Application-based Feature Selection for Internet Traffic Classification

Taoufik En-Najjary
and Guillaume Urvoy-Keller
Eurecom, France

Marcin Pietrzyk
and Jean-Laurent Costeux
Orange Labs, France

***Abstract*—Recently, several statistical techniques using flow features have been proposed to address the problem of traffic classification. These methods achieve in general high recognition rates of the dominant applications and more random results for less popular ones. This stems from the selection process of the flow features, used as inputs of the statistical algorithm, which is biased toward those dominant applications. As a consequence, existing methods are difficult to adapt to the changing needs of network administrators that might want to quickly identify dominant applications like p2p or HTTP based applications or to zoom on specific less popular (in terms of bytes or flows) applications on a given site, which could be HTTP streaming or Gnutella for instance. We propose a new approach, aimed to address the above mentioned issues, based on logistic regression. Our technique can automatically select distinct, per-application features that best separate each application from the rest of the traffic. In addition, it has a low computation cost and needs only to inspect the first few packets of a flow to classify it, which means that it can be implemented in real time. We exemplify our method using two recent traces collected on two ADSL platforms of a large ISP.**

I. Introduction

Application identification is of major interest for networks operators, especially Internet Service Providers and enterprise network administrators. However mapping flows to applications is not straightforward and has attracted a lot of attention from the research community. Indeed, Internet traffic is the product of a complex multi factor system involving a range of networks, hosts and seemingly uncountable variety of applications. Its complexity is continually increasing as developers keep producing new applications and inventing new usages of the old ones.

Many different methods have been proposed to solve the traffic classification problem. In the early Internet, traffic classification relied on the transport layer identifiers. However, the advent of new protocols like p2p, and the increase of applications tunneled through HTTP make port-based classification significantly misleading. Many studies have confirmed the failure of port-based classification [7]. This triggered the emergence of deep packet inspection (DPI) solutions that identify the application layer protocol by searching for signatures in the payload. The increasing use 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.

Recently, several solutions based on statistical classification techniques and per flow features to probabilistically map flows to applications have been proposed [11], [3], [10], [12].These approaches generally consist of a first phase where flow features are selected based on some intrinsic characteristics like (the lack of) correlation and a second phase where flows are clustered according to the selected features. In general, the overall performance of the proposed statistical classifiers are satisfactory when considering all flows and applications in a given data set. The latter means that the dominant applications, typically Web transfers and some p2p applications like eDonkey, are well classified but other applications that represent a small fractions of transfers, like streaming, might not be correctly identified by the statistical classifier. The reason behind those varying performance might lay in the feature selection process that tends to pick features that are representative of the dominant applications in the considered data set. More generally, we identified a number of challenges for traffic classification that current approaches fail to correctly address:

- A feature selection strategy that selects for each specific (family of) application(s) a *distinct* set of features that best discriminates it from the rest of the traffic.
- The ability to zoom in and out in the traffic as the focus might be on a family of applications like all P2P applications, or on specific applications like eDonkey or Gnutella.
- Resilience to the problem of data over-fitting observed in cross-site studies [14] whereby the statistical classifier capture so-called local information, like port numbers of p2p applications used by local users, that are detrimental when the classifier is applied on a site different from the one where it was trained.
- A classification method with a low computation cost that is further able to work in real time, i.e., after the observation of the first few packets of a connection.

In this paper, we propose to cast any traffic classification question as a logistic regression problem (Section III). Using this approach, we develop a method that has the potential to respond to the above challenges.

The rest of the paper is organized as follows. Section II discusses the related work. Section III provides formal statements of the problems we address, the background on logistic regression, and the classification process. Section IV explains how we obtained and processed the data for our validation experiments, and Section V provides the results from our experimentation with real traffic. Section VI summarizes the work and indicates future avenues of research.

II. Related work

Recent studies have relied on statistical classification techniques to probabilistically map flows and applications [11], [3], [2], [10], [12], [13].Hereafter, we cite a representative sample of traffic classification research. For a much more complete survey, see the work by Nguyen et al. [17].

Moore et al. in [12] presented an approach based on a naive Bayes classifier to to solve the classification problem of TCP traffic. They used a correlation-based filtering algorithm to select the 10 most relevant flow-behavior features. The resulting accuracy, between 93% and 96%, demonstrated the the discriminative power of a combination of flow features and machine learning algorithms.

