Challenging Statistical Classification for Operational Usage: the ADSL Case

Marcin Pietrzyk and Jean-Laurent Costeux
Orange Labs, France
{marcin.pietrzyk,jeanlaurent.costeux}@orange-ftgroup.com

Guillaume Urvoy-Keller and Taoufik En-Najjary
Eurecom, France
{urvoy,ennajjar}@eurecom.fr

ABSTRACT

Accurate identification of network traffic according to application type is a key issue for most companies, including ISPs. For example, some companies might want to ban p2p traffic from their network while some ISPs might want to offer additional services based on the application. To classify applications on the fly, most companies rely on deep packet inspection (DPI) solutions. While DPI tools can be accurate, they require constant updates of their signatures database. Recently, several statistical traffic classification methods have been proposed. In this paper, we investigate the use of these methods for an ADSL provider managing many Points of Presence (PoPs). We demonstrate that statistical methods can offer performance similar to the ones of DPI tools when the classifier is trained for a specific site. It can also complement existing DPI techniques to mine traffic that the DPI solution failed to identify. However, we also demonstrate that, even if a statistical classifier is very accurate on one site, the resulting model cannot be applied directly to other locations. We show that this problem stems from the statistical classifier learning site specific information.

Categories and Subject Descriptors: C.2.3 [Computer Communication Networks]: Network Operations

General Terms: Measurements, Algorithms.

Keywords: Traffic Classification, Machine Learning.

1. INTRODUCTION

A key issue for companies and Internet Service Providers (ISPs) is the ability to precisely identify the applications flowing in their networks. Motivations behind this need are manifold: (i) enforcement of internal or national rules, e.g., banning p2p traffic from an Intranet, (ii) better understanding of actual and emerging applications (iii) assessment of the impact of those applications on peering agreements and/or the return on investment if some p4p initiative was taken [26] or (iv) possibility to offer additional services based on application, e.g., protection of multimedia transfers.

The current state of the art for most companies, including ISPs, is to rely on some proprietary solutions that implement deep packet inspection (DPI) techniques featuring signatures and ad-hoc heuristics to detect current applications. While this approach can be accurate, it is expensive, scales poorly to high bandwidth and requires constant updates of the signatures database to detect new applications or new usage of existing applications or protocols. Furthermore, the growing trend of obfuscating traffic highlights the need of alternative detection methods. Recently, several solutions based on machine learning techniques and per flow features were proposed in the literature e.g. [16, 3, 2, 15, 17]. The majority of these techniques were tested on academic traces, use different traffic features as inputs to the statistical classification algorithm and define flows and application classes differently.

In this paper, we adopt the perspective of an ADSL provider. We are evaluating statistical classification[1] as a complementary tool to deep packet inspection. Indeed, it might be too costly to deploy a DPI tool at each point of presence (PoP) of an ISP. A typical use could be to devise a statistical classifier built upon the knowledge (reference point) collected on the PoPs where some DPI solutions are available, to be deployed where those DPI solutions are missing. In addition, whatever the DPI tool is used, there is always a fraction of traffic that it can not identify. In our traces, this unidentified traffic represents between 8 and 24% of the bytes. A statistical classification solution could help decreasing those values.

We have collected several hour long traces at various ADSL PoPs of a French ISP. Our data set is unique, as those traces form an homogeneous set in the sense that they were captured at about the same period (beginning of 2008) and all PoPs are under the control of the same ISP. Using those traces, we address the following issues:

- *Can we obtain a high classification accuracy, and this, for all the applications of interest?*
- *Can statistical methods help in mining the traffic that*

[1]In this paper, we focus on supervised statistical classification where a specific machine learning algorithm is trained on a so-called training set (for which the reference point is known), using a specific set of features. We call statistical classifier the resulting tool.

IMC'09, November 4–6, 2009, Chicago, Illinois, USA.

DPI tools failed to classify?

- *Is the statistical model representative of the applications, i.e., can we train the classifier on one site and use it on another one without specific adjustments or re-training? Could we use a statistical tool as a alternative to commercial DPI tools?*

Contributions of our study can be categorized into two sets. The first set relates to the use of statistical techniques for **each site independently** of the others. In such a scenario, we demonstrate that:

- Statistical classification can help revealing the traffic left unknown by the ground truth establishment tool. More precisely, we demonstrate that supervised classification techniques can divide by a factor of 2 the amount of bytes previously unidentified by our DPI tool.
- Statistical classification is flexible enough to allow to group traffic based on application rather than protocol. This is particularly important for the case of HTTP that is a bearer for many applications ranging from mail to video streaming.

When the statistical classifier is applied on a site **different** from the one on which it was trained, we show that:

- Average performance is good – when considering all flows and applications – but results can greatly deteriorate on an application basis. This means that some applications that are correctly classified when the classifier is applied on the same site where it was trained, become difficult to identify when applied on another site. We demonstrate that many applications can suffer from this problem, including mail, ftp and some p2p applications. A precise investigation of those cases allows us to prove that the problem stems from an *overfitting* of the data, where the classifier learns some site specific characteristics used by local users/applications.

The remainder of this paper is organized as follows. After reviewing related work in Section 2, we describe our data, reference point establishment method and methodology in Sections 3 and 4. Section 5 presents the results of classification per site. In Section 6, we challenge the classifier in cross-site experiment. We show how statistical classification can help mining unknown traffic in Section 7. Section 8 concludes the paper.

2. RELATED WORK

Recently, many different methods have been introduced to solve the traffic classification problem. Early approaches relied on port numbers. Observation of the decrease of accuracy of classical port number approaches was reported notably in [12]. It triggered the emergence of deep packet inspection (DPI) solutions. In this approach, packet payloads are checked against signatures of known applications [21]. The emergence of encryption and obfuscation of packet content, the need of constant updates of application signatures and governments regulations, might however undermine the ability to inspect packets content.

Newer approaches classify traffic by recognizing statistical patterns in externally observable attributes of the traffic. Their ultimate goal is either clustering IP traffic flows into groups that have similar traffic patterns, or classifying one or more applications of interest. Moore et al. in [17] presented a statistical approach to classify the traffic into different types of services based on a combination of flow features. This line of inquiry attracted particular attention, resulting in a variety of machine learning algorithms, flow features and heuristics, e.g., [18, 25, 8, 2, 4]. A systematic survey of eighteen recent works is provided in [24].

Experience has shown that the combination of a small number of flow features already has a strong discriminative power to differentiate services or network applications on a given dataset. In this work, we focus on the spatial stability of the classification of ADSL traffic, i.e., the ability to train a statistical classifier on one site before using it to monitor other sites. This is key issue for the operational deployment. To the best of our knowledge, the only studies that tackled this problem in a way similar to our are [15] and [16]. However, they considered either overly heterogeneous traces [15] or traces collected in academic environments [16] and with long periods of time (one year) between subsequent traces.

3. TRAFFIC DATA

In this section, we present our dataset, how we establish the reference point (ground truth) that is used as benchmark for our statistical classifier, the definition of our traffic classes and the traffic breakdown.

3.1 Dataset

Our dataset consists of four recent packet traces collected at three different ADSL PoPs in France from the same ISP. All traces were collected using passive probes located behind a Broadband Access Server (BAS), which routes traffic to and from the digital subscriber line access multiplexers (DSLAM) and the Internet. Captures, which include full packet payloads, were performed without any sampling or loss and contains over four million TCP flows. Each trace contains at least one hour of full bidirectional traffic, with similar number of active local users varying between 1380 and 2100. For details, see Table 1.

Traces have some important *spatial* and *temporal* features: traces MSI and RIII were captured at exactly the same time at two different locations which helps assess spatial stability of the method[2]. Traces RII and RIII were captured at the same location with an offset of seventeen days between them.

