% of the mail traffic consists of spam emails. Spam blocker software is mostly based on content analysis and spam email generators try to avoid detection by inserting certain text and picture files into the emails leading to a certain flow characteristic. This effect can explain the high accuracy rate. For all traffic types except for mail, the recall increases as the number of training data flows for that traffic type is increased. Moreover, our experiments suggest a negative effect of content delivery, service, and mail traffic on other traffic types when BNs are employed for traffic classification, as seen in Table 8.

Based on the results and the correlation between traffic types as investigated above, a training set has been composed that yields a better accuracy than the training sets used in the systematic investigation. It comprises 80 000 P2P flows, 80 000 web flows, 80 000 content delivery flows, 20 000 bulk flows, 20 000 service flows and 5000 mail flows. Although some of the recall values do not reach the absolute maximum obtained in the previous experiments, the overall accuracy of traffic classification could be improved to 98.0%, as shown in Table 9.

For comparison, we perform port-based classification according to the assignment by IANA [2] for our testing set. The result in Table 9 suggests that although a good recall can be achieved for applications with well-known default ports, applications with variable ports (P2P) or unknown port assignments (content delivery) cannot be classified. The latter cases emphasize the relevance of BNs for Internet traffic classification.

Table 10 lists the classification rate (cr) and the build time (bt) with respect to different amounts of training data, as defined in Section 2.1. In [9], 22 features are used for classification of flows with BNs. Similar to our work, the overall accuracy is reported to be greater than 97%. The classification rate and build time evaluate to around 11 000 flows/s and 13 s respectively. A possible explanation of the better computational cost of our experiments is the smaller number of 9 features used for training.

Table 9
BNs: Recall results for different traffic types; maximum achieved accuracy.

	P2P (%)	Web (%)	Content delivery (%)	Bulk (%)	Service (%)	Mail (%)	Accuracy (%)
Maximum value	99.7	97.5	92.3	99.8	99.9	100	97.9
Minimum value	98.5	95.1	89.0	99.5	98.6	100	97.4
Best data set	99.6	97.4	91.5	99.8	99.5	100	98.0
Port-based	0	96.1	0	100	99.1	100	65.8

Table 10
BNs: Computational performance with respect to the amount of training data.

# of flows	90 000	105 000	120 000	165 000	180 000	240 000	Average
cr (flows/s)	68 379	69 178	68 337	62 122	52 425	51 139	61 930
bt (s)	2.5	2.7	3.2	5.2	5.4	7.5	4.4

Table 11
Observations for different traffic types under DT classification.

Traffic type	Observations
P	–20 000 training flows result in a high recall –can be wrongly classified as all traffic types –negative impact: (M80)
W	–strong dependency on the amount of training data –negative impact: (C20), (C80), (P5), (P80)
C	–if 5000 training flows are used, the recall can be as low as 70% –less confusion with service traffic, recall can increase up to 99.3%
B	–can be correctly classified with a high recall (100%) for all experiments
S	–80 000 flows required for a high recall as BNs –negative impact: (C80).
M	–sufficient to use 5000 flows for training to get a recall of 100%

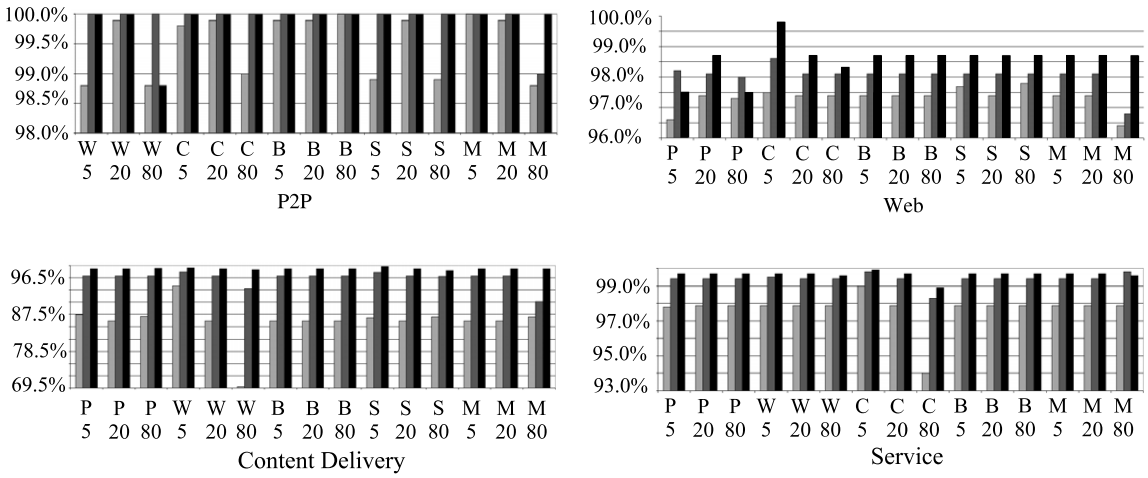

Fig. 3. DTs: Classification recall depending on the amount of training data: Vertical axis: Recall; horizontal axis: Amount of training data.