Bernaille et al. presented in [3] an approach for early identification of applications using start-of-flow information. The authors used the size and direction of the first 4 data packets and port numbers in each flow as features on which they trained K-means, Gaussian mixture model and spectral clustering respectively. Resulting clusters were used together with labeling heuristics to design classifiers. Their results have shown that information from the first packets of a TCP connection are sufficient to classify applications with an accuracy over 90%. The authors further specialized their work to the identification of encrypted traffic in [2].

Karagiannis et al. [8] studied traffic behavior by analyzing interactions between hosts, protocol usage and per-flow features. Their techniques were able to classify 80%-90% of the traffic with a 95% accuracy. In their recent work [9], they applied those techniques to profile the users activities, and to analyze the dynamics of host behaviours.

More recently, Pietrzyk et al. [14] investigated the use of statistical classification algorithms for operational usage. They point out that data over-fitting is a main weakness of statistical classifiers. Indeed, even if a classifier is very accurate on one site, the resulting model cannot be applied directly to other locations. This problem stems from the statistical classifier learning site specific information.

III. Learning classifier using Logistic regression

The use of logistic regression modeling has proliferated during the past decade. From its original use in epidemiological research, the method is now commonly used in many fields including business and finance [18] or criminology [19] to name a few. Logistic regression is designed for dichotomous variables, i.e., to model the relation between a binary variable (true vs. false) and a set of covariates.

In this work we use logistic regression to classify flows of a given application against the rest of the flows. In the remaining of this section, we introduce the logistic regression model. We show how to estimate its parameters for a given application, and how we select the relevant features for the classification of a specific application.

A. Problem statement

The problem of traffic classification consists in associating a class to a network flow, given the information or features that can be extracted from this flow. A flow is defined as a sequence of packets with the same source IP address, destination IP address, source port, and destination port. Let X be the n-dimensional random variable corresponding to the flow features. To each flow a vector x consisting of the n the measured features is associated. Each flow is generated by an application y corresponding to a random variable Y that takes values in the set $\{1, 2, \cdots, c+1\}$, where c is the number of applications. This defines $c+1$ classes; each application defines a class and the $(c+1)^{th}$ class is the default class that contain flows that cannot be associated with any application. The problem of statistical classification is to associate a given flow x with an application y. Logistic regression is a way of defining the relation between x and y. While using logistic regression, we will consider only one application (we call it A) at a time, i.e. $Y = 1$ if the flow is generated by the application of interest and 0 otherwise.

B. Logistic regression model

Consider a flow with the following features vector $x = (x_1, x_2, \cdots, x_n)$. We wish to have a probability of whether this flow is generated by application A or not. Formally, we can state this as

$$p(Y = 1|X = x) = P(x, \beta_A), \tag{1}$$

where $p(Y = 1|X = x)$ is the conditional probability that the flow with features $x = (x_1, x_2, \cdots, x_n)$ is generated by the application A and P is a function of x parametrized by the weights vector $\beta_A = (\beta_0, \beta_1, \cdots, \beta_n)$. Since the function P represents a probability, it must take value between 0 and 1. Within the Logistic regression framework, one assumes a specific function P:

$$P(x, \beta_A) = \frac{e^{\beta_0 + \sum_{j=1}^{n} \beta_i x_i}}{1 + e^{\beta_0 + \sum_{j=1}^{n} \beta_i x_i}}, \tag{2}$$

From the above equation, we can derive a linear function between the odds of having application A and the features vector x, called the logit model:

$$\log\left(\frac{P(x, \beta_A)}{1 - P(x, \beta_A)}\right) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n, \tag{3}$$

Unlike the usual linear regression model, there is no random disturbance term in the equation for the logit model. That does not mean that the model is deterministic because there is still room for randomness in the probabilistic relationship between $P(x, \beta_A)$ and the application A.

To implement any logistic regression model, one needs to choose the $\beta_1, \ldots, \beta_n$ values based on a given training set, i.e., a set of flows for which we know whether they have been generated by A or not. We discuss this issue in the next section.

C. Parameter estimation

For the sake of clarity, we avoided indexing of many variables with the application A. However we would like to point out the fact that the following procedure is done for each application of interest. In particular, it leads to β vectors that are application dependent.

Assigning the parameters to the logit model boils down to estimating the weights vector β, which is usually done using maximum likelihood estimation.