3.2 Reference point

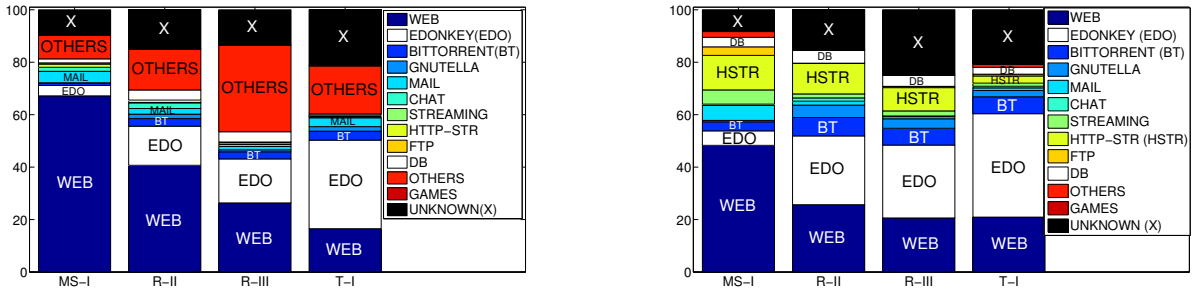
In order to benchmark the performance of any classification method, a dataset with pre-labeled classes of traffic is needed. We term such a dataset our reference point (a.k.a ground truth). Establishing a reference point is fundamental when evaluating traffic classification mechanisms to provide trust-worthy results. As a human-labeled dataset is almost impossible to have, we rely on DPI tools.

Signatures commonly used in recent works [15, 8] provide deceptive results with our traces, as more than 55 % of the flows are classified as unknown. To label applications in our dataset, we rely on an internal tool of Orange, that we term Orange_DPI_tool, or ODT for short. ODT is constantly under developement and in use on several PoPs of Orange in France. It can detect several types of applications, includ-

[2]We also term this problem as the "cross-site" issue.

Set	Date	Start	Dur	Size [GB]	Flows [M]	TCP [%]	TCP Bytes [%]	Local users	Distant IPs
MS-I	2008-02-04	14:45	1h	26	0.99	63	90.0	1380	73.4 K
R-II	2008-01-17	17:05	1h 10m	55	1.8	53	90.0	1820	200 K
R-III	2008-02-04	14:45	1h	36	1.3	54	91.9	2100	295 K
T-I	2006-12-04	12:54	1h 48m	60	4.1	48	94.7	1450	561 K

Table 1: Traces summary.



(a) Breakdown in flows

(b) Breakdown in bytes

Figure 1: Application breakdown in the data sets.

ing encrypted ones. We have compared ODT to Tstat [23], whose latest version features DPI functions, in [19]. Specificaly, we have shown that ODT and Tstat v2 offer similar performance and outperfom signature based tools used in the litterature [15, 8]. As ODT embeds a larger set of signatures than Tstat v2, we rely on the former to establish the ground truth in our study.

We are aware that the wording ground truth remains tricky as even DPI tools might fail. We face here the same issue as former studies in the domain. However, there barely exists any alternative to DPIs. Some approaches have been recently proposed to obtain high quality reference data sets. In [10], the authors propose a network driver installed on end hosts. This middleware flags flows according to the application generating traffic. However, this solution is not applicable to the case of large ADSL traces. The stance we take in this study is thus to compare various statistical classifiers with one another, given an accurate, but not perfect, ground truth establishment tool.

3.3 Traffic breakdown

Classes used in this work are summarized in Table 2. This choice of classes can be considered as a typical one for an ISP that monitors its network. It calls for a few remarks. First, HTTP traffic is broken into several classes depending on the application implemented on top: Webmail is categorized as mail, HTTP streaming as streaming, HTTP file transfers as FTP, etc. Second, popular p2p applications have their own class. Less popular p2p applications are merged into the P2P-REST class. The OTHERS class aggregates less popular applications that ODT recognized (See Table 2).

Figure 1 shows classification results obtained by ODT, in flows and bytes, for our four traces. On PoPs where ODT is used continuously, we checked that the application breakdown is typical of the traffic observed on longer periods of time (day or week). Among the p2p applications, most bytes and flows are due to eDonkey (more precisely eMule client [7]) followed by Bittorrent and Gnutella. Concerning eDonkey, we observed that obfuscated traffic accounts typically for half of the bytes in the EDONKEY class. Less popular file sharing applications (including the P2P-REST class) generated a negligible amount of flows and bytes. We exclude them from our subsequent analysis. We also exclude the NEWS class for similar reasons.

The vast majority of traffic in the HTTP Streaming class is due to Dailymotion [5] and Youtube [27], which account for 80% of the bytes. P2P streaming applications, that fall into the STREAMING class, are active during short time periods, e.g., popular sport events, which probably explains why we do not observe such traffic in our data [1]. The OTHERS class contains mostly unidirectional flows to ports 135, 445 and 139. Those Windows services are targeted by a large family of self-propagating malware (see for instance [20]).

Overall, ODT provides fractions of UNKNOWN bytes that range between 8% and 24% depending on the trace. In Sections 5 and 6, we consider only traffic known by ODT, keeping unclassified flows aside. We focus on the UNKNOWN class in Section 7.

4. CLASSIFICATION METHODOLOGY

This section describes our classification methodology to build our statistical classifier, including the classification algorithms, the flow definition and the performance metrics.

4.1 Classification algorithms

In this paper, we rely on machine learning algorithms provided in the Weka suite [6], that is widely used in the context of traffic classification [16, 15, 25]. Specifically, we evaluated the following supervised learning algorithms [16, 15]:

Naive Bayes Kernel Estimation: this algorithm is a generalization of the Naive Bayes one, which models features using several Gaussian distributions. It is known to be more accurate than Naive Bayes.

Bayesian Network: this algorithm makes use of a model

Class	Application/protocol
WEB	HTTP and HTTPs browsing
EDONKEY	eDonkey, eMule obfuscated
MAIL	SMTP, POP3, IMAP, IMAPs POP3s, HTTP Mail
CHAT	MSN, IRC, Jabber Yahoo Msn, HTTP Chat
HTTP-STR	HTTP Streaming
OTHERS	NBS, Ms-ds, Epmap, Attacks
DB	LDAP, Microsoft SQL, Oracle SQL, mySQL
BITTORRENT	Bittorrent
FTP	Ftp data, Ftp control, HTTP file transfer
GAMES	NFS3, Blizzard Battlenet, Quake II/III Counter Strike, HTTP Games
STREAMING	MS Media Server, Real Player iTunes, Quick Time
GNUTELLA	Gnutella
ARES	Ares
TRIBALL	Triball
P2P-REST	Kazaa, SoulSeek, Filetopia, Others
NEWS	Nntp
UNKNOWN	-

Table 2: Application classes.

Abbreviation	Description
Push_pkt_down	Count of packets with Push flag downstream
Push_pkt_up	Count of packets with Push flag upstream
Avg_seg_size_down	Data bytes divided by # of packets downstream
Min_seg_size_down	Minimum segment size down
Data_pkt_down	Packets with payload downstream
Pkt_size_median_up	Packet size median upstream
Local_port	Local TCP port
Distant_port	Distant TCP port

Table 3: Set B - Per flow features.

that represents a set of features (or classes) as its nodes, and their probabilistic relationship as edges. In some cases, Bayesian Network may outperform Naive Bayes.

C4.5 Decision Tree: this algorithm constructs a model based on a tree structure, in which each internal node represents a test on features, each branch representing an outcome of the test, and each leaf node representing a class label. The version we use incorporates a number of improvements such as pruning that aims at reducing data overfitting. More details about the algorithm can be found in [11].

For all the scenarios we investigated, C4.5 offered the best performance in terms of accuracy and precision (see Section 4.4 for precise definitions). *Unless stated otherwise* results presented in this work were obtained with the C4.5 decision tree algorithm. We will elaborate on the results of the other algorithms in Section 6.4.