Table 12
DTs: Correlation between the training instances and recalls.

	P	W	C	B	S	M		P	W	C	B	S	M
P	+	○	○	○	○	−	B	○	○	○	+	○	○
W	−	+	−	○	○	○	S	○	○	−	○	+	○
C	○	−	+	○	−	○	M	○	○	○	○	○	○

DTs: An analogous investigation, as for BNs, is carried out for the ML algorithm C4.5 that is based on DTs. The results are presented in Tables 11, 12 and Fig. 3.

The results show that DTs achieve a high accuracy only if enough training flows are used. This observation is illustrated by the minimum recall values in Table 13, where, e.g., the classification recall of content delivery traffic drops below 70.0%.

Table 13
DTs: Recall results for different traffic types; maximum achieved accuracy.

	P2P (%)	Web (%)	Content delivery (%)	Bulk (%)	Service (%)	Mail (%)	Accuracy (%)
Maximum value	100	99.8	99.3	100	99.9	100	99.2
Minimum value	98.8	96.4	69.8	99.8	94.0	100	94.9
Best test set	100	98.1	98.1	100	98.9	100	99.2
Port-based	0	96.1	0	100	99.1	100	65.8

Table 14
DTs: Computational performance with respect to amount of training data.

# of flows	90 000	105 000	120 000	165 000	180 000	240 000	Average
cr (flows/s)	168 919	118 773	127 388	152 879	161 871	143 834	145 611
bt (s)	6.1	7.3	8.3	15.3	16.0	27.8	13.4
Leafs	127	141	151	171	189	232	189
Nodes	253	182	301	341	377	462	336

Table 15
Observations for different traffic types under MLP classification.

Traffic type	Observations
P	–80 000 training flows result in a high recall –high recall with (B5), (B80) –negative impact: (C5), (S80)
W	–80 000 training flows result in a high recall –high recall with (M5) –negative impact: (P80), 5000 training flows result in a very low recall
C	–high recall with (M5) –no significant increase in the recall after 20 000 training flows –negative impact: (W80)
B	–80 000 training flows result in a high recall –high recall with (M5), (S20) –negative impact: (M80)
S	–no significant increase in the recall after 20 000 training flows –high recall with (B5), (C5), (M5) –sufficient to use 5000 flows with (C5), (M5), (W80) –negative impact: (C80)
M	–high recall with (B5), (W80) –negative impact: (B80), 5000 training flows result in a very low recall

Table 16
MLPs: Correlation between the training instances and recalls.

	P	W	C	B	S	M		P	W	C	B	S	M
P	+	−	∘	∘	−	−	B	∘	∘	∘	+	∘	−
W	−	+	−	∘	−	−	S	∘	∘	−	∘	+	∘
C	∘	−	+	+	∘	−	M	∘	∘	∘	−	∘	+

The accuracy can be further improved when choosing the training set that comprises 20 000 P2P flows, 20 000 web flows, 80 000 content delivery flows, 20 000 bulk flows, 80 000 service flows, and 5000 mail flows. It yields the highest overall accuracy of 99.2%. This result is a considerable improvement compared to the maximum accuracy of 98.0% obtained for training BNs. The corresponding results obtained by a port-based classification of our testing data are listed in Table 13. It can be verified that it is beneficial to use C4.5 in order to correctly classify traffic types that do not use well-known default ports. The computational cost of traffic classification using C4.5 is summarized in Table 14.

In [9], the overall accuracy for C4.5 is reported to be more than 97%. Furthermore, the classification rate of 54 700 flows/s is lower than in our experiments, while the build time of 19 s is larger considering that much smaller training sets are used.

MLPs: Tables 15, 16 and Fig. 4 depict the results of the experiments for the MLP algorithm with backpropagation.

The experiments suggest that it is not possible to have high recalls for all traffic types at the same time. For example the training data set which produces the highest recall for Akamai traffic also has a recall above 90% for other traffic types, except for bulk traffic whose recall is less than 10%. Accordingly, high accuracy values cannot be achieved using MLP, as illustrated in Table 17. The training data composition which is found to yield the highest accuracy has 20 000 P2P flows, 80 000 web flows, 20 000 content delivery flows, 80 000 bulk flows, 20 000 service flows and 20 000 mail flows.

