Traffic Classification through Joint Distributions of Packet-level Statistics

Alberto Dainotti and Antonio Pescapé
University of Napoli Federico II (Italy)
Email: {alberto,pescape}@unina.it

Hyun-chul Kim
Seoul National University (Korea)
Email: hkim@mmlab.snu.ac.kr

***Abstract*—Interest in traffic classification, in both industry and academia, has dramatically grown in the past few years. Research is devoting great efforts to statistical approaches using robust features. In this paper we propose a classification approach based on the joint distribution of Packet Size (PS) and Inter-Packet Time (IPT) and on machine-learning algorithms. Provided results, obtained using different real traffic traces, demonstrate how the proposed approach is able to achieve high (byte) accuracy (till 98%) and how the new discriminating features have properties of robustness, which suggest their use in the design of classification/identification approaches robust to traffic encryption and protocol obfuscation.**

I. Introduction

Network traffic classification is fundamental to build knowledge on the use of network links, but it is also a crucial functionality to impose security or quality-of-service policies, to perform accounting, and many other relevant tasks. Traditional approaches based on transport-layer protocol ports are becoming increasingly unreliable and Deep Payload Inspection (DPI) approaches using packet payload have to cope with increasing network link speed and privacy issues. At the same time, finding alternative solutions has been demonstrated to be a non-immediate task. In recent years, research community and networking industry have investigated and developed several approaches. None of them solves the problem definitively and all of them show some drawbacks related to issues such as on-the-field applicability and reliability [1], [2], [3], [4], [5], [6], [7]. Moreover, while the state of the art is rapidly improving, Internet protocols and applications are continuously evolving (encryption is a notable example closely related to traffic classification) opening new challenges.

In this paper, we propose a novel approach considering the joint distribution of Packet Size (PS) and Inter-Packet Time (IPT). We apply a strong discretization to the estimate of their joint Probability Density Function (PDF) and using machine-learning algorithms like K-Nearest Neighbor (K-NN) and Support Vector Machines (SVM). Our approach is heavily based on the following research result: network traffic from different applications shows distinctive properties when the traffic is analyzed at packet-level, that is, in terms of PS and IPT. The major contribution of this study is represented by the introduction of effective packet-level features, which, as far as we know, have never been previously proposed in literature for the traffic claassification. We show that, in conjunction with a machine-learning classification algorithm, such features allow to build a traffic classifier able to achieve high (byte) accuracy (till 98%) and that looks promising in terms of robustness to evasion from identification. Several classification techniques based on machine-learning, indeed, heavily rely on *weak* features, e.g. related to the very first packets [4] [3] [5]. These features can be easily altered with the purpose of obfuscation, whereas the features here proposed are very dependent on the behavior of the application.

[0]This work has been partially funded by Seven One Solution srl and by LINCE project of the FARO programme jointly financed by the Compagnia di San Paolo and by the Polo delle Scienze e delle Tecnologie of the University of Napoli Federico II. Hyun-chul Kim was supported by NAP of Korea Research Council of Fundamental Science and Technology and the ITRC support program NIPA-2011-(C1090-1111-0004) of MKE/NIPA.

II. Classification Approach

A. Traffic view

We decompose network traffic into *biflows*, which represent an extension to the common definition of flows by considering traffic in both directions – respectively called *upstream* and *downstream*[1]. The biflows are considered expired after an inactivity timeout of 90 seconds [8]. Biflows allow to obtain better classification accuracy by exploiting the evident correlation between traffic in both directions, since generated by the same network application. Even if it is not always possible collecting traffic in both directions (e.g. on a backbone), it is straightforward to apply the same approach to unidirectional traffic but with a possible decrease in classification accuracy.

B. Statistical features

In past work [9], [10], [11], [12], [6] we observed – in accordance with literature [13] – that network applications exhibit (i) distinctive behaviors for marginal distributions and autocorrelations of PS and IPT; (ii) strong invariance of average profiles of PDFs when looking at traffic from different links and taken at different times (*space and time invariance*). By "average profiles" we mean that the PDFs of thousands flows have been averaged to a single PDF, however we observed this behavior also when looking at single profiles (i.e. separately considering the PDF of each single flow). Moreover, we found that IPT and PS of the same packet are usually very correlated [12]. This can be taken into account by considering the joint distribution of PS and IPT. We therefore developed a set of features based on this observation

[1]Upstream refers to packets sent by the host initiating the communication.