4.2 Features

Two broad families of features have been used for classification in the literature. The first one relies on packet-level information like packet sizes [3, 2]. The second family of features consists of flow-level statistics like duration or fraction of push flags [16]. Accordingly, we use two feature sets, one from each family. The first one, that we designate as **set A**, was proposed in [3]. It consists of the size and direction of the first few data packets of a transfer. The second one, **set B**, consists of per flow features inspired by [16]. The full list of features we use is given in Table 3[3]. In this work, we test separately both sets. To extract packet sizes we used the tool released by authors of [3]. For set B, we used ad-hoc tools.

4.3 Flow definition

[3]The features were computed over the whole flow in contrast to [16] where the first five packets of each transfer was used.

We seek to classify bidirectional TCP flow. We use the definition of a flow based on its 5-tuple {*source IP address, destination IP address, protocol, source port, destination port*}. We restrict our attention to TCP flows as they carry the vast majority of bytes in our traces. We are still left with the issue of defining the set of flows to be analyzed. We might restrict ourselves to flows for which a three-way handshake is observed. We can be even more restrictive by imposing observation of a FIN or RST flag at the end of the transfer. The latter option is advocated by the authors in [16], as they observed that for their (academic) traces, imposing this additional constraint does not significantly reduce the fraction of flows and bytes to be analyzed. This is not the case with our traces as we will see below.

Some restrictions might also be imposed by the classification method itself. For instance, when using as features the size of the first 4 data packets (the choice of 4 is justified in Section 5.1), we implicitly exclude all flows with less than 4 data packets. Note that padding small flows with zeros would fool the classifier, and thus it is not an option.

To gain a clear view of the impact of the various filtering options, we applied successively the three following flow definitions to the flows in our traces:

- **S/S:** Only flows with a three way handshake.
- **S/S+4D:** Only flows with a three way handshake and at least four data packets. We used the tool publicly released after the work in [3].
- **S/S+F/R:** Only flows with a three way handshake and with a FIN or RST flag at the end of the data transfer.

Results are depicted in Table 4 for the case of the MS-I trace (other traces offer similar results), with one line per application and the last line presenting average results. Clearly, imposing constraints on the termination of the flow appears extremely restrictive as about 50% of the bytes are exluded from the analysis. On a per application case, the issue can be even more pronounced.

Even imposing the observation of a three way handshake can heavily impact some applications. This is the case for STREAMING, GAMES, DB, and OTHERS. The latter case (OTHERS) results from the nature of traffic carried (presumably attacks), as explained in Section 3.2. For the other classes, this decrease in bytes can be due to flows for which we do not observe the beginning.

Observing the beginning of a transfer is however crucial for traffic classification in general, as it carries application level specific information (while the rest of the transfer might be user data for instance). We thus analyzed only those flows for which we observed a proper three-way handshake. Note that even though the amount of bytes is reduced for some classes, the remaining number of flows per class is large enough (at least several hundreds) to justify further statistical analysis.

Our first set of features (packet sizes) imposes that we have at least 4 data packets per transfer. As we can see from Table 4, this further reduces the number of flows per application but has little impact on the number of bytes due to the heavy-tailed nature of the Internet traffic.

Class	MS-I [flows%/bytes%]		
	S/S+4D	S/S	S/S+F/R
WEB	32%/73%	89%/83%	80%/64%
EDONKEY	88%/91%	97%/98%	86%/51%
MAIL	78%/79%	86%/80%	57%/55%
CHAT	81%/80%	87%/80%	80%/60%
HTTP-STR	85%/98%	92%/99%	81%/79%
OTHERS	11%/35%	22%/42%	16%/24%
DB	27%/11%	33%/12%	15%/9%
BITTORRENT	31%/83%	90%/90%	80%/38%
FTP	29%/65%	76%/67%	71%/64%
GAMES	33%/7%	53%/7%	44%/5%
STREAMING	44%/25%	67%/32%	60%/18%
GNUTELLA	12%/90%	96%/95%	91%/46%
UNKNOWN	19%/19%	39%/21%	34%/14%
OVERALL	34%/69%	77%/75%	68%/55%

Table 4: Remaining flows/bytes depending on the flow definition.

4.4 Performance Metrics

We use performance metrics to assess the quality of our statistical classifier that are commonly used in classification studies. They are built upon the notion of True Positives (TPs), True Negatives (TNs), False Positives (FPs) and False Negatives (FNs). These notions are defined with respect to a specific class. Let us consider such a specific class, say the WEB class. TPs (resp. FNs) are the fraction of WEB flows that are labeled (resp. not labeled) as WEB by the statistical classifier. FPs (resp. TNs) are the fraction of flows not labeled as WEB by ODT that are labeled (resp. not labeled) as WEB by the statistical classifier.

We use the following metrics to assess the performance of the classification method:

- **Accuracy**, **a.k.a Recall**: Accuracy corresponds to the fraction of flows of a specific class correctly classified. It is the ratio of TPs to the sum of TPs and FNs for this class. For example, an accuracy of 50% for the WEB class means that only half of the WEB flows are labelled correctly by the statistical classifier.

- **Precision**: For a given class, it is the ratio of TPs of a class. For example, a precision of 100% for the WEB class means that the statistical classifier has put in this class only WEB flows. This result is satisfactory only if all WEB flows are actually in this class, which is measured by the accuracy.

- **Overall Accuracy**: Sum of all True Positives to the sum of all True Positives and False Positives for all classes (i.e., the sum of all samples). Overall Accuracy is the fraction of correctly classified flows over all classes. If one class has more samples, it will have a larger weight in the overall accuracy.

A classifier works well if it offers, not only high overall accuracy, but both high accuracy and precision for all classes. To explain specific misclassification results, we further make use of the confusion matrix, which indicates how the members of each class are actually classified, i.e. in which class they actually fall. In case we have perfect classification this matrix would be diagonal.

4.5 Training set

With supervised machine learning algorithm, one generally trains the classifier on a fraction of the dataset and tests its performance by applying the (trained) classifier on the remaining of the dataset. Classically, one relies on the 10-fold cross validation technique: for each trace, the algorithm is trained on one tenth of the data and then applied on the remaining flows for each possible slice comprising 10% of the data. Reported results are averages of those ten experiments.

A problem faced in traffic classification is that the number of samples per class is highly varying. This might lead the most prevalent classes to bias the training phase of the classifier. As an alternative, one can use a training set with the same number of flows per class. This approach was advocated in [3]. With our dataset and classes definition, we must limit the number of flows per class to a few hundreds if one wants to apply this approach.

In order to evaluate the impact of different learning scenarios, we trained our classifier using two training sets: (i) 200 flows for each class, (ii) 10,000 flows for the applications with enough flows, and the maximum number of available flows for the less popular applications.

In both cases we obtained similar results with our datasets: less popular classes (e.g. HTTP-STREAMING, GAMES, DB) obtained higher accuracies as compared to the legacy 10-fold cross validation technique, but we observe a decrease of accuracy for the dominant classes, e.g., it drops from 97% to 53% for the WEB class in trace R-III. A closer look at the confusion matrix reveals that by balancing the number of training flows, we are favoring less popular applications causing popular classes to be misclassified. More generaly, we can conclude that in case of unbalanced data sets like ours, there apparently exists a tradeoff between the overall accuracy and the accuracy of less popular traffic classes.

Given the above observations, we decided to use 10-fold cross validation in Section 5 where training and testing are performed on the same trace. On the contrary, when training and testing are performed on difference traces – Section 6 – we use the whole dataset to build the model.

5. CLASSIFICATION - STATIC CASE

In this section we investigate the performance of statistical classification on each site, independently of the others. We term "static case" this issue, as compared to the cross-site case that we will detail in Section 6.

5.1 Number of packets

When using the sizes of the first data packets of a transfer as classification features, we must choose the actual number of packets to be considered. We denote this number as k. We choose the lowest k value that offers good accuracy and precision per application. In Figures 2 and 3, we depict the evolution of accuracy and precision for increasing k values. Results presented were obtained using trace MS-I, as they are similar with other traces. Based on those results, we set k to four packets for the rest of this paper. Note that this value is in line with the ones recommended in [3].