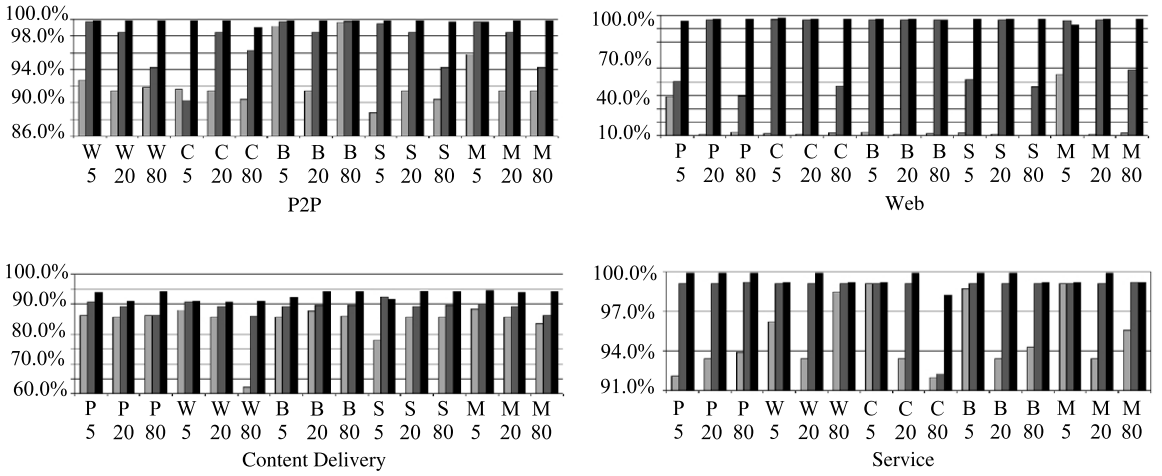

Fig. 4. MLPs: Classification recall depending on the amount of training data: Vertical axis: Recall; horizontal axis: Amount of training data.

Table 17
MLPs: Minimum and maximum recall results for different traffic types; maximum achieved accuracy.

	P2P (%)	Web (%)	Content delivery (%)	Bulk (%)	Service (%)	Mail (%)	Accuracy (%)
Maximum value	99.9	98.2	94.6	100	99.9	100	83.2
Minimum value	88.8	0	62.2	0.2	91.1	0	59.9
Best test set	95.9	96.7	90.8	45.4	99.1	71.8	83.2
Port-based	0	96.1	0	100	99.1	100	65.8

Regarding the computational cost of the MLP algorithm, an average classification rate of 73 395 flows/s and an average build time of 20 min is observed in our experiments.

4.2. Conclusions and comparison of Bayesian networks, decision trees and multilayer perceptrons for traffic classification

Looking at the overall classification accuracy, the best results (up to 99.2%) are obtained if DTs are used. P2P traffic, bulk traffic, and mail traffic can be classified with a recall of 100%. The content delivery traffic can be classified with a better recall up to 99.3% than the BNs. Content delivery traffic is mostly confused with service traffic by the BNs. In general, BNs are expected to model the training data by expressing more complicated relations between nodes by implementing *conditional probability tables*. In contrast, the rule-based operation of the DT is more effective with respect to Bayesian classifiers if there is a candidate root node. In our experiments, the Source Port feature is strongly differentiating between service and content delivery traffic, and hence constitutes a good candidate for the root node of the DT, resulting in the visible improvement compared to BNs.

It is interesting to note that the same correlations as observed for BNs can be verified for the case of C4.5 and content delivery, bulk, service, and mail traffic (See Tables 8 and 12). The most significant difference occurs for web traffic classification, which is mostly influenced by service and mail traffic for BNs and by content delivery traffic for C4.5.

Traffic classification with the MLP algorithm results in a considerably lower accuracy. The main difficulty in applying the MLP algorithm with backpropagation to traffic classification can be explained by examining the dependencies between the recalls of the respective traffic types and an increasing amount of training data for the other traffic types, as illustrated in Table 16. In order to increase the accuracy by increasing the recall of web flows it would be necessary to use more web training flows. However, according to Table 16, this would lead to a decrease in the classification recall of P2P and content delivery traffic. This very sensitive mutual dependency between different traffic types has been observed throughout all of our experiments, and proves to be the main reason for the poor accuracy values that can be achieved with the MLP algorithm.

Considering the computational cost, BNs show the smallest build times especially for large training sets. A much higher classification rate compared to BNs and MLPs is achieved for DTs due to the easier computation of the respective traffic type from the flow features (if/else statements). Relatively large build times are required for larger training sets. This effect can also be observed from the increasing number of nodes and leaves of the respective decision trees. The MLP algorithm requires a considerably larger build time than the other two algorithms and has a similar classification rate to BNs.

Together, our findings suggest that DTs are most suitable for traffic classification, while the performance of MLPs with backpropagation is considerably lower than that of BNs and DTs if all of the 6 traffic types studied in this paper have to be classified.

Table 18
Effect of dynamic and masquerading ports. PB: Port-based.