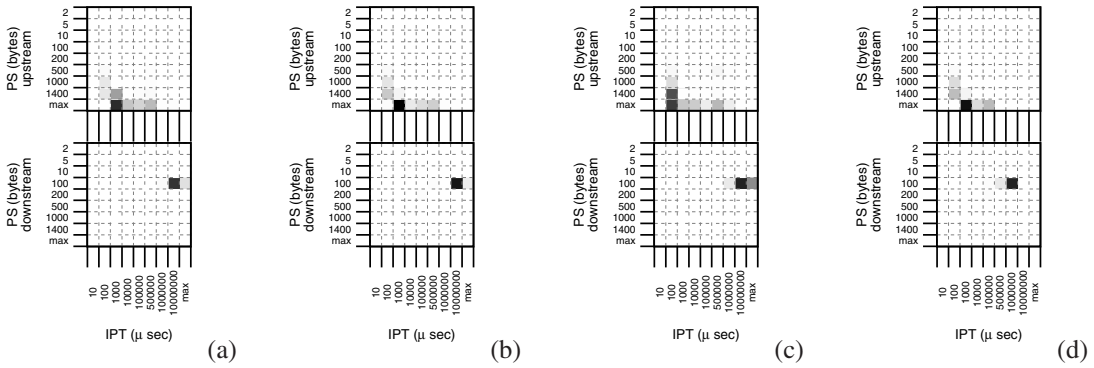

Figure 1. Sample fingerprints through joint PDFs of Edonkey traffic.

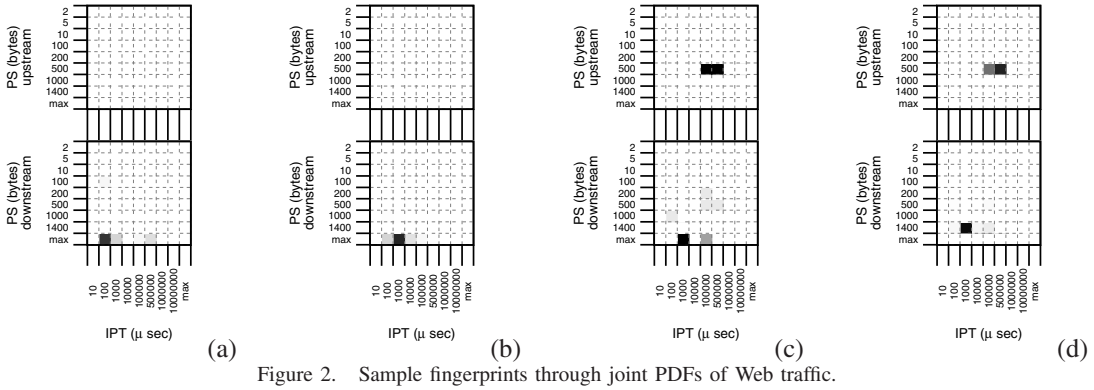

Figure 2. Sample fingerprints through joint PDFs of Web traffic.