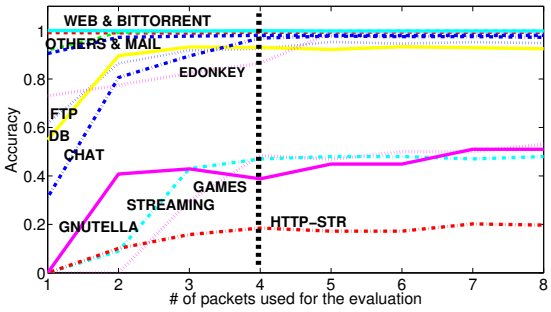


Figure 2: Per-class accuracy vs. number of packets used.

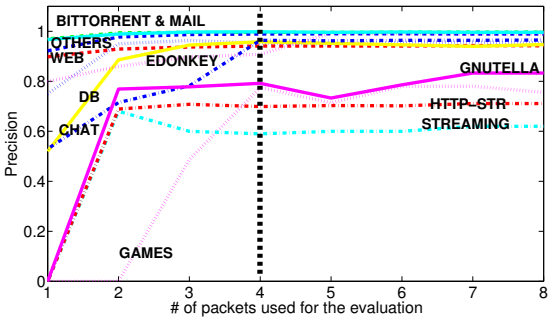


Figure 3: Per-class precision vs. number of packets used.

5.2 Static results

When the classifier is run on the trace on which it was trained, we obtained overall accuracies (over all classes) that are consistently high, above 90% for both sets A and B. The reason behind this result is that the dominant classes in each traces (WEB and EDONKEY) are always very well classified by the statistical classifier. Results on a per application basis are however much more contrasted. Per application accuracy and precision are presented in Figures 5 and 6 for set A and B respectively (results for R-III are omitted as they are similar to the ones of R-II).

The main observation we make is that there exist two broad families of classes. The first family features both a high accuracy and precision for all traces. It contains the following classes: WEB, EDONKEY, BITTORRENT, GNUTELLA, CHAT, FTP, MAIL and OTHERS (GNUTELLA and OTHERS classes have lower accuracy for some traces but the results are still reasonably good).

The second family of classes is characterized by a high precision but a low accuracy. This means that in such a class, one finds mostly correctly classified flows, but a large fraction of the flows that should be in this class, have been classified elsewhere. This is the case for GAMES, STREAMING and HTTP-STREAMING. In order to better understand the problem of those poorly performing classes, we use the confusion matrix (see Figure 4 obtained for set A). To keep the figure clear we indicate only the misclassifications higher or equal to 2%. We found that for the case of HTTP-STREAMING, almost all misclassified flows fall into the WEB class, which is understandable as it might be difficult to discriminate between a streaming and a Web browsing transfer. In contrast, Webmail and HTTP-file transfers, are correctly classified in the WEB and FTP class respectively. This outlines that the application semantics is more important than the lower level protocols in those cases. This is especially important for the case of HTTP as it becomes a bearer for more and more diverse applications.

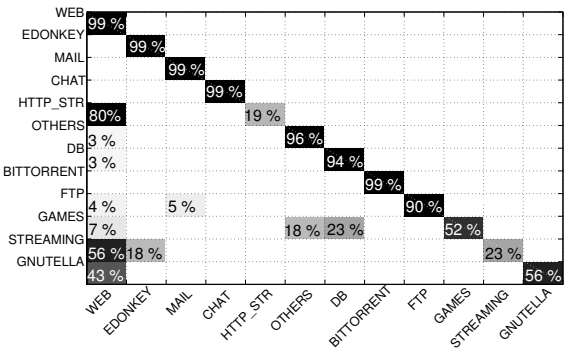
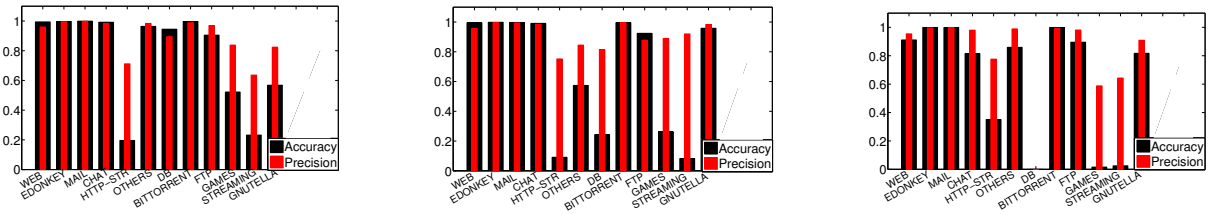


Figure 4: Confusion Matrix for MSI trace, features set A.(Class considered on Y axis is classified as classes on X axis).

For the case of GAMES and STREAMING, misclassified flows are scattered mostly in the WEB and EDONKEY classes. For the case of GAMES, we note that this class aggregates applications with widely different behaviors. This heterogeneity might explain the difficulties faced by the statistical classifier. This observation is further backed by the fact that classification performance are poor for both features sets that we use – see Figures 5 and 6.

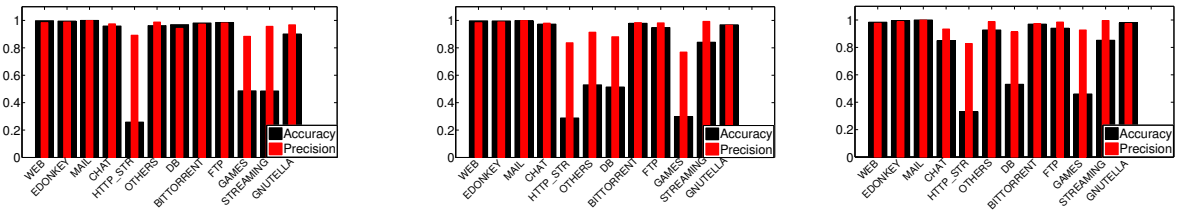
5.3 Static results - Discussion

Results of statistical classification per site are in line with the current knowledge about the state of the art flow features classifiers. Using both set of features we obtained good results for most application classes. However, we would like to assess feasibility of statistical classifier usage as a stand alone solution not accompanied by any DPI tool. In such a case static experiment is not sufficient. We need to verify if the model built over one site is representative enough to be applied on different platforms. We discuss this issue in the next section.



(a) MS-I (b) R-II (c) T-I

Figure 5: Accuracy and Precision using packet sizes (set A) for static case.



(a) MS-I (b) R-II (c) T-I

Figure 6: Accuracy and Precision using set B for static case.

6. CLASSIFICATION - CROSS SITE

In this section, we address the problem of training a classifier on one site and then applying it to an other. Such a technique could be useful for an ISP that would deploy some deep packet inspection tool on one of its major PoPs, train a statistical classifier there and then apply it to its other PoPs. As in the static case, we will first look at the overall performance of the classifier, which means that we focus on the dominant classes. In a second stage, we will detail results per application to illustrate the main outcome of this section, namely the overfitting problem faced by statistical classifiers in cross-site studies.

6.1 Overall Results

In Figure 9, we present the overall accuracy obtained using one trace as a training set (on the y axis) and the others as test sets (on the x-axis). The left matrix corresponds to the use of set A (packet sizes) while the right matrix correspond to set B (flow features). Results are qualitatively similar: the overall accuracy is in general high for the two feature sets, though not as large as in the static case - see Figure 5. The more pronounced degradation is when the T-I trace is considered (as a training or test trace). This might be due to the fact that this trace is older (Dec. 2006) than the other ones. Let us now dig into the details of each class for each different feature sets.

6.2 Set A (packet sizes)

Let us now dig into the details of each class. We focus in this section on the case where the first feature set (set A) is used. Figure 10 depicts the per class accuracy[4] in the cross-site process. Note that we provide results only for the classes that performed well (high accuracy and precision – See Figures 5 and 6) in the static case: WEB, BITTORRENT, CHAT, FTP, MAIL, EDONKEY, GNUTELLA and OTHERS.

[4]Please note that Figures 9 and 10 use different color scales.