	BN: % recalls and % accuracy						DT: % recalls and % accuracy					
	P	W	C	B	S	Acc.	P	W	C	B	S	Acc.
Ideal	99.6	97.4	91.5	99.8	99.5	97.8	100	98.1	98.1	100	98.9	99.2
Case I	88.2	86.4	91.6	25.5	99.4	81.6	97.6	98.1	98.3	98.6	98.9	98.6
Case I PB	0	96.6	0	0	99.1	49.3	0	96.6	0	0	99.1	49.3
Case II	94.3	97.4	91.5	99.8	99.4	97.0	99.0	97.7	98.3	100	98.9	99.0
Case II PB	0	96.1	0	100	99.1	65.9	0	96.1	0	100	99.1	65.9

4.3. Variations of the original traffic classification problem

4.3.1. Applications with dynamic and masquerading ports

In our experiments, it is observed that the Source Ports feature dominates the classification. In this section, we investigate the robustness of ML algorithms in case the Source Ports feature is not reliable. Different from the studies presented up to now, we investigate the effect of applications that either dynamically assign certain ports or that masquerade, i.e., that actively use the default ports of other applications. To this end, we employ the characteristics of FTP applications in *passive mode* [24] to dynamically choose ports, and the behavior of P2P which can dynamically choose ports and even intentionally adopt well-known ports such as the HTTP port 80 to masquerade [1]. We analyze the following two cases that reflect these issues for the ML algorithms with the highest accuracy: BNs and DTs.

Case I represents the situation where our FTP server is configured in passive mode using the ports 1024 to 65 535, while the P2P applications exhibit their classical behavior of dynamically assigning ports in the range from 0 to 65 535 and other protocols use their default ports. Hence, it is not possible to correctly classify FTP and P2P flows only based on their source ports. Case II considers that a ratio of 10% of the P2P applications is masquerading with port 80, while all other protocols use their default ports.

In our experiments, we use training sets that are composed according to the ideal sets obtained in Section 4.1. Analogous to Section 3.3, testing sets with equal amounts of 100 000 flows per traffic type are generated from distinct flow data. Both the training and the testing flows exhibit the characteristics of Case I and Case II, respectively.

The obtained results are listed for BNs and DTs in Table 18. For comparison, we performed the port-based classification for the same data and list the respective recalls as well as the recalls determined in Section 4.1 for applications that neither use dynamic ports nor masquerade (ideal). The recall for M is always 100%.

It can be concluded from Table 18 that the BN algorithm can compensate P2P applications with masquerading ports with a moderate impact, while FTP applications in passive mode have a considerably negative effect on the accuracy. When applying classification with C4.5, a very moderate decrease of the recall of P2P and bulk traffic can be observed in Case I. This effect can be explained by the fact that P2P flows and FTP flows share ports, which leads to the confusion of 2.3% of P2P flows as FTP traffic and 1.3% of FTP flows as P2P traffic. For Case II, the recall of web traffic decreases due to masquarading P2P traffic for both ML and port-based classification. However, as features other than Source Port are also employed in ML algorithms, the recalls are less affected compared to port-based classification.

In all of the cases, it is observed that the contribution of the Source Port feature for the classification decreases and a port-based classification tool would give unsatisfactory results due to the applications that use unknown ports. Together, it can be noted that introducing FTP applications in passive mode has a negative effect for classification with both ML algorithms, whereas the impact is significantly higher when BNs are used. Similarly, DTs perform better than BNs if P2P traffic masquerades as web traffic (port 80).

4.3.2. Effect of erroneous training data

Incorrectly labeled training flows do not capture the structure of the network traffic accurately. Hence, the validity of the results presented in the previous sections relies on the correct labeling of our custom-made data according to Section 3.2. In this section, we quantitatively analyze the significance of using correctly classified training data by artificially injecting wrongly classified training data. To this end, the training set with 20 000 flows per traffic type is modified such that a certain percentage (1%, 5%, 10%, 25%, 50%, 75%) of the P2P training data is intentionally wrongly labeled as web traffic flows.

Table 19 shows the performance degradation due to the erroneous training data for BNs and DTs respectively. It can be concluded that erroneous P2P training data have a statistically significant effect only on the recall of the P2P traffic classification.

Here, it turns out that up to 10% and 25% of erroneous training data can be tolerated by BNs and DTs respectively. The relatively high recall values achieved for the erroneous cases suggest that P2P traffic exhibits certain distinctive features such that wrong training data can be accommodated to some degree in the BN algorithm. A dramatic reduction of the classification recall (6.6%) has to be noted for 75% of erroneous training data if DTs are used. This impact can be explained by the fact that the effectiveness of DT based ML algorithms relies on a large enough amount of training data. In the above cases with a high fraction of erroneous data, the amount of P2P training flows is small.