to build traffic fingerprints of different network applications to be processed by machine-learning algorithms. Because we consider biflows, a *fingerprint* of the application will be built by considering the joint distributions of both directions. The machine-learning algorithms targeted however need as input-features a discrete set of data. We therefore need to identify a binning criterion for the joint distributions of IPT and PS. An ideal grid is overimposed to the plane identified by the PS axis and the IPT axis. A bin corresponds to each cell of the grid. The normalized "height" of each bin is a feature. Basing on results from our studies in traffic analysis we applied a non-uniform binning. For example, we considered the following upper boundaries (in bytes) for the binning of PS: $\{2,5,10,100,200,500,1000,1400,\infty\}$. Whereas for IPT we considered the following upper boundaries (in microseconds): $\{10,100,1000,10000,500000,1000000,10000000,\infty\}$. We developed a traffic processing platform [14] that assigns each observed packet to a biflow and to one of its two directions. It then updates the discretized joint PDF assigned to that direction by incrementing the counter associated to a specific bin. The corresponding bin is identified by evaluating the packet's PS and its IPT with respect to the previous packet from the same direction, and comparing such values against the aforementioned boundaries. At the end of the flow (or at program termination) the values inside the matrix of bin heights are normalized and dumped to a log file reporting other biflow information (unique ID, biflow-tuple, flow-level statistics, etc.). We can visually represent the two upstream and downstream matrices assigned to biflows as shown in Fig. 1 and 2, where we consider the binned joint PDFs of biflows. The height of each bin is indicated by the darkness of the corresponding cell and it represents the relative frequency of the packets into the associated range of PS-IPT values. The PS and IPT values, respectively on the x and y axes, refer to the upper boundaries of each cell. To give an intuitive idea of how such features could really be used as *fingerprints* for classifying traffic, in Fig. 1 and 2 we show random samples of Joint PDF for Edonkey and Web respectively. These figures show that samples from the two applications look quite different. Samples related to Edonkey present an upstream traffic mainly constituted of full-size packets with a majority of small IPT (traffic associated to a data transfer). On the opposite direction, instead, we observe small-size packets with large IPTs of the order of tens of seconds (traffic associated to a sort of control channel, e.g. subsequent requests for chunks of files). Looking at Fig. 2 we observe that samples (a) and (b) have a different behavior compared to (c) and (d). The first ones have an empty upstream distribution because the upstream direction was made of a single packet (thus no IPT values): traffic related to a single HTTP request (upstream) followed by the transfer of the requested file (downstream). Samples (c) and (d) instead are probably related to biflows with HTTP using *persistent connections*, in which subsequent requests are made inside the same TCP connection. Medium-size packets (HTTP requests often do not fill an entire packet) and IPT of the order of fractions of seconds to few seconds (requests generated directly by the browser or caused by user clicks) support our hypothesis.

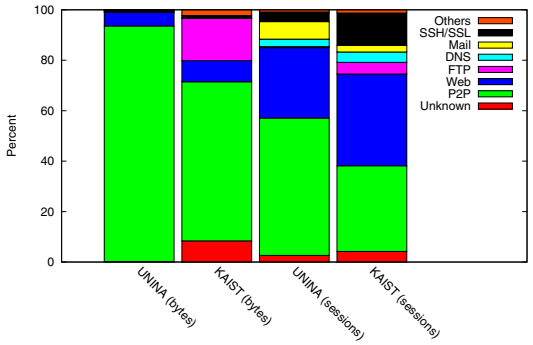

Figure 3. Application breakdown by percentage of biflows and bytes.

C. Machine-Learning Approaches

To test our approach based on the use of the traffic features described in Sec. II-B we focus on machine-learning algorithms performing supervised learning. We suppose indeed that the traffic classes are defined and that we have availability of pre-classified data to train our classifier. The objective of this study is to investigate the effectiveness of using packet-level features extracted from PS-IPT joint distributions and to identify an appropriate classification algorithm. For our experimental analysis we used the WEKA machine-learning software suite [15], often used in traffic classification studies [16], [17], [18], [7], [3], [1], [2] because it supports a large set of highly configurable machine-learning algorithms. To separate training and testing sets, 50% of each considered trace is chosen randomly to form a pool of training flows, and the remaining 50% is used for a pool of testing ones. After experimenting with several machine-learning algorithms we restricted our final experiments only on k-NN and SVM because we achieved best results with them. Here we briefly describe such algorithms:

k-Nearest Neighbors (k-NN) [19] computes Euclidean distances from each test instance to the k nearest neighbors in an n-dimensional feature space. The classifier assigns the majority class label among the k nearest neighbors to the test tuple. We use k = 1, 3, 5. k determines the number of training instances against which the algorithm checks the distance with the sample under classification.

Support Vector Machines (SVM) [20], [21], [22] The principle at the base of SVM is to construct a separating hyperplane that maximizes the distance between two sets of vectors (each set pertaining to a class) in an n-dimensional feature space [21]. Pairwise classification can be easily extended to multi-class problems in several ways. The parameters of the separating hyperplane with the maxium distance are derived by solving an optimization problem. In our setup we used the Sequential Minimal Optimization (SMO) [23], which decomposes the optimization problem into several 2-dimensional sub-problems that can be solved analytically instead of requiring numerical optimization. Two important parameters in SVM are the complexity parameter C and the polynomial exponent p [22], [20]. We use 1 for both of them as in [20].