Consider a training data set of N flows characterized by the features vectors $X = (X_1, X_2, \cdots, X_n)$, where $X_i = (x_1^i, x_2^i, \cdots, x_n^i)$ is the features of flow i, and let the vector $Y = (y_1, y_2, \cdots, y_n)$ be such that $y_i = 1$ if flow i is generated by the application A and $y_i = 0$ otherwise. The likelihood function is given by a standard formula [5]

$$\begin{aligned} P(X,\beta) &= \prod_{j=1}^{N} p(Y = y_j|X_j) \\ &= \prod_{j=1}^{N} (p(Y = 1|X_j)^{y_j}(1 - p(Y = 1|X_j))^{1-y_j} \end{aligned} \tag{4}$$

As the values of p are small, it is common to maximize the log-likelihood $L(X,\beta) = \log P(X,\beta)$ instead [5], to avoid rounding errors,

$$L(X,\beta) = \sum_{j=1}^{N} [y_j log(p(Y = 1|X_j)) + (1 - y_j) log(1 - p(Y = 1|X_j))] \tag{5}$$

By substituting the value of $p(Y = 1|X_j)$ by its value defined in Equation (2) we get the log-likelihood for the logistic regression:

$$L(X,\beta) = \sum_{i=1}^{N} \left[y_i \beta^T X_i - log(1 + e^{\beta^T X_i}) \right] \tag{6}$$

In the logistic regression model, we wish to find β that maximizes Equation (6). Unfortunately, this can not be achieved analytically. In this work, we compute it numerically using the Newton-raphson algorithm [5]. This algorithm requires two main components: the first derivative of the log likelihood and the Hessien matrix, i.e., the second derivative matrix with respect to β.

From Equation (6) we can derive the first derivative

$$\frac{\partial L(X,\beta)}{\partial \beta} = \sum_{i=1}^{N} X_i (y_i - p(x_i,\beta)) \tag{7}$$

We now derive the Hessien matrix

$$\frac{\partial^2 L(\beta)}{\partial \beta \partial \beta^T} = -\sum_{i=1}^{N} X_i X_i^T p(x_i,\beta)(1 - p(x_i,\beta)) \tag{8}$$

The pseudo code of Newton-Raphson algorithm is depicted in Algorithm 1. We start with a first guess of β, then we use the first derivative and the Hessien matrix to update β. Using the new β we compute the new log likelihood. This is repeated until there is no further change of β. The Newton-Raphson algorithm has been shown to converge remarkably quickly [6]. In this work, it takes less than one second to output an estimate of β.

Algorithm 1 Newton-Raphson algorithm

```
initialize β
while ||β_new − β_old|| > thr1 and abs(L_new − L_old) > thr2) do
    Calculate g = ∂L/∂β
    Calculate H = ∂²L/∂β²
    Set β_old = β_new
    Calculate β_new = β_old − H^{-1} g
    Set L_old = L_new
    Calculate L_new
end while
Calculate variance matrix V̂
```

D. Selection of relevant features

As we estimate a new model for each application, the weights β_j given for each features emphasis the importance for the corresponding feature to this application. Moreover, logistic regression provide a way to test the relevance of a given feature to the classification output. This can be done through the formulation and testing of a statistical hypothesis to determine whether the corresponding variables in the model are "significantly" related to the outcome variable Y. In other words, for each feature j, we test the hypothesis that the corresponding weight β_j is equal to zero. If we can't reject this hypothesis, this means that this parameter is not relevant to classify this application and, thus, can be removed from the model [6].

In this work, we use the Wald test [6] that tests, individually, for each β_j the null hypothesis that $\hat{\beta}_j = 0$. The Wald statistic $W(j)$ is obtained by comparing the maximum likelihood estimate of each parameter $\hat{\beta}_j$ to an estimate of its standard deviation $\hat{V}(\hat{\beta}_j)$.

$$W(j) = \frac{\hat{\beta}_j}{\hat{V}(\hat{\beta}_j)} \tag{9}$$

The standard deviation $\hat{V}(\hat{\beta}_j)$ of β_j is given by the j^{th} diagonal element of the variance matrix given by Equation (10) [5], that is computed as the last iteration of the Newton-Raphson algorithm (Alg. 1).

$$\hat{V} = \left\{ -\frac{\partial^2 L(\beta)}{\partial \beta \partial \beta^T} \right\}^{-1} \tag{10}$$

Under the *null hypothesis* that $\beta_j = 0$, $W(j)$ follows a standard student *t-distribution* with $n - 1$ degree of freedom t_{n-1}.

For a given significance level α, for each β_j we compute the p-value $pv_j = p(t_{n-1} > W(j))$, and we reject the hypothesis of $\beta_j = 0$ if $\alpha > pv_j$. Otherwise, if we fail to reject the hypothesis of $\beta_j = 0$, we exclude the corresponding feature

from our model. By doing so, we keep a minimum number of features relevant to the application under study.