Table 19
Effect of erroneous training data on the classification recall.

Erroneous P training data (%)	BN: % recalls and % accuracy					DT: % recalls and % accuracy				
	P	W	C	S	Acc.	P	W	C	S	Acc.
0	98.9	97.0	90.6	99.8	97.7	100	98.1	97.0	99.4	99.1
1	98.9	97.0	90.6	99.8	97.7	100	98.2	97.0	99.4	99.1
5	96.3	97.1	90.5	99.5	97.3	99.9	98.1	97.0	98.4	99.1
10	92.7	97.0	90.6	99.5	97.3	97.8	98.2	97.4	99.4	98.8
25	90.4	97.0	90.6	99.8	96.3	96.8	98.2	97.4	99.4	98.6
50	88.8	97.0	90.6	99.8	96.0	80.5	98.3	97.1	99.4	95.9
75	70.0	96.9	90.6	99.8	92.9	6.6	98.2	97.0	99.4	83.5

Table 20
% Accuracy. PB: Labels by port-based classifier, TS: Labels by trusted server.

	Reference flows			10% incorrect ports			20% incorrect ports		
	BN	DT	MLP	BN	DT	MLP	BN	DT	MLP
PB	99.98	99.95	99.86	99.79	99.98	99.17	99.68	100.00	99.40
TS	99.69	99.95	88.03	98.16	97.92	86.49	77.30	64.35	58.88

4.3.3. Effect of erroneous training and testing data: Evaluation accuracy of ML algorithms

The premise for an accurate evaluation of different ML techniques for traffic classification is the use of training and test data that accurately represent the characteristics of the Internet traffic. In particular, this means that it is not possible to exclude flows with certain characteristics from the training and testing data, and it is essential that all flows used in the experiments are labeled correctly.

The use of port-based classifiers for labeling potentially violates both of these requirements. On the one hand, traffic flows that cannot be classified by their ports are not included, which does not correctly capture the network traffic characteristics. On the other hand, if the applications adopt well-known ports other than their own, their flows are not classified and labeled correctly. In that case, the accuracy values reported for the different ML algorithms might not demonstrate their actual performance, since the evaluation only considers the match between the outcome of the classifier and the (potentially wrong) labels of the testing data.

In this section we compare the accuracy of our evaluation of the ML algorithms with an evaluation that relies on training and testing data that are labeled by port-based classification similar to [9,27]. To this end, we carry out three experiments with 100 000 flows of training data and 500 000 flows of testing data that are acquired using trusted servers, as described in Section 3.2.[2] In each experiment, we first perform our evaluation of the BN, DT and MLP algorithms with the respective data sets. Then, we conduct an evaluation of the same ML techniques assuming that the training and testing data are pre-classified by a port-based classifier. In particular, this means that modified training and testing sets are used, where flows with unknown ports are removed from the original training and testing data, and it is possible that the classification of certain flows is incorrect.

In the first experiment, we use the original training and testing data, and re-label these flows according to their ports to evaluate the ML algorithms under a port-based pre-classification similar to [9,27]. As a result, 78 593 and 395 545 flows can be port-based classified for training and testing, respectively. The unclassified flows, which constitute about 22% of the data set, are not included in the evaluation as in [9,27]. In addition about 100 flows are wrongly labeled as they adopt other well-known ports. In the second and third experiments, we study the effect of an increasing number of wrongly classified flows. Accordingly, we change 10% and 20% of the original training and testing flows' ports to other well-known ports. We first train and test the ML algorithms with these flows using their correct labels determined according to the trusted servers. Then we pre-classify them based on their ports and train and test the ML algorithms with these flows. Table 20 lists the accuracy values of these three experiments.

We observe that our accuracy drops with an increasing amount of wrongly labeled data. This is expected since the characteristics of the traffic classes become more heterogeneous (in particular, the flows for each traffic class can now have all different ports).

On the other hand, the ML evaluation based on port-based classification always results in a similar high accuracy. This peculiarity can be explained by the fact that if port-based pre-classification is used, the classification is strongly dominated by the Source Ports feature, since all training flows with the same port are assigned to the same traffic class. Now, consider a testing flow of a certain traffic class X whose port is wrongly labeled with the port of another traffic class Y. Then, because of the dominance of the Source Ports feature, the flow will be "wrongly" classified as the other traffic class Y. However, this wrong classification goes unnoticed since the flow is incorrectly labeled as traffic class Y by the port-based pre-classification and hence counted as correctly classified.

[2] Note that we exclude content delivery traffic which cannot be classified by port inspection from our data set to have a fair comparison.

Together, the above discussion shows that port-based pre-classification can lead to an inaccurate evaluation since unclassified flows are not included in the experiments and the impact of wrongly labeled flows might not be reflected in the classification accuracy.