III. Experimental Analysis

A. Traces and Datasets

We tested the proposed approach on two full real traffic traces. More precisely, datasets consisted of anonymized payload traces collected at two edge links located in Korea and Italy (Tab. I). The KAIST trace was captured at one of four external links connecting a 1 Gb/s KAIST campus network and a national research network in Korea. The UNINA trace was captured at a 1 Gb/s link connecting one of UNINA campus network to the national research network in Italy. For establishing the *ground truth* in order to evaluate our classification approach we used Crl_pay, a classifier based on payload inspection but also adopting some heuristics, which has been developed on top of the CoralReef suite [24] and made available by CAIDA. Crl_pay has been originally used in [25] and [26] (besides more recent work) and details about the techniques adopted are given in [27]. Crl_pay was augmented with more payload signatures, from [28], [18], [29]. Moreover we verified several results from the classifier through manual payload inspection. The applications recognized are summarized in Tab. II. We performed a filtering of data before training and testing machine-learning algorithms with them. We removed all biflows of very small size, most of them made of a single packet (for which therefore a joint PDF could not be built because of lack of IPT). Specifically, by removing all biflows with less than 10 packets for both directions we removed a high fraction of biflows (around 80%) from the data set that would only confuse (or make their work harder) the machine-learning algorithms, while still keeping 99.7% of total traffic in terms of bytes transferred. This was done with the purpose of (i) removing biflows generated by single packets, errors, scans, etc. (ii) relieving the load of the machine-learning algorithms (iii) focusing on the biflows that really weigh upon links traffic. Moreover, it is worth noting that more than 80% of the flows that were filtered out were classified as unknown or uncertain by Crl_pay, confirming that such biflows could have been counterproductive for our tests. Such filtering had also the effect of almost totally removing the biflows classified as unknown by the ground-truth software and thus not usable for training and testing. However, we must note that by excluding such categories of traffic (e.g. scans), such approach as developed and tested here, cannot be considered for the identification of different kinds of attacks (e.g. port scans, scanning worms), since we are focusing on traffic generated by applications carrying data.

Fig. 3 shows payload classification results for our traces after applying the filtering. This application breakdown is shown by grouping the applications into the categories shown in Tab. II, whereas the total number of distinct applications identified in each trace was around thirty.

B. Classification Results

We performed several sets of experiments, with different machine-learning algorithms and with different combinations of features. For the algorithms we considered: 1-NN, 3-NN, 5-NN, and SVM. The best results were always obtained by the 1-NN algorithm, closely followed by 3-NN and 5-NN,

Table I
CHARACTERISTICS OF ANALYZED TRACES.

Set	Date	Day	Start	Duration	Link type	Packets	Bytes	Biflows
KAIST	2006-09-14	Thu	16:37	21h 16m	edge	357 M	259 G	221K
UNINA	2008-05-16	Fri	10:42	18m	edge	52 M	40 G	42 K

Table II
APPLICATION CATEGORIES.

Category	Application/Protocol
web	http, https
p2p	FastTrack, eDonkey, BitTorrent, Ares Gnutella, WinMX, OpenNap, MP2P SoulSeek, Direct Connect, GoBoogy Soribada, PeerEnabler
ftp	ftp
dns	dns
mail/news	smtp, pop, imap, identd, nntp
streaming	mms(wmp), real, quicktime, shoutcast vbrick streaming, logitech Video IM
network operation	netbios, smb, snmp, ntp, spamassassin GoToMyPc
encryption	ssh, ssl
games	Quake, HalfLife, Age of Empires, Battle field Vietnam
chat	AIM, IRC, MSN Messenger, Yahoo messenger
unknown	-

Table III
CLASSIFICATION PERFORMANCE METRICS FOR UNINA TEST SET.