A crucial aspect of using logistic regression is the choice of an α level to judge the importance of features. Bendel et al [1] have shown that the choice of α smaller than 0.01 is too stringent, often excluding important variables from the model. In this work, we use $\alpha = 0.01$, and we will show in section V-C that it enables to reduce the number of features for each application without decreasing the classification scores.

E. Classification process

Logistic regression falls into the class of supervised machine learning techniques[17]; thus it consists of two main steps. A training step and a classification step.

Training step consist of building a classifier for each application of interest. Consider, for example, the application WEB. Using Newton-raphson algorithm we estimate a vector β_{web} that maximize the probability of being WEB for all WEB flows and minimize this probability for all non-WEB flows.

The classification step is done as follows: a given feature vector $x = (x_1, \cdots, x_p)$ is classified as WEB if $P(x, \beta_{web})$ is larger than a threshold th. A usual choice of the threshold is $th = 0.5$ [6], [5]. By using Equation (3), this boils down to deciding that the new flow x is generated WEB if $\beta_0^{web} + \sum_{i=1}^{n} x_1 \beta_i^{web} > 0$.

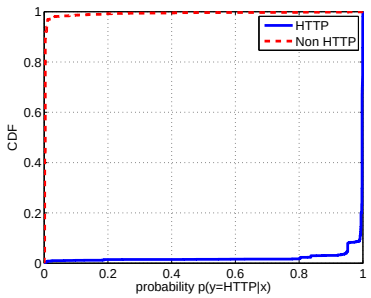

Fig. 1. CDFs of the probability of being a HTTP flows for HTTP flows and Non HTTP flows. Training and test data are from R-III trace (see table III)

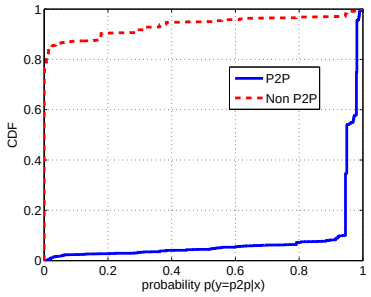

Fig. 2. CDFs of the probability of being a P2P flows for P2P flows and Non HTTP flows. Training and test data sets are from R-III trace (see table III)

The choice of $th = 0.5$ is very conservative, as the logistic regression has a strong discrimination power. For example, Figure 2 shows the cumulative distribution functions of the probability $p(y = p2p|x)$ for P2P and non-P2P flows in one of the trace used in Section V. A choice of th corresponds to a vertical line at value th on the x-axis. Figure 2 shows that the classification in p2p/non-p2p is almost unaffected by the exact th value. Indeed, more than 80% of non-p2p flows have a probability to be p2p flow less than 0.01, and more than 90% of p2p flows have a probability of being a p2p larger than 0.95. This is even more pronounced in the case of HTTP flows (Figure 1) where 99% of non-HTTP flows have a probability of being HTTP flows less than 0.005, and more than 90% of HTTP flows have a probability larger than 0.99. These figures show clearly that the choice of a larger threshold would change only slightly the classification results.

IV. Experiment setting

In this section, we present our data set, 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.

A. Data sets

Our data set consists of two recent packet traces collected at two *different* ADSL Points of Presence (PoPs) in France from the same ISP. Both traces were collected at *the same time* using passive probes located behind a Broadband Access Server (BAS), which routes traffic to and from the digital subscriber line access multiplexers (DSLAM) to the Internet. Captures were performed without any sampling or loss. Traces contains one hour of full bidirectional traffic, with similar number of active users. More details are provided in table I.

We restrict our attention to TCP flows as they carry the vast majority of the bytes in both traces. We are still left with the issue of defining the set of flows to be analyzed. Restriction is imposed by the classification method itself as we are using as features information derived from the first 4 data packets. We de facto exclude all flows with less than 4 data packets as well as the ones for which we did not observe the initial three way handshake. This typically leaves around 70% of volume for the analysis. Details about the impact of the flow definition on the amount of data excluded for each application class can be found in [14].

B. Application breakdown

In order to benchmark the performance of any classification method, a data set with pre-labeled classes of traffic is needed. We term such a data set our reference point. Establishing a correct reference point is fundamental when evaluating traffic classification mechanisms to provide trust-worthy results. As a human-labeled data set is almost impossible to have, we rely on DPI tools. In [15], We have compared an internal tool of Orange, that we term Orange_DPI_Tool or ODT for short, to Tstat [16], whose latest version features DPI functions. ODT and Tstat v2 offer similar performance and outperform signature based tools used in the literature [10], [4]. More details about the reference point issue can be found in [15].