5. Conclusion

The capability of accurate traffic classification is very important for the management, monitoring and provision of networks. In addition to accuracy, scalability of the selected classification method, computation cost of classification and preservation of user privacy are issues that have to be considered.

In this paper we systematically investigate the use of three supervised Machine Learning (ML) algorithms, Bayesian Networks (BNs), Decision Trees (DTs) and Multilayer Perceptrons (MLPs) for flow-based traffic classification, with respect to performance metrics of correctness and computational cost (model build time and classification rate). The traffic classes that we consider are Peer-to-Peer (P2P), web (HTTP), content delivery (Akamai), bulk (FTP), service (DNS) and mail (SMTP). The experiments are conducted by using the software tool WEKA.

The previous studies that investigate ML algorithms for traffic classification are mostly based on a limited number of flow traces that are labeled with a possible inaccuracy due to payload inspection-based or port-based approaches. In addition, traffic flows which cannot be labeled by these approaches are not included in the data sets, limiting the generality of the results. Our first contribution employs more than one million recent and correctly labeled flow traces for the training and testing of the investigated ML algorithms. We include P2P traffic, which can dynamically change port or masquarade with other known ports. Furthermore, we include content delivery traffic (Akamai), which is not studied in the context of flow-based traffic classification in the previous literature.

The second contribution of our work is the systematic investigation of the training data composition for BNs, DTs and MLPs. The correlation between the training data sizes of different traffic classes and its relative impact on the accuracy of the classification is demonstrated by experiments. Our results show that the amount of training data for a certain traffic class can affect the classification accuracy of not only itself but also other traffic classes. Due to our systematic approach, it is possible to construct specific training sets that feature the best accuracy results. Furthermore, a comparison to a port-based classifier shows that, although port-based classification performs well for traffic that uses well-known default ports, applying ML algorithms for traffic classification is advantageous, especially if traffic types such as P2P or content delivery traffic that do not use well-known default ports have to be classified.

Our third contribution is the evaluation of the performance of BNs, DTs and MLPs using the large amount of correctly labeled recent data that we collected from the National Academic Network of Turkey. The performance results indicate that DTs have both a higher accuracy and a higher classification rate than BNs. However, DTs require a larger build time and are more susceptible in the case of incorrect or small amounts of training data. A detailed analysis of traffic classification with MLPs that are trained by backpropagation is carried out to identify the drawbacks of this algorithm that have been previously reported in the literature. In this respect, our fourth contribution is to identify a strong mutual dependency of web, bulk, and mail traffic classification that is very sensitive with respect to changes in the amount of training data. As a result, it is not possible to simultaneously achieve acceptable recall values for these traffic types when the MLP algorithm is used.

Our fifth contribution is composed of three additional experiments. We present a first study of the effect of P2P applications that masquerade, i.e., that use ports of other applications (such as port 80), and FTP applications in passive mode, i.e., with a dynamic port assignment in a certain range. It turns out that the accuracy of the BN algorithm is considerably affected by FTP applications in passive mode, while the DT algorithm only shows a small impact in all of the studied cases. In addition, it could be verified that both algorithms perform significantly better than a purely port-based classification. Next, we perform a fine-grained investigation of the effect of erroneous training data. We incorrectly label a fraction of the P2P training data to demonstrate the significance of accurate training data on the classification recall and accuracy. It can be observed that traffic classification using DTs is less affected than traffic classification with BNs, as long as 25% of the P2P training data are correctly classified. On the other hand, the DTs algorithm is more sensitive with respect to higher fractions of erroneous training data. Finally, we study the effect of incorrectly classified training and testing data, which can for example be the result of a port-based classification, on the accuracy. Our experiments confirm that it is essential to use correctly classified and labeled data in order to obtain valid accuracy results.

Our future work includes investigating the classification performance of unsupervised ML algorithms following a similar methodology, and using a large amount of reliable data. The classification of P2P traffic is a problem of high relevance. We will investigate the effects of including different individual P2P applications in the training and testing data sets on the accuracy of the classifier.

References

[1] T. Karagiannis, A. Broido, N. Brownlee, K.C. Claffy, M. Faloutsos, Is P2P dying or just hiding, in: IEEE Global Telecommunications Conference, GLOBECOM 04, 2004.
[2] The Internet Assigned Numbers Authority, IANA, [Online], Available: http://www.iana.org/.
[3] Cisco systems [Online], Available: http://www.cisco.com/.