Class	TP Rate	FP Rate	Precision	Recall	F-Measure
P2P	0.944	0.072	0.943	0.944	0.944
Web	0.915	0.035	0.914	0.915	0.914
FTP	0.413	0.003	0.4	0.413	0.406
SSH	0.938	0.001	0.857	0.938	0.896
Mail	0.931	0.005	0.937	0.931	0.934
SSL	0.713	0.008	0.763	0.713	0.737
DNS	0.921	0.004	0.884	0.921	0.902
Chat	0.849	0.001	0.836	0.849	0.843

Table IV
CLASSIFICATION PERFORMANCE METRICS FOR KAIST TEST SET.

Class	TP Rate	FP Rate	Precision	Recall	F-Measure
P2P	0.913	0.05	0.909	0.913	0.911
News	0.931	0.001	0.901	0.931	0.916
Web	0.929	0.04	0.933	0.929	0.931
FTP	0.927	0.005	0.911	0.927	0.919
SSH	0.959	0	0.949	0.959	0.954
Mail	0.932	0.001	0.947	0.932	0.939
SSL	0.985	0.002	0.985	0.985	0.985
DNS	0.92	0.004	0.92	0.92	0.92
Streaming	0.676	0	0.742	0.676	0.708

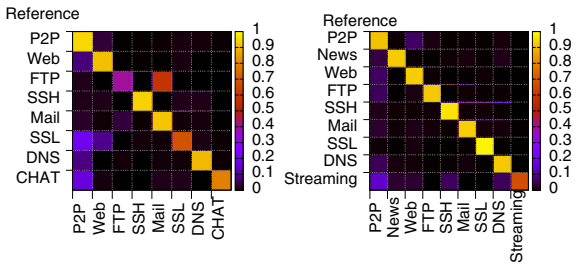

Figure 4. Confusion Matrices on UNINA (left) and KAIST (right) traffic.

which is congruent with the results in [2]. In our experiments, what 1-NN does is to assign the class label of the nearest neighbor in the 2-dimentional feature space of PS and IPT to the given test tuple. The result indicates (i) traffic biflows from a same application typically are generated in a way that they mostly use the same or most neighboring values for PS and IPT, and (ii) our design choices are right for accurate traffic classification; using biflows, PS and IPT for discerning traffic features, and the euclidean distance-based nearest neighbor algorithm. The SVM algorithm achieved an overall classification accuracy a few percentage points lower than the others. Therefore, unless otherwise specified, the results shown in this section have been obtained with the 1-NN algorithm and a feature set containing the values from the joint PDF matrices and two additional features: upstream-downstream packet ratio and biflow duration. For the first one, we used the ratio given by the number of upstream packets divided by the sum of upstream and downstream packets. As for the duration of the biflows, we applied a *log10* transformation to the values measured in milliseconds. These two features were added because such information related to the packets transferred cannot be derived by a joint distribution (e.g. the information on the number of packets is lost when computing the PDF). It is worth to note that overall accuracy does not decrease more than 2% when excluding these two features. To measure the performance of classification on the test set, we use four metrics: *Overall accuracy* is the ratio of the sum of all True Positives (TP) to the sum of all the True Positives and False Positives (FP) for all classes[2]. We apply this metric to measure the accuracy of a classifier on the whole test set. The latter three metrics are used to evaluate the quality of classification results for each application class instead. *Precision* is the ratio of objects properly attributed to a class over the total number of biflows attributed to that class. *Recall* is the percentage of biflows from a given class that are properly attributed to that class; and *F-Measure* is calculated as: $2x precision x recall/(precision + recall)$, this last metric is useful to rank and compare the per-class performance of different classification algorithms. Moreover, we compute overall accuracy also in terms of bytes (*byte-accuracy*) in order to evaluate the ability of the classifier to accurately identify the vast majority of traffic running on a link. Indeed, because biflows can be made of a very variable number of packets, it is important to evaluate accuracy by also considering the *weight* of each biflow (per bytes or per packets). We define overall byte-accuracy as the ratio of the sum of all bytes carried by the correctly classified objects to the sum of all bytes in the traffic considered[3]. Finally, we also consider the *confusion matrix* to better understand classification results and to identify which kind of misclassifications most frequently happen. Since the classification features here proposed are closely related to the typical behavior of the applications in terms of the traffic that they generate, we expect that applications supporting the same kind of service behave similarly and thus present

[2]True Positives is the number of correctly classified biflows, False Positives is the number of flows falsely ascribed to a given application, and False Negatives is the number of flows from a given application that are falsely labeled as another application.