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-III	2008-02-04	14:45	1h	36	1,3	54	91.9	2100	295 K

TABLE I
TRACES SUMMARY

Class	Application/protocol
WEB	HTTP and HTTPs browsing
HTTP-STR	HTTP Streaming
EDONKEY	eDonkey, eMule obfuscated
BITTORRENT	Bittorrent
GNUTELLA	Gnutella
CHAT	MSN, IRC, Jabber
	Yahoo Msn, HTTP Chat
MAIL	SMTP, POP3, IMAP, IMAPs
	POP3s, HTTP Mail
FTP	Ftp-data, Ftp control
GAMES	NFS3, Blizzard Battlenet, Quake II/III
	Counter Strike, HTTP Games
STREAMING	Ms. Media Server, Real Player
	iTunes, Quick Time
OTHERS	NBS, Ms-ds, Epmap, Attacks
UNKNOWN	-

TABLE II
APPLICATION CLASSES

Traffic classes recognized by ODT are summarized in Table II. Breakdown of traffic is presented in Tables III and IV. Traffic proportions are very different in both locations even though both traces were collected in the same country and at the same time. Web and eDonkey are the dominant classes in terms of flows while in terms of bytes, these are Web, eDonkey and HTTP streaming, the latter reflecting the popularity of streaming service providers like YouTube. While HTTP traffic is broken into many classes, it is important to note that the most important ones for HTTP applications in our data sets are Web browsing, HTTP-streaming and HTTP chat. We will term those three categories as HTTP in Section V, neglecting the minority of HTTP flows in the mail and games classes.

TABLE III
TRAFFIC BREAKDOWN RIII. FOR FLOWS $\geq$ 4 DATA PACKETS

	Flows		Size	
	Number	%	MB	%
WEB	160802	49.16	5519.56	24.61
HTTP-STR	4282	1.31	2654.14	11.84
EDONKEY	119057	36.40	8295.35	36.99
BITTORRENT	8789	2.69	1529.69	6.83
GNUTELLA	4718	1.44	1093.83	4.89
CHAT	4365	1.33	46.66	0.22
MAIL	4206	1.29	244.47	1.10
STREAMING	679	0.21	451.09	2.02
FTP	437	0.13	156.06	0.71
GAMES	182	0.06	3.87	0.02
OTHERS	835	0.26	12.54	0.07
UNKNOWN	18501	5.66	2248.00	10.03

TABLE IV
TRAFFIC BREAKDOWN MSI. FOR FLOWS $\geq$ 4 DATA PACKETS

	Flows		Size	
	Number	%	MB	%
WEB	319009	78.91	8368.85	52.41
HTTP-STR	6901	1.71	2777.43	18.72
EDONKEY	23212	5.75	1106.59	9.06
BITTORRENT	2313	0.57	649.81	4.15
GNUTELLA	223	0.06	104.19	0.66
CHAT	7539	1.87	86.87	0.55
MAIL	18406	4.56	856.33	5.46
STREAMING	207	0.05	372.43	2.39
FTP	1129	0.28	470.52	3
GAMES	183	0.05	1.68	0.02
OTHERS	8803	2.19	196.23	1.25
UNKNOWN	13535	3.36	275.96	1.76

C. Flow Features

Most studies on traffic classification rely on statistics computed once all the packets of a flow have been observed, e.g., duration, number of packets, mean packet size, or inter-arrival time [17]. This clearly prevents any online classification. In contrast, we evaluate the feasibility of application identification in the early stage of a connection. A few works have tackled this challenge. In particular, [3] and [11] showed that statistical features extracted from the first k packets of each connection, where k is typically in the range of 4 to 5 packets, lead to a good overall classification performance. We however uncovered in [14] some weaknesses of those approaches related to the ability to detect some key applications like HTTP streaming, which is gaining in popularity and a data overfitting issue when one wants to apply a classifier on a trace collected on a location different from the one it was trained on. The latter situation could typically be the one of an ISP that trains the classifier on its major PoP, where DPI tools are available, before deploying it on its other PoPs. We will show in this section that logistic regression is able to overcome those weaknesses.

The choice of flow level features turns out to be a major task in traffic classification. As explained before, state of the art approaches often rely on a preliminary feature selection phase, e.g. the correlation based filter technique in [10], [12]. Such method outputs a single set of features which is *the same* for all applications. In contrast, logistic regression picks for each application of interest *distinct features* that best separates it from the rest of the traffic.

As we want to evaluate the ability of logistic regression to perform traffic classification on the fly, we selected an initial set of features that can be computed by the observation of the beginning of the flow: size and direction of the first 4 data

packets, presence of push flags and port numbers. Out of this set, logistic regression picks the most relevant features for each application. Size and direction of the first data packets have been shown to lead to good classification results in [3]. We enrich this set with a push flag indicator that indicates whether a data packet has its PUSH flag set or not.