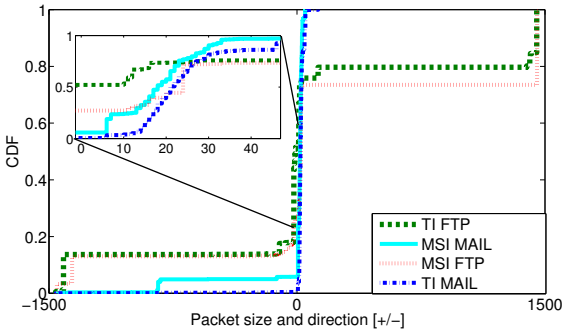


Figure 7: CDF of size of the second packet for MAIL and FTP.

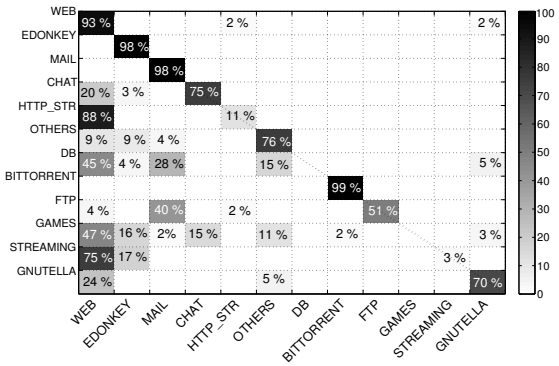


Figure 8: Confusion Matrix for TI (training) on MSI (testing).(Class considered on Y axis is classified as classes on X axis).

A first striking result is that EDONKEY appears immune to performance degradation in a crosssite context[5]. This it not the case for the other classes, even if most of the problems seem to stem from the T-I trace (older trace). This is however not the only explanation behind the observed degradations as there are also problems with BITTORRENT, GNUTELLA, FTP and OTHER classes for the three traces captured in 2008 (See Table 1).

As indicated in Section 3.1, we have two interesting pairs of traces in our dataset. R-II and R-III have been captured on the same site while MS-I and R-III were captured simultaneously. We do observe from Figure 10 that spatial similarity seems more important than temporal similarity. Indeed, for R-II and R-III results are consistently good: over 95% for all classes except OTHERS, which is at 83%. However, the latter class is a potpourri class and we are not certain of having an homogeneous set of applications for this class in the two traces. The picture is different when we focus on MS-I and R-III, as here results can degrade significantly. For FTP, accuracy falls to 52% when MS-I is used as a training trace and R-III as a test trace (and 69% for the other way around). This is in clear contrast with the static case where the accuray was above 90% for the two traces.

We further investigated the case of FTP that seems extremely surprising. We picked on purpose one of the worse performing cases (T-I against MS-I) in order to highlight the problem. While the T-I trace is older, our focus is on FTP and there is no reason to believe that its fundamental characteristics have changed between the end of 2006 and the beginning of 2008. The confusion matrix is a useful tool to pinpoint problems. Figure 8 presents the confusion matrix for the case of training over T-I trace and testing over MS-I. We observe that a significant fraction of FTP is categorized as MAIL. It turns out that the root of this problem is that the distribution of packet sizes on different sites for FTP and MAIL classes sometimes overlap. For instance, we present in Figure 7 the distribution of sizes of the second packet for MS-I and T-I, where we observe this problem. The above issue is a typical case of data overfitting where the classifier has learned overly specific site characteristics. We made similar observations for other cases where a significant degradation was observed from static to cross-site case.

Confusion matrix (Figure 8) shows that misclassifications take place for almost all traffic classes. In most cases we observe significant bias toward most popular classes, namely EDONKEY and WEB. Some applications are also confused with MAIL (like the FTP case discussed above) and OTHERS.

One might argue that the overfitting problem we have highlighted is directly related to the feature set we use. This is however not the case as we will exemplify in the next section with our second set of features.

6.3 Set B (Advanced statistics)

Similarly to the case of set A, we observed significant degradation during our cross-site study with set B. For instance, the CHAT or BITTORRENT classes perform well in the static case but significantly degrade in cross-site studies. Set B consists of several features, each of them being a potential source of data overfitting. It would be a daunting task to study each feature in isolation. We rather take the stance of focusing on one feature, namely port number, for which data overfitting is easy to explain.

It has been claimed in a number of studies [15, 16] that ports have high predictive power and thus should increase classification accuracy. The use of port number is however puzzling as it is treated as a quantitative and not qualitative value. Indeed, most classification algorithms make use of similarity metrics (distances) among the features of the different samples, and from this perspective, port 80 is closer to port 25 than to port 443 or 8080.

To gain a better understanding of the impact of the port number, we applied our second set of features *with* and *without* the port number on the static and cross-site cases. We detail these two cases below.

Port impact - static case.

In all static cases, including port numbers increases both accuracy and precision, typically by a few percent in the case of p2p applications to as much as 38% in the case of FTP class. Let us detail the results for WEB and p2p classes:

- The WEB class is almost unaffected, i.e., ports have minor impact on this class. This is good news given that Web use widely different ports, esp. 80, 443, and 8080.
- Accuracy and precision of p2p classes, especially the EDONKEY class, are significantly increased when using the port number, even though we observed that the legacy ports of those applications are rarely used: 18 to 40% of the flows for EDONKEY and at most 16% for BITTORRENT.

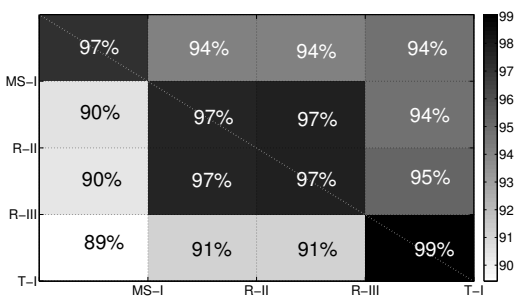
Port impact - cross-site case.

In a cross-site study, using the port number is detrimental, especially for p2p traffic. In fact, in the static case, when the port number is used, the classifier learns particular non legacy port numbers of users. They are predictive in the static case, but misleading in the cross-site case because the non legacy port numbers are not the same between two sites. This is illustrated by Figure 11 for the MS-I and R-II traces (that were captured two weeks apart). We observe that the distribution of remote port numbers is very similar for both traces (Figure 11(b)) while the distribution of local ones clearly differ (Figure 11(a)). The former was to be expected due to the way p2p networks work. As for the latter, it is partly due to some heavy-hitters, i.e. local clients that generate a lot of transfers using e-Donkey. The presence of heavy-hitter being a known and global phenomenon, we can expect to observe a similar phenomenon irrespectively of the actual size of a PoP. To sum up, the port number, although it has a strong predictive power, must be used with caution, as we might run into the problem of overfitting the data. This issue is clearly related to the current usage of p2p applications.

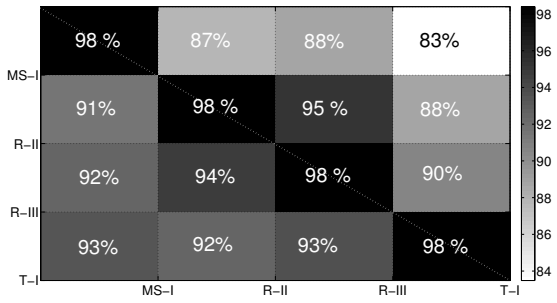
6.4 Impact of the Classification Algorithm

So far, we have considered a single machine learning algorithm, namely C4.5 and different features sets. In this section, we address the other dimension of the problem, namely the impact of the classification algorithm. We consider two

[5]Note that the 99% accuracy in cross-site case comes from the fact that size of some packets for each eMule transfer is deterministic.