[3]Accuracy *per-packets* instead of *per-bytes* is very close to the latter.

Table V
CLASSIFICATION ACCURACY.

Trace	Overall Accuracy	Overall Byte-Accuracy
UNINA	92.3%	98%
KAIST	92.9%	87%

similar features. For this reason we grouped the applications considered into the categories reported in Tab. II, each category corresponds to a class (a common approach in literature [2]). The high values of overall accuracy shown in Tab. V confirm our intuition. Especially in the case of the byte-accuracy achieved for the UNINA trace, the tested approach is very successful in correctly classifying the entire traffic contained in our traces. We suggest that the difference in byte-accuracy between the two traces can be explained with the misclassification in the KAIST trace of few P2P biflows carrying large quantities of bytes. This hypothesis is consistent with the performance metrics of the *P2P* traffic class obtained for the two traces and reported in Tab. III and IV. In Fig. 4 the confusion matrices for UNINA (left) and KAIST (right) dataset are represented in graphical form. Each row represents how a single class (reference on y axis) is classified by the algorithm (prediction on x axis). The confusion matrix is built by counting the biflows for each cell and by normalizing each row to 1. Values on the main diagonal represent the percentage of correctly classified biflows for each class. Both matrices show how the classifier performs excellently (yellow cells on the main diagonal) for almost all traffic categories. However, as for the KAIST trace, we note that the Streaming category is confused with P2P traffic in several cases. This is confirmed by looking at Tab. IV, where Streaming is the only class with values of precision, recall, and F-Measure below 0.9. The worst performance, in the case of the UNINA trace, is achieved for the FTP category, which presents all three metrics around 0.4. Chat and SSL also show some problems. As regards FTP, the confusion matrix reveals that it is often misclassified as MAIL traffic. Our explanation of this phenomenon is that most of the misclassified biflows carry FTP signaling sessions, whose behavior is indeed very similar to some Mail sessions (e.g. POP) when they do not transfer much data. This hypothesis is confirmed by two observations: (i) the opposite misclassification does not happen: the Mail category is not often misclassified as FTP; (ii) despite of the considerably low value of precision for FTP traffic, the overall byte-accuracy reaches a very high percentage, which suggests that the misclassified FTP biflows do not have a large byte-count. We conclude the analysis of experimental results by showing in Fig. 5 the confusion matrices that we obtained when exploding the P2P traffic category and separately considering up to eleven different P2P file sharing applications plus the *Probably P2P* class. From such graphics we notice two things: firstly, the overall classification accuracy decreases when considering separate applications instead of grouping them into categories. Secondly, most of the added *confusion* when exploding the P2P category happens among classes associated to P2P applications. The fact that confusion happens within classes of that category (P2P) confirms that

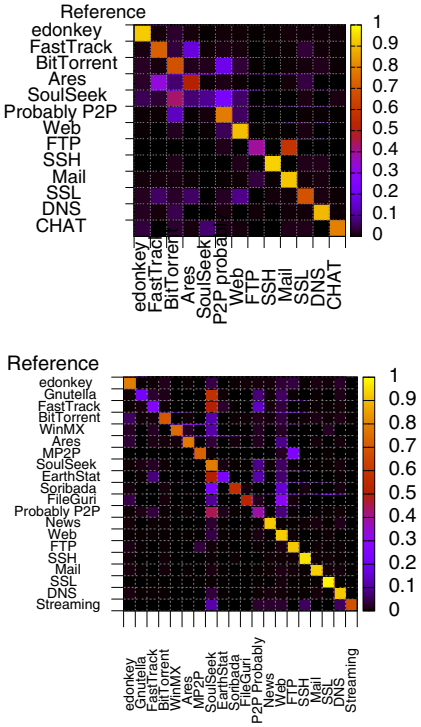

Figure 5. Confusion Matrices with P2P group exploded: UNINA (up), KAIST (down).

the classification approach here presented, and in particular the adopted traffic features, are able to catch the distinctive behavior of different categories of traffic rather than protocol or packet details that can be easily altered and are not necessarily shared by applications supporting the same kind of service (e.g. PS of the first 4 packets). We therefore look at such behavior as a symptom of robustness of the presented approach with respect to evasion techniques that change simple features (TCP flags, packet size of the first few packets, string-based signatures) in order to confuse classifiers. We indeed plan to investigate this aspect in future work, also with the intent to understand the impact of traffic encryption on such classification features.