We thus end up having a mix of quantitative and qualitative features. While logistic regression can handle both types of parameters, it is recommended to transform quantitative parameters into qualitative ones [6]. We proceeded as follows:

- **Size of data packets**: we classify each data packet as small or not small packet. We used a fixed threshold, derived from empirical distributions of packet sizes, of 200 bytes for all applications and all traces.
- **Port numbers**: the quantization technique used depends on the application of interest. For applications using the HTTP protocol, we assign the port variable to 1 if the source or destination port number belongs to the set $80, 8080, 443$ and 0 otherwise. For P2P applications, we assign the port variable to 1 if both the source and destination ports are above 1024. Note that other quantization strategies are possible. For instance, for p2p applications, one could have used legacy port numbers of considered p2p applications. It turned out however that the quantization technique we use, which makes no use of such a priori information, offers satisfactory results.

D. Performance metrics

We present results in terms of True Positives (TPs) and True Negatives (TNs) ratios. These notions are defined with respect to a specific class. Let us consider such a specific class, say the HTTP streaming class. TPs are the fraction of HTTP streaming flows that are labeled as such by the statistical classifier, i.e., logistic regression. TNs are the fraction of flows not labeled as HTTP streaming by our DPI tool that are also not labeled as HTTP streaming by logistic regression. For an ideal classifier, TPs and TNs should be both equal to 100%.

V. Evaluation

A. Feature Selection

For each application we estimated a logistic regression model, and using the statistical test presented in section III-D, we select the subset of features relevant to each application. The list of selected features is presented in Table V. We observe that the set of features kept for HTTP applications is (almost) the intersection of the ones kept for each individual HTTP applications. Indeed, logistic regression selects, for each application, the features that maximize the difference between the flows of this application and the rest of the flows in the datasets. When focusing on HTTP streaming, it might thus use most of the specific features used for detecting all HTTP applications and add a few additional ones (e.g., the push variable for the third data packet here) to further differentiate those flows from other flows. Conversely, when logistic regression has to handle all HTTP applications, it keeps only features that allow to distinguish those flows from the non HTTP flows in the data set, thus getting rid of features that might be important to specifically detect HTTP chat or HTTP streaming for instance.

Note that, while our features are enough to separate the p2p flows as a whole group from the other flows, as we will see in the next section, none of our features seem enough statistically significant to separate BitTorent flows from the remaining P2P flows. Indeed, the p-values (see section III-D) computed for relevance of our features are all larger the the significance level α. This might explain why in recent studies [14], [3], the classification of BitTorrent flows, using similar features set combined with other statistical algorithms, appears challenging.

B. Overall performance

For both traces we have, the logistic regression achieve overall TPs and TNs ratios over 98% and 97% respectively. These results are similar to the results obtained by most statistical classifiers, see [17]. This is because dominant applications like web or edonkey are well classified in all cases. A challenge in the traffic classification domain, is to be able to work at at different level of granularity, e.g., either groups of applications or specific applications. In the next sections, we will focus on two sets of applications: (i) applications that use the HTTP protocol like Web browsing, HTTP streaming (e.g., YouTube) or HTTP chat and (ii) p2p applications. In each case, we will evaluate the ability of logistic regression to either detect the whole family, e.g. all HTTP applications or specific members like HTTP streaming and the impact of parameters selection on the classification results.

Please note that in each experiment, including cross site case, we use 5% of flows for training and the remaining for testing.

For the cases where training and testing is done with the same trace, we only present results for our first trace as the results are highly similar for the two traces. A different scenario where training and testing is done on different traces will be discussed in Section V-E

C. HTTP driven applications

In this section we focus on the HTTP applications found in our datasets, namely: Web browsing, HTTP streaming and HTTP chat.