[4] S.B. Kotsiantis, Supervised machine learning: A review of classification techniques, Informatica 31 (2007) 249–268.
[5] A.W. Moore, D. Zuev, Internet traffic classification using Bayesian analysis techniques, in: ACM SIGMETRICS 05, 2005.
[6] M. Soysal, E.G. Schmidt, An accurate evaluation of machine learning algorithms for flow-based P2P traffic detection, in: 22nd International Symposium on Computer and Information Sciences, ISCIS07, 2007.
[7] T. Auld, A.W. Moore, S.F. Gull, Bayesian neural networks for Internet traffic classification, IEEE Transactions on Neural Networks 18 (2007) 223–239.
[8] A.W. Moore, D. Zuev, Traffic classification using a statistical approach, in: Passive and Active Measurement Workshop, PAM05, 2005.
[9] N. Williams, S. Zander, G. Armitage, A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification, Computer Communication Review 36 (2006) 7–15.
[10] F.V. Jensen, An Introduction to Bayesian Networks, UCL Press, 1996.
[11] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Francisco, 1993.
[12] M. Crotti, F. Gringoli, P. Pelosato, L. Salgarelli, A statistical approach to IP-level classification of network traffic, in: IEEE International Conference on Communications, ICC06, 2006, pp. 170–176.
[13] Glossary of terms, Machine Learning 30 (1998) 271–274.
[14] T. Karagiannis, K. Papagiannaki, M. Faloutsos, BLINC: Multilevel traffic classification in the dark, SIGCOMM Computer Communication Review 35 (4) (2005) 229–240.
[15] A.W. Moore, K. Papagiannaki, Toward the accurate identification of network applications, in: Lecture Notes in Computer Science, Springer, Berlin, 2005, pp. 41–54.
[16] S. Sen, C. Spatscheck, D. Wang, Accurate, scalable innetwork identification of P2P traffic using application signature, in: 13th International Conference on World Wide Web, 2004.
[17] T. Karagiannis, A. Broido, M. Faloutsos, K.C. Claffy, Transport layer identification of P2P traffic, in: 4th ACM SIGCOMM Conference on Internet Measurement, 2004.
[18] K. Wang, S.J. Stolfo, Anomalous payload-based network intrusion detection, in: Lecture Notes in Computer Science, Springer, Berlin, 2004.
[19] Snort [Online], Available: http://www.snort.org/.
[20] S. Yi, B.-K. Kim, J. Oh, J. Jang, G. Kesidis, C.R. Das, Memory-efficient content filtering hardware for high-speed intrusion detection systems, in: ACM Symposium on Applied Computing, 2007, pp. 264–269.
[21] D. Moore, K. Keys, R. Koga, E. Lagache, K.C. Claffy, The CoralReef software suite as a tool for system and network administrators, in: 15th USENIX Conference on System Administration, LISA01, pp. 133–144.
[22] S. Sen, J. Wang, Analyzing peer-to-peer traffic across large networks, IEEE/ACM Transactions on Networking 12 (2004) 219–232.
[23] M. Roughan, S. Sen, O. Spatscheck, N. Duffield, Class-of-service mapping for QoS: A statistical signature-based approach to IP traffic classification, in: 4th ACM SIGCOMM Conference on Internet Measurement, IMC04, 2004, pp. 135–148.
[24] Active FTP vs. Passive FTP, a definitive explanation [Online], Available: http://slacksite.com/other/ftp.html/.
[25] F. Constantinou, P. Mavrommatis, Identifying known and unknown peer-to-peer traffic, in: 5th IEEE International Symposium on Network Computing and Applications, NCA06, 2006.
[26] T.M. Mitchell, Machine Learning, McGraw-Hill, 1997.
[27] N. Williams, S. Zander, G. Armitage, Evaluating machine learning algorithms for automated network application identification, in: Center for Advanced Internet Architectures, CAIA, Technical Report 060410B, 2006.
[28] J. Erman, A. Mahanti, M. Arlitt, Internet traffic identification using machine learning, in: IEEE Global Telecommunications Conference, GLOBECOM'06, 2006.
[29] E.G. Schmidt, M. Soysal, An intrusion detection based approach for the scalable detection of P2P traffic in the national academic network backbone, in: International Symposium on Computer Networks, ISCN06, 2006, pp. 128–133.
[30] R. Yuan, Z. Li, X. Guan, An SVM-based machine learning method for accurate Internet traffic classification 2008, Information Systems Frontiers.
[31] US Patent 6, 804, 201: Gelenbe, E., Cognitive Packet Network, October 11, 2004.
[32] E. Gelenbe, R. Lent, Nunez, Self-aware networks and QoS, Proceedings of the IEEE 92 (2004) 1478–1489.
[33] E. Gelenbe, Users and services in intelligent networks, Proceedings IEE (ITS) 153 (2006) 213–220.