C. Comparison with literature

Among the closest work in literature to ours, the one allowing at least comparison of metrics is [3], being the only one applied to the whole link traffic and reporting both session-accuracy and byte-accuracy. It is based on a Bayesian machine-learning approach, and among other features it uses also protocol ports (as mentioned, we do not use protocol ports as features, even if we verified that it increases classification accuracy). The best results there presented report an overall accuracy around 96% and byte-accuracy around 84%. Therefore, in terms of accuracy our approach is comparable to the one presented in [3], and even outperforms it when both are evaluated in terms of byte-accuracy.

IV. RELATED WORK

Features related to PS and IPT, and in particular related to their marginal distributions, for traffic classification purposes have already been presented in literature. However, to our

knowledge, the *approximation of the PS-IPT joint distribution* represents a novel set of features that has never been used in the field of traffic classification. In [30], 249 possible discriminators for classification of traffic flows are listed. Among them there are features like mean and standard deviation of PS or of IPT, minumum, maximum, quartiles of their distributions, etc.. Some of these features have been adopted in several papers [3] [31] [32]. However, not only the detailed approximations of the distributions are never listed and considered, but such PS and IPT were never taken into account *jointly*. Summary statistics like mean, median and standard deviation obviously do not contain the same information (and thus discriminative power) of approximations of the PDF. For example, by averaging two possible modes that can be present into an application – e.g. very small packets alternated to full-size packets are averaged into a mean PS of medium-size packets – we lose a lot of information (even if variance can give us some hints). Moreover the joint characterization of PS and IPT carries even much more information than the two separate distributions. For example, without a joint modeling we could not distinguish between an application generating full-size packets with small IPT and small-size packets with large IPT, and an application that associates IPT and PS in the opposite way. The results we provide in this paper confirm indeed the discriminative power of the features here considered. As regards the use in literature of similar features for traffic classification, we clarify that the features used in [4] are totally different. The joint distributions considered in that paper, indeed, represent another kind of information: the authors consider the order of packet arrivals for all the biflows analyzed, and build a joint distribution for each category of packets depending on their order of arrival, e.g. considering 5 packets per biflow would bring 5 joint distributions. The first of these joint distributions would therefore represent statistical properties of the *typical* first packet seen in the biflows of the application fingerprinted.

V. Conclusion

In this paper we proposed a novel machine learning approach for classifying Internet traffic using novel features that achieves high (byte) accuracy. Since results look promising, in our ongoing work we are: (i) developing a prototype implementation of the proposed approach as a TIE [33] classification plugin; (ii) we are further developing the set of features used by adding to the joint distributions of PS and IPT of each direction, information regarding behavior on the opposite direction (e.g. number of bytes transmitted in the opposite direction before the considered packet was received); (iii) specifically testing the robustness of this classification approach to traffic encryption and obfuscation techniques.

References

[1] N. Williams, S. Zander, and G. Armitage. A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification. *ACM SIGCOMM CCR*, 36(5):7–15, October 2006.

[2] H. Kim, KC Claffy, M. Fomenkov, D. Barman, M. Faloutsos, and K. Lee. Internet traffic classification demystified: myths, caveats, and the best practices. In *CoNEXT '08*, pages 1–12, NY, USA, 2008. ACM.

[3] A. Moore and D. Zuev. Internet traffic classification using bayesian analysis techniques. In *ACM SIGMETRICS*, June 2005.

[4] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli. Traffic classification through simple statistical fingerprinting. *ACM SIGCOMM CCR*, 37(1):7–16, January 2007.

[5] L. Bernaille, R. Teixeira, and K. Salamatian. Early application identification. In *ACM CoNEXT*, December 2006.

[6] A. Dainotti, W. De Donato, A. Pescapè, and P. Salvo Rossi. Classification of network traffic via packet-level hidden markov models. In *IEEE GLOBECOM 2008*, December 2008.

[7] 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. In *IEEE LCN*, November 2006.