In Table VI, we present on the right column ('before selection'), TPs and TNs ratios for all HTTP applications taken together and each type of HTTP application in isolation for MS-I trace. We observe very high TPs and TNs for the 'All HTTP' and 'Browsing' cases and quite high values for 'HTTP streaming'. The latter result for HTTP streaming is a noticeable one as, to the best of our knowledge, no statistical classification technique has been able so far to isolate HTTP streaming traffic only – see for instance [14] where the features selected in [3] and [11] are used on data set MS-1 and lead to poor TNs results. However, the TN score obtained here is not high enough, since it means that 16% of flows from other classes are misclassified as HTTP streaming. Given

TABLE V
THE SET OF SELECTED FEATURES (WITH X) FOR EACH APPLICATION

	1st packet			2nd packet			3rd packet			4th packet			port number
	direction	push	size	direction	push	size	direction	push	size	direction	push	size	
All HTTP			X	X	X		X			X	X		X
Web		X	X	X	X		X			X			X
HTTP streaming			X		X		X	X		X	X	X	X
All P2P	X	X	X	X	X	X	X	X	X	X		X	X
eDonkey	X		X	X		X	X	X	X	X	X	X	X
BitTorrent													
Gnutella			X	X	X	X					X		X

that the number of HTTP streaming flow is fairly low, these misclassified flows in fact represent a significant fraction of the flows in the HTTP streaming class. Still, the good news is that all those flows are Web flows. This means that our features are enough to label the HTTP streaming flows as HTTP-based application, but not enough to separate them from Web. We leave for future work the search of additional features to better discriminate between Web and HTTP streaming flows. Our method, that enables to select the discriminative power of each feature for a particular application will be helpful to choose among potential candidate features.

The left column of Table VI shows results of logistic regression where only features corresponding to statistically significant β values are considered. We do observe no significant changes before and after the selection procedure. This reveals that logistic regression indeed gives no significance to the parameters that have no discriminative power for the considered applications or set of applications. Thus, we can safely remove the non relevant features without accuracy degradation which reduces the computational cost of the classification.

TABLE VI
THE PERCENTAGE OF TRUE POSITIVES (TP) AND TRUE NEGATIVES (TN) OF HTTP FLOWS USING ALL THE FEATURES (BEFORE SELECTION) AND ONLY THE FEATURES SELECTED BY THE ALGORITHM

	after selection		before selection	
	TP	TN	TP	TN
All HTTP	99%	99%	98%	99%
Web	98%	97%	98%	97%
HTTP streaming	83%	84%	83%	84%
HTTP Chat	94%	98%	94%	98%

D. P2P Application

In this section, we focus on the p2p applications observed in our datasets. In Table VII, we present results for the MS-I trace. Logistic regression achieves very good performance for p2p as a group as well as for eDonkey. Gnutella achieves a lower TPs ratio, which can be explained by the small number of Gnutella flows in our data set (only 223 flows, see Table IV). However, even in this case, we limit the risk of misclassifying a (non Gnutella) flow as Gnutella as the TNs ratio is very high. The only risk is to miss a small fraction of actual Gnutella transfers.

As pointed in the previous section, the set of features used is not diverse enough to discriminate BitTorent from the rest of the p2p applications, which leads to poor classification scores. Like for the case of HTTP streaming, we leave for future work the search of additional features to better discriminate BitTorrent traffic using our method to test the effectiveness of potential candidates.

TABLE VII
THE PERCENTAGE OF TRUE POSITIVES (TP) AND TRUE NEGATIVES (TN) OF P2P FLOWS USING ALL THE FEATURES (BEFORE SELECTION) AND ONLY THE FEATURES SELECTED BY THE ALGORITHM

	After selection		Before selection	
	TP	TN	TP	TN
All P2P	96%	95%	96%	95%
eDonkey	97%	96%	97%	95%
BitTorrent	–	–	67%	68%
Gnutella	83%	98%	83%	98%

E. Cross-site Evaluation

We performed a cross-site evaluation where, for each case (application or set of application), we train the classifier, using the selected features given in Table V, on one trace, e.g., MS-I and apply it on the other trace, e.g., R-III. Such a validation is important for practical usage of any classifier as it verifies whether the statistical model we build is representative of application and does not incorporate site dependent data.

We present the full cross-site results for our two traces in Table VIII. We did not present results for BitTorrent, due to its poor performance observed in the single site case. We observe good performance in all cases. The only exception is Gnutella when training is done on MS-I and testing on R-III. This is because we have only 223 Gnutella flows in trace MS-I, we apparently miss part of the diversity of this class. Note that when training and testing is done in the other direction, the TP ratio reaches 84%, as now we have a higher diversity in the training set. While this result was to be expected in the case of HTTP applications, it constitutes a major achievement in the case of p2p applications as it was demonstrated in [14] that a data overfitting issue could occur with p2p applications. The latter stems from the fact that the classifier learns ports used by p2p applications of local users, which then fool the classifier when the set of local users is changed. We attribute the good performance observed here with logistic regression to the quantization technique used for the port number that gets rid of specific port values but simply check if the two ports correspond to well-known ports or not.