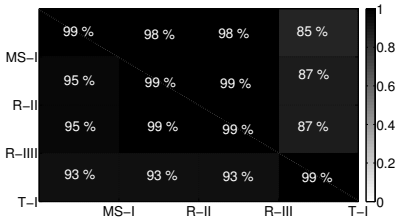


(a) Set A - Sizes of packets

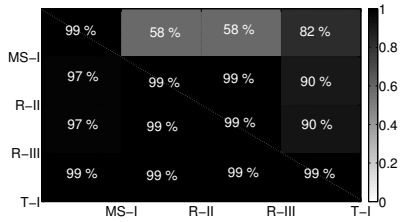


(b) Set B - flow features

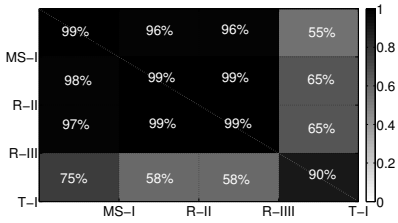
Figure 9: Cross site overall accuracy. (training trace on Y axis, test trace on X axis).



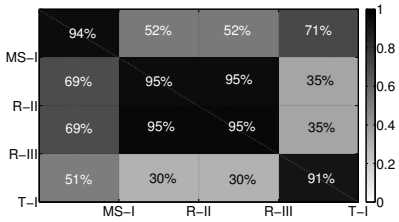
(a) WEB



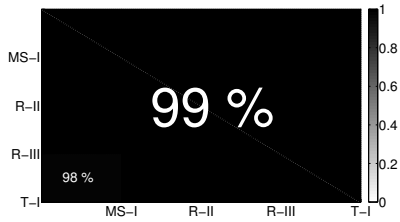
(b) BITTORRENT



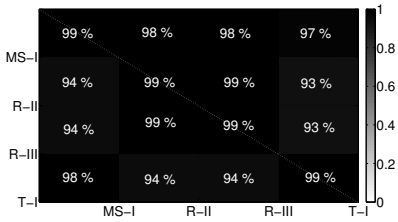
(c) CHAT



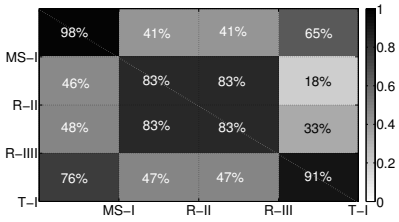
(d) FTP



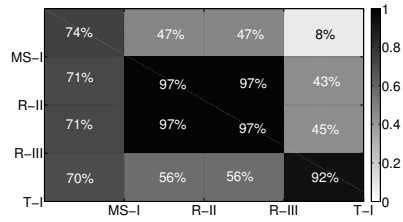
(e) EDONKEY



(f) MAIL



(g) OTHERS



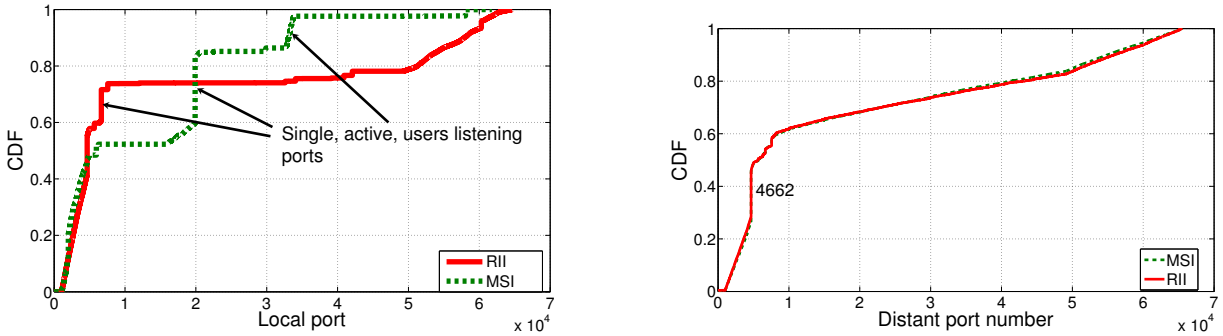
(h) GNUTELLA

Figure 10: Cross site accuracy per application using packet sizes. (training trace on Y axis, test trace on X axis).

alternatives to C4.5: Naive Bayes with kernel estimation and Bayesian Network. As we will see shortly, the issues described in the previous sections persist and can be even more pronounced with these algorithms.

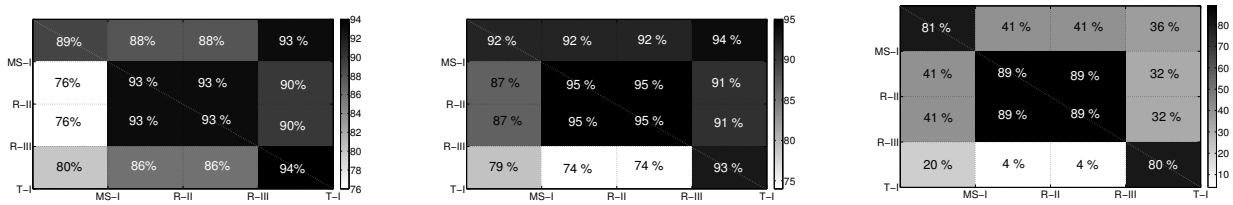
In Figures 12(a) and 12(b) we depict the overall accuracy for both algorithms considered using set A. While using C4.5 for the cross-site studies, we observed that the FTP case turned out to be a complex one. In Figure 12(c), we present accuracy for FTP using Bayesian Network. Detailed, per application, results are omitted for the sake of clarity. From those figures we conclude that:

- In almost all cases C4.5 performs the best in terms of overall accuracy in both static (diagonal elements) and cross-site experiments (non diagonal elements).
- Degradation of overall accuracy for Naive Bayes with kernel density estimation and Bayesian Network in cross-site cases is similar or higher (17 % in the worse case) than with C4.5.
- Per application accuracy degradation, can be even more pronounced for Naive Bayes with kernel density estimation and Bayesian Network than with C4.5. We also



(a) Local port (b) Distant port

Figure 11: Ports for EDONKEY MSI and RII.



(a) N.Bayes kernel estimation (overall) (b) Bayesian Network (overall) (c) Bayesian Network (FTP)

Figure 12: Cross site accuracy for other algorithms (features A). (training trace on Y axis, test trace on X axis).

observed issues with the same classes of applications (e.g., FTP) that caused problems for the decision tree.

Those results confirm our previous findings. The data overfitting issue turns out to be a complex problem that apparently persists when one varies the features set or the machine learning algorithm.

6.5 Cross site - Discussion

The main lesson from this cross-site study is that although the degradation in terms of overall accuracy is often acceptable, some classes, that work correctly in the static case, might suddenly degrade. The above result persits for the various features set or machine learning algorithms we used. We have demonstrated that data overfitting is at the root of the problem. To the best of our knowledge, such a phenomenon was never pointed out before. From this point on, the conclusion is twofold. On one hand, it shows that training a classifier on one site before running on other can lead to unpredictable results. On the other hand, it shows that cross-site studies allow to pinpoint problems that can not be observed otherwise.

A last conclusion suggested by our results is that once a classifier has been trained on a site, it can be used for a significant period of time on this site. However, more work needs to be done to validate this observation that we made for two traces collected two weeks away on the same PoP.

7. MINING THE UNKNOWN CLASS

In most studies where supervised machine learning algorithms are used, results from the statistical classifier are benchmarked against the known traffic, i.e., the traffic identified by the ground truth tool that is used. The rest of the traffic, that we term unknown traffic, is excluded from further analysis. In this section, we go one step further and investigate results obtained when the statistical classifier is used over the UNKNOWN class. Such a classifier could be included as a module of tools like ODT and used as source of information or help in the process of the tool development, in case an increase of unknown traffic is noted. To the best of our knowledge this is the first study that tackles this problem using supervised methods.

7.1 Methodology

Study of the filtering scenarios (see Table 4) revealed that this class consists of a large fraction of connections (61% to 84% depending on the trace) for which the beginning is missing. Those truncated connections however carry the majority of bytes in this class, from 79% to 86%. To maximize the number of bytes for which a prediction could be made, we adopted the following strategy:

1. We used the second set of features. The first one (packet sizes) would have de facto reduced the number of flows and bytes for which a prediction could be made (see Table 4).
2. We trained the classifier on all known traffic for which a three-way handshake was observed (S/S).
3. We apply the classifier on all flows of the UNKNOWN class, without any a priori filtering.
4. Our classifier outputs for each flow a class prediction associated with confidence level.