[8] KC Claffy, Hans-Werner Braun, and George C. Polyzos. A parameterizable methodology for internet traffic flow profiling. *IEEE JSAC Special Issue on the Global Internet*, 1995.

[9] A. Dainotti, A. Pescapé, and G. Ventre. A packet-level characterization of network traffic. *CAMAD 2006*, 2006.

[10] Thomas Silverston, Olivier Fourmaux, Alessio Botta, Alberto Dainotti, Antonio Pescapé, Giorgio Ventre, and Kavé Salamatian. Traffic analysis of peer-to-peer iptv communities. *Computer Networks*, 53(4):470 – 484, 2009. Content Distribution Infrastructures for Community Networks.

[11] A. Dainotti, A. Pescapé, P. Salvo Rossi, G. Iannello, F. Palmieri, and G. Ventre. An hmm approach to internet traffic modeling. *2006 IEEE GLOBECOM*.

[12] A. Dainotti, A. Pescapé, P. Salvo Rossi, F. Palmieri, and G. Ventre. Internet traffic modeling by means of hidden markov models. *Computer Networks*, 52(14):2645–2662, 2008.

[13] Alice Este, Francesco Gringoli, and Luca Salgarelli. On the stability of the information carried by traffic flow features at the packet level. *SIGCOMM Comput. Commun. Rev.*, 39(3):13–18, 2009.

[14] *Plab*. http://www.grid.unina.it/software/Plab.

[15] *WEKA*. http://www.cs.waikato.ac.nz/ml/weka/.

[16] A. McGregor, M. Hall, P. Lorier, and J. Brunskill. Flow clustering using machine learning techniques. In *PAM*, April 2004.

[17] J. Erman, A. Mahanti, and M. Arlitt. Internet traffic identification using machine learning. In *IEEE Globecom*, Nov. 2006.

[18] J. Erman, M. Arlitt, and A. Mahanti. Traffic classificaton using clustering algorithms. In *ACM SIGCOMM MineNet Workshop*, September 2006.

[19] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield. Class-of-service mapping for qos: a statistical signature-based approach to ip traffic classification. In *ACM IMC*, October 2004.

[20] N. Williams, S. Zander, and G. Armitage. Evaluating machine learning algorithms for automated network application identification. Technical Report 060401B, CAIA, Swinburne Univ., April 2006.

[21] K. Bennett and C. Campbell. Support vector machines: Hype or hallelujah? *ACM SIGKDD Explorations*, 2(2):1–13, 2000.

[22] Z. Li, R. Yuan, and X. Guan. Accurate classification of the internet traffic based on the svm method. In *ICC*, June 2007.

[23] John C. Plat. Sequential minimal optimization: A fast algorithm for training support vector machines. Technical Report MSR-TR-98-14, Microsoft Research, April 1998.

[24] *CoralReef*. http://www.caida.org/tools/measurement/coralreef/.

[25] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. Blinc: Multilevel traffic classification in the dark. In *ACM SIGCOMM*, August 2005.

[26] T. Karagiannis, A. Broido, M. Faloutsos, and kc claffy. Transport layer identification of p2p traffic. In *ACM IMC*, October 2004.

[27] T. Karagiannis, K. Papagiannaki, and M. Faloutsos. Blinc: Multilevel traffic classification in the dark. Technical report, University of California Riverside, 2005.

[28] S. Sen, O. Spatscheck, and D. Wang. Accurate, scalable in-network identification of p2p traffic using application signatures. In *WWW*, May 2004.

[29] Y. J. Won, B. Park, H. Ju, M. Kim, and J. W. Hong. A hybrid approach for accurate application traffic idenficiation. In *IEEE/IFIP E2EMON*, April 2006.

[30] A. Moore, D. Zuev, and M. Crogan. Discriminators for use in flow-based classification. Technical Report RR-05-13, Department of Computer Science, Queen Mary, University of London, 2005.

[31] T. Auld, A. W. Moore, and S. F. Gull. Bayesian neural networks for internet traffic classification. *IEEE Transactions on Neural Networks*, 18(1):223–239, January 2007.

[32] S. Zander, T. Nguyen, and G. Armitage. Self-learning ip traffic classification based on statistical flow characteristics. In *PAM*, 2005.

[33] *TIE*. http://tie.comics.unina.it/.