[34] S. Dobson, S. Denazis, A. Fernández, D. Gaïti, E. Gelenbe, F. Massacci, P. Nixon, P. Saffre, N. Schmidt, M. Smirnow, F. Zambonelli, Autonomic communications, ACM Transactions on Autonomous and Adaptive Systems 1 (2006) 223–259.
[35] E. Gelenbe, Steps toward self-aware networks, Communications of the ACM 52 (2009) 66–75.
[36] E. Gelenbe, P. Liu, J. Lainé, Genetic algorithms for route discovery, IEEE Transactions on Systems, Man and Cybernetics, Part B 36 (2006) 1247–1254.
[37] E. Gelenbe, R. Lent, Power aware ad hoc cognitive packet networks, Ad Hoc Networks Journal 2 (2004) 205–216.
[38] E. Gelenbe, G. Sakellari, M. d'Arienzo, Admission of QoS aware users in a smart network, ACM Transactions on Autonomous and Adaptive Systems 3 (2008) 1–28.
[39] G. Oke, G. Loukas, E. Gelenbe, Detecting denial of service attacks with Bayesian classifiers and the random neural network, in: IEEE International Fuzzy Systems Conference, Fuzz-IEEE 2007, 2007, pp. 1964–1969.
[40] R. Ghanea-Hercock, E. Gelenbe, N.R. Jennings, O. Smith, Allsopp, A. Healing, H. Duman, S. Sparks, N.C. Karunatillake, P. Vytelingum, Hyperion: Next-generation battlespace information services, The Computer Journal 50 (2007) 632–645.
[41] The National Laboratory for Applied Network Research, NLANR, [Online], Available: http://www.nlanr.net/.
[42] Cooperative Association for Internet Data Analysis, CAIDA, [Online], Available: http://www.caida.org/home/.
[43] Auckland data sets [Online], Available: http://www.wand.net.nz/wand/wits/auck/.
[44] NetSpec [Online], Available: http://www.ittc.ku.edu/netspec/.
[45] D. Emma, A. Pescape, G. Ventre, Analysis and experimentation of an open distributed platform for synthetic traffic generation, in: 10th IEEE International Workshop on Future Trends of Distributed Computing Systems, FTDCS 2004, 2004, pp. 277–283.
[46] K.V. Vishwanath, A. Vahdat, Realistic and responsive network traffic generation, in: Conference on Applications, Technologies, Architectures, and Protocols For Computer Communications, SIGCOMM '06, 2006, pp. 111–122.
[47] Akamai [Online], Available: http://www.akamai.com/.
[48] ULAKBIM Internal Report, 2008.
[49] Ipoque Internet study [Online], Available: http://www.ipoque.com/news_&_events/internet_studies/internet_study_2007.
[50] S. Saroiu, K.P. Gummadi, R.J. Dunn, S.D. Gribble, H.M. Levy, An analysis of Internet content delivery systems, in: 5th Symposium on Operating Systems Design and Implementation, OSDI02, 2002.
[51] A. Su, D.R. Choffnes, A. Kuzmanovic, F.E. Bustamante, Drafting behind Akamai (travelocity-based detouring), SIGCOMM Computer Communication Review 36 (2006) 435–446.
[52] Turkish Academic Network and Information Center, ULAKBIM, [Online], Available: www.ulakbim.gov.tr/.
[53] Flow-tools [Online], Available: http://www.splintered.net/sw/flow-tools/.
[54] The FreeBSD project http://www.freebsd.org/.
[55] Azureus [Online], Available: http://www.azureus.com/.
[56] DC++ [Online], Available: http://www.dcplusplus.org/.
[57] LimeWire [Online], Available: http://www.limewire.com/.
[58] eMule [Online], Available: http://www.emule.com/.
[59] Kazaa [Online], Available: http://www.kazaa.com/.
[60] Apache [Online], Available: http://www.apache.org/.
[61] WS FTP [Online], Available: http://www.wsftp.com/.

[62] Bind [Online], Available: http://www.isc.org/products/BIND/.
[63] PostFix [Online], Available: http://www.postfix.org/.
[64] Waikato environment for knowledge analysis [Online], Available: http://www.cs.waikato.ac.nz/ml/weka/.

Murat Soysal received his B.S. and M.S. degrees in electrical and electronics engineering from the Middle East Technical University, Ankara, Turkey. Currently he is carrying out his Ph.D. study in electrical and electronics engineering in the Middle East Technical University. He works as a researcher in Network Technologies Department in ULAKBİM. He performs the design and administration of the National Academic Network (ULAKNET) wide area backbone. His research interests include network traffic engineering, network security and machine learning.

Ece Guran Schmidt received the B.S. degree in electrical and electronics engineering from the Middle East Technical University, Ankara, Turkey, in 1997 and the M.S. and Ph.D. degrees in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, in 2001 and 2004, respectively.

She is currently a Faculty Member with the Department of Electrical and Electronics Engineering, Middle East Technical University. Her research interests include high-speed networks, networked control systems, and vehicular communication networks.