5. We make use of the confidence level returned by the C4.5 algorithm to select the flows for which we consider the prediction as plausible.

High level procedure is presented in Figure 13. In the latter step of the methodology described above, we used a threshold of confidence level of 95%.

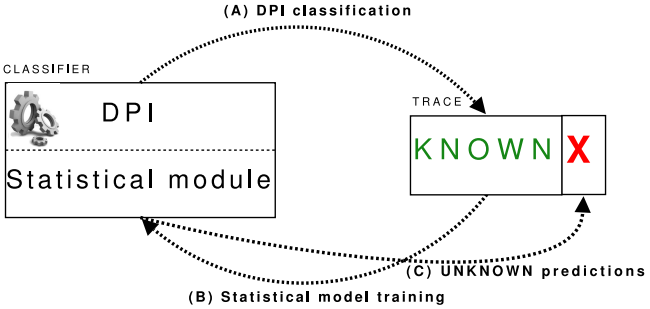


Figure 13: Mining the unknown - schema.

7.2 Predictions

Figure 14 depicts the cumulative distribution function of per flow confidence levels for the flows in the UNKNOWN class. With a threshold of 95%, we observe that, depending on the trace, a fraction between 40% to 70% of the flows are kept for further analysis.

Predictions (classifications) are reported in Table 5. We present only results for classes that performed well in the static case and carry at least 1% of bytes for at least one of the traces. Those results are in line with the ones obtained for the known traffic as we observe a majority of Web, e-Donkey and BitTorrent traffic.

Class	MSI	RII	RIII	TI
EDO.	18%/32%	17%/46%	26%/42%	28%/71%
BT.	1%/15%	5%/14%	8%/12%	2%/9%
GNU.	1%/3%	1%/10%	2%/3%	3%/≤1%
WEB	8%/≤1%	5%/≤1%	9%/≤1%	3%/≤1%
$\sum$	28%/50%	28%/71%	45%/58%	36%/81%

Table 5: Unknown class predictions, [flows%/bytes%].

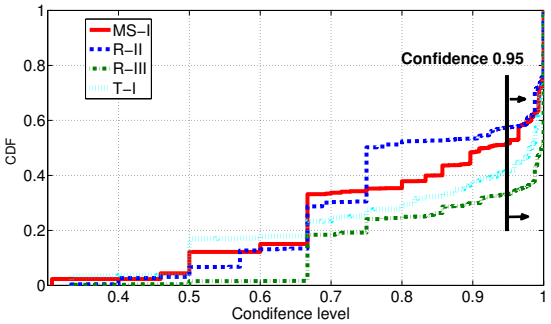


Figure 14: Confidence level vs. fraction of flows.

7.3 Validation

As in this section we operate on unknown traffic, ODT does not provide us any reference point. We need to validate the predictions of the statistical classifier using some other methods. In this section, we perform several side tests to challenge the predictions we obtained for the unknown traffic. We will mainly use the knowledge about the {*IP, port*} pairs of the endpoints of the flows.

7.3.1 Peer-to-peer predictions

For the case of peer-to-peer predictions we use the following additional sources of information per flow:

- **Port numbers.** Even for p2p applications, there is still a fraction of users that use legacy ports [15]. List of legacy ports for popular p2p applications is given in Table 6. If ever such a port is observed for a flow for which the classifier outputs "P2P class", we consider that this information backs the result of the classifier.
- **Endpoint information:**
  - We search for connections to the same remote endpoint e.g. the same {IP,port} pair, in the known set. This method was inspired by the work in [13].
  - We perform **reverse dns lookups** for each remote IP searching for ADSL machines. Most of the providers use simple syntax consisting of IP address and some keywords to identify the hosts of their users. The list of keywords we used is provided in Table 7. It is inspired by [22][6], and based on the hypothesis that communication between two ADSL hosts is likely to be due to a p2p application.

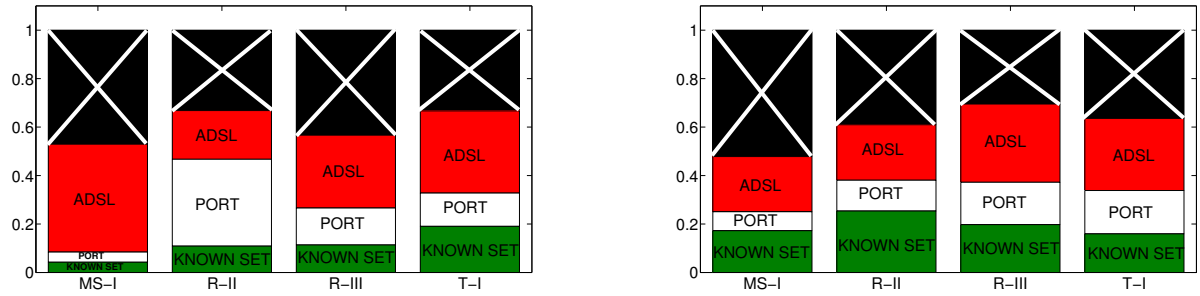
The above procedure is formalized in Algorithm 1. Results for the p2p predictions are presented in Figure 15. Overall, we obtained that at least half of the bytes and flows classified with high confidence predictions are further reinforced by the results of Algorithm 1. The reason why a fraction of p2p flows were not classified by ODT lies in the method used to detect these applications. In most cases, DPI tools need to monitor the beginning of the flows.

7.3.2 Web predictions

For the flows classified as Web, we perform connections attempts to each endpoint, using *wget*, searching active Web servers. The hit ratios was very low, below 3%. However traces are more than one year old, so we can not verify how many servers were really active during the time of the capture.

Using reverse dns queries, we verified that most of the endpoints involved in the flows predicted as WEB flows were residential hosts. In such a case, the existence of transient Web servers can be due to malicious activities like Fast Flux networks [9], which are botnets where compromised machines are used as proxies to hide a Web server. There is also an increasing trend of using HTTP protocol to control bots which

[6]We also implemented a simple google querying tool proposed in [22]. This method relies on parsing the google answers for the {*IP, port*} pairs of the flows seeking for application indication. However the number of hits obtained was too low.



(a) Flows

(b) Bytes

Figure 15: Results of the validation algorithm 1 for the P2P applications.

Algorithm 1: Endpoints profiling.

```
foreach f=flow in P2P do
    if f.prediction.confidence ≥ 0.95 then
        if f.remote.endpoint in known set then
            Known.insert(f)
        else
            if f.local.port==legacy OR
            f.remote.port==legacy then
                Port.insert(f)
            else
                if f.remote.endpoint in adsl set then
                    ADSL.insert(f)
                else
                    Reject.insert(f)
                end
            end
        end
    else
        Reject.insert(f)
    end
end
```

Class	Port
WEB	80, 8080, 443
P2P-EDONKEY	4662, 4672
P2P-BITTORRENT	6881-6889
P2P-GNUTELLA	6346

Table 6: Legacy ports used.

Keyword	Provider
wanadoo	Orange
proxad	Free
dsl/DSL/ADSL	Other providers

Table 7: Keywords used to detect DSL hosts.

is supposed to make the detection more difficult [14]. Such behavior could explain results of our classifier and the fact that the flows were unknown to ODT. We leave for future work study of this hypothesis.

7.3.3 Throughput distributions comparison

A last technique we used to challenge the predictions made by the statistical classifier is to plot distributions of throughput for flows in a given class in the known and unknown sets. We present the resulting cdfs in Figure 16. We observe from this figure that EDONKEY and BITTORRENT predictions seem reasonable as the throughputs for both sets are similar. In addition, those throughputs are clearly smaller than the throughputs of the flows in the known WEB class, which is in line with the fact that residential end hosts are less provisioned than Web servers in general. On the contrary, the unknown WEB class significantly differs from the known one, which is in line with the observation made on the previous section that the remote server was a residential host, and gives further weight to the hypothesis that malicious activities are at play.