To further investigate this hypothesis, we applied again logistic regression for each trace and for the cross site test

using the initial port number rather than its quantized version. As expected, we observed slightly worse performance on a trace basis and significant performance decrease in the cross site case. A striking example is the one of Gnutella whose TPs ratio decreases from 83% to 70% on R-III trace when no discretization is applied and from 84% to 42% when the logistic regression algorithm is trained on R-III and applied to MS-I.

As a conclusion, the ability of logistic regression to handle qualitative and not only quantitative values as well as per application feature selection enables us to minimize the risk of data over-fitting in cross site studies that were observed in previous work.

TABLE VIII
THE PERCENTAGE OF TRUE POSITIVES (TP) AND TRUE NEGATIVES (TN) IN CROSS CASE

	R-III to MS-I		MS-I to R-III	
	TP	TN	TP	TN
All HTTP	98%	99%	99%	99%
Web browsing	95%	91%	98.5%	96%
HTTP Streaming	80%	81%	90%	82%
HTTP Chat	75%	98%	75%	98%
All P2P	94%	91%	90%	95%
eDonkey	97%	95%	94%	96%
Gnutella	84%	98%	22%	99.7%

VI. CONCLUSION AND FUTURE WORK

In this paper, we have proposed a novel on-line classification algorithm based on the logistic regression model. It is a flexible classification framework that overcomes important weaknesses of state of the art methods proposed so far. We have validated the performance of the proposed methods using ADSL traffic traces obtained from a major French ISP. This method incorporates the following new features:

- It automatically selects the best possible subset of distinct features relevant to each (family of) application(s).
- It can be used for application based or protocol based classification. For instance, it can classify all P2P file-sharing at once, or focus on one of them only, e.g., eDonkey.
- It can handle both quantitative and qualitative features, while current approaches are able to handle quantitative features only. This is important as some features might be more useful when considered as qualitative rather than quantitative information.
- It can be made resilient to the data over-fitting problem encountered in cross-site studies: it can be trained on data collected on one location and used for traffic data from other sites. This turns out to be a very useful feature for companies or ISPs managing several sites.
- It has a constant and low computational cost as logistic regression boils down to comparing a linear combination of the flow features with a fixed threshold to take its classification decision.
- It can work in real-time as it needs to consider features extracted from the first four data packets of a transfer only to take an accurate classification decision.

We consider a number of future extensions to this work. We intend to carry a systematic study of selective features for key applications like BitTorrent or HTTP streaming with our method. Also, we have considered TCP traffic only so far. However with the growing trend of UDP traffic, we would like to generalize the method to handle UDP traffic as well.

REFERENCES

[1] RB Bendel and AA. Afifi. Comparison of stopping rules in forward regression. *Journal of the American Statistical Association*, pages 46–53, 1972.

[2] Laurent Bernaille, Renata Teixeira, Universit Pierre, and Marie Curie Lip-cnrs. Early recognition of encrypted applications. In *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, 2006.

[4] 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*, 2006.

[5] James W. Hardin and Joseph W. Hilbe. *Generalized Linear Models and Extensions, 2nd Edition*. StataCorp LP, 2007.

[6] David W. Hosmer and Stanley Leshow. *Applied Logistic Regression*. New York ; Chichester ; Brisbane : J. Wiley and Sons, cop., 2001.

[7] T. Karagiannis, A. Broido, N. Brownlee, K.C. Claffy, and M. Faloutsos. Is p2p dying or just hiding? [p2p traffic measurement]. In *Global Telecommunications Conference, 2004. GLOBECOM '04. IEEE*, pages 1532–1538 Vol.3, Nov.-3 Dec. 2004.

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

[9] Thomas Karagiannis, Konstantina Papagiannaki, Nina Taft, and Michalis Faloutsos. Profiling the end host. In *In Passive and Active Measurement conference (PAM 07)*, April.

[10] 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, 2008.

[11] 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.

[12] 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, 2005.

[13] 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.

[14] Marcin Pietrzyk, Jean-Laurent Costeux, Taoufik En-Najjary, and Guillaume Urvoy-Keller. Challenging statistical classification for operational usage : the adsl case. In *IMC '09: Proceedings of the 9th ACM SIGCOMM conference on Internet measurement*, 2009.

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

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

[17] 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.

[18] Jon Tucker and Dr Jon Tucker. Neural networks versus logistic regression in financial modelling: A methodological comparison. In *in Proceedings of the 1996 World First Online Workshop on Soft Computing (WSC1*, 1996.

[19] Jeffery T. Walker and Sean Maddan. *Statistics in criminology and criminal justice, Third Edition*. Sudbury, Mass., USA, Jones and Bartlett Publishers, 2009.