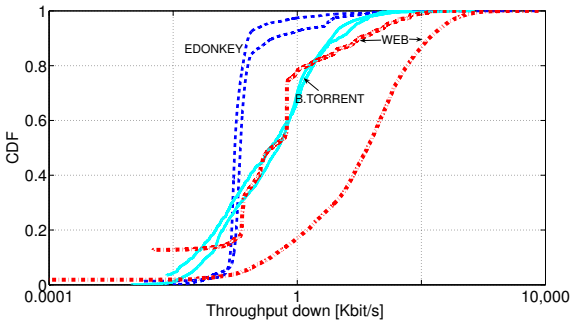


Figure 16: Throughput distributions for KNOWN and predicted sets. Trace MSI.

7.4 The Unknown Class - Discussion

We have shown that a supervised model of traffic classification can be useful to mine the unknown traffic. High confidence predictions were further validated by a number of heuristics based on a variety of endpoint informations and port numbers. We presented the usage of statistical classifier as a complementary method for tools like ODT. Prediction module can be included in the tool and used as a additional source of information in the labor intense process of updating signatures for new versions of emerging applications.

8. CONCLUSION AND FUTURE WORK

In this paper, we adopted the perspective of an ADSL provider, and critically evaluated the potential benefits coming from a production deployment of statistical tools for application identification.

Our conclusions are manifold. On the positive side, statistical classification turns out to be useful to mine the traffic left unidentified by DPI tools. Statistical classification also offers high performance when applied on the same site where they were trained. We have further demonstrated that they allow us to discriminate between applications, even if they rely on the same protocol, e.g. Web mail and Web file transfers.

On the negative side, we have demonstrated that statistical classification tools might suffer from data overfitting, which prevents a simple strategy such as: train on the largest PoP (where ground truth is available) and deploy on all other sites. To the best of our knowledge, this has never been observed before. This problem is complex as it persisted over the whole range of features sets and machine learning algorithms we considered. An important by product of this study is to highlight the need to test new classifiers not simply on traces collected on a given site, but also on traces collected on different sites. The latter needs to be done on "homogeneous" traces in terms of type of traffic and capture time. Indeed, previous attempt to address the cross-site issue, namely [15] and [16] to the best of our knowledge, either considered overly heterogeneous traces [15] or traces collected in academic environments [16] and with long periods of time (one year) between subsequent traces.

As future work, we would like to devise a strategy to select features that would be immune when used in cross-site studies. One of the possible solutions would be to use special set of features depending on application instead of a shared set for all classes. Alternatively, one might be interested in determining under which conditions, some applications or classes are more immune to this problem than others.

9. ACKNOWLEDGMENTS

Thanks are due to our colleagues: Patrick Brown, Ernst Biersack, Daniele Croce, Louis Plissonneau and Paweł Szczepaniec for their feedback on an early version of this paper. We would also like to thank our shepherd Kavé Salamatian as well as the anonymous reviewers for their valuable suggestions.

10. REFERENCES

[1] M. Meo D. Rossi A. Finamore, M. Mellia. Kiss: Stochastic packet inspection. In *COST 2009 : Springer : Lecture Notes in Computer Science, Vol 5537, 2009.*, May 2009.

[2] Laurent Bernaille, Renata Teixeira, Université Pierre, and Marie Curie Lip-cnrs. Early recognition of encrypted applications. *In Passive and Active Measurement conference (PAM 07)*, 2007.

[3] Laurent Bernaille, Renata Teixeira, and Kave Salamatian. Early application identification. In *CoNEXT '06: Proceedings of the 2006 ACM CoNEXT conference*, pages 1–12, New York, NY, USA, 2006. ACM.

[4] Dario Bonfiglio, Marco Mellia, Michela Meo, Dario Rossi, and Paolo Tofanelli. Revealing skype traffic: when randomness plays with you. *SIGCOMM Comput. Commun. Rev.*, 37(4):37–48, 2007.

[5] Dailymotion. http://dailymotion.com/.

[6] WEKA data mining. http://www.cs.waikato.ac.nz/ml/weka/.

[7] eMule. http://www.emule-project.net/.

[8] Jeffrey Erman, Martin Arlitt, and Anirban Mahanti. Traffic classification using clustering algorithms. In *MineNet '06: Proceedings of the 2006 SIGCOMM workshop on Mining network data*, New York, NY, USA, 2006. ACM.

[9] Fast Flux. http://www.darkreading.com/security/perimeter/showarticle.jhtml?articleid=208804630.

[10] S. Malomsoky G. Szabo, D. Orincsay and I. Szabó. On the validation of traffic classification algorithms. *In Passive and Active Measurement conference (PAM 08)*, 2008.

[11] J.R.Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1992.

[12] Thomas Karagiannis, Andre Broido, Michalis Faloutsos, and Kc claffy. Transport layer identification of p2p traffic. In *IMC '04: Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 121–134, New York, NY, USA, 2004. ACM.

[13] Thomas Karagiannis, Konstantina Papagiannaki, and Michalis Faloutsos. Blinc: multilevel traffic classification in the dark. *SIGCOMM Comput. Commun. Rev.*, 35(4), 2005.

[14] Anestis Karasaridis, Brian Rexroad, and David Hoeflin. Wide-scale botnet detection and characterization. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 7–7, Berkeley, CA, USA, 2007. USENIX Association.

[15] Hyunchul Kim, KC Claffy, Marina Fomenkov, Dhiman Barman, Michalis Faloutsos, and KiYoung Lee. Internet traffic classification demystified: myths, caveats, and the best practices. In *CONEXT '08: Proceedings of the 2008 ACM CoNEXT Conference*, pages 1–12, New York, NY, USA, 2008. ACM.

[16] Wei Li, Marco Canini, Andrew W. Moore, and Raffaele Bolla. Efficient application identification and the temporal and spatial stability of classification schema. *Computer Networks*, 53(6):790 – 809, 2009.

[17] Andrew W. Moore and Denis Zuev. Internet traffic classification using bayesian analysis techniques. In *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 50–60, New York, NY, USA, 2005.

[18] T.T.T. Nguyen and G. Armitage. Training on multiple sub-flows to optimise the use of machine learning classifiers in real-world ip networks. pages 369–376, Nov. 2006.

[19] Marcin Pietrzyk, Guillaume Urvoy-Keller, and Jean-Laurent Costeux. Revealing the unknown adsl traffic using statistical methods. In *COST-TMA 2009 : Springer : Lecture Notes in Computer Science, Vol 5537, 2009.*, May 2009.

[20] Honeypot project. http://www.leurrecom.org.

[21] Subhabrata Sen, Oliver Spatscheck, and Dongmei Wang. Accurate, scalable in-network identification of p2p traffic using application signatures. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 512–521, New York, NY, USA, 2004. ACM.

[22] Ionut Trestian, Supranamaya Ranjan, Aleksandar Kuzmanovi, and Antonio Nucci. Unconstrained endpoint profiling (googling the internet). *SIGCOMM Comput. Commun. Rev.*, 38(4), 2008.

[23] Tstat. http://tstat.tlc.polito.it/.

[24] G Armitage TTT Nguyen. A survey of techniques for internet traffic classification using machine learning. *Communications Surveys and Tutorials, IEEE*, 10(4):56–76, 2008.

[25] Nigel Williams, Sebastian Zander, and Grenville Armitage. A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification. *SIGCOMM Comput. Commun. Rev.*, 36(5):5–16, 2006.

[26] Haiyong Xie, Y. Richard Yang, Arvind Krishnamurthy, Yanbin Grace Liu, and Abraham Silberschatz. P4P: provider portal for applications. *SIGCOMM Comput. Commun. Rev.*, 38(4), 2008.

[27] Youtube. http://youtube.com/.