Journal of Computer and System Sciences 80 (2014) 1021–1036

Contents lists available at ScienceDirect

Journal of Computer and System Sciences

www.elsevier.com/locate/jcss

Internet traffic clustering with side information

Yu Wang [a], Yang Xiang [a,*], Jun Zhang [a], Wanlei Zhou [a], Bailin Xie [b]

[a] School of Information Technology, Deakin University, Melbourne, Australia
[b] Cisco School of Informatics, Guangdong University of Foreign Studies, Guangzhou, China

ARTICLE INFO

Article history:
Received 25 September 2012
Received in revised form 15 March 2013
Accepted 27 August 2013
Available online 12 February 2014

Keywords:
Traffic classification
Semi-supervised machine learning
Constrained clustering

ABSTRACT

Internet traffic classification is a critical and essential functionality for network management and security systems. Due to the limitations of traditional port-based and payload-based classification approaches, the past several years have seen extensive research on utilizing machine learning techniques to classify Internet traffic based on packet and flow level characteristics. For the purpose of learning from unlabeled traffic data, some classic clustering methods have been applied in previous studies but the reported accuracy results are unsatisfactory. In this paper, we propose a semi-supervised approach for accurate Internet traffic clustering, which is motivated by the observation of widely existing partial equivalence relationships among Internet traffic flows. In particular, we formulate the problem using a Gaussian Mixture Model (GMM) with set-based equivalence constraint and propose a constrained Expectation Maximization (EM) algorithm for clustering. Experiments with real-world packet traces show that the proposed approach can significantly improve the quality of resultant traffic clusters.

1. Introduction

Internet traffic classification is the process of identifying network applications and classifying the traffic accordingly. It plays an important role in modern network management and security systems and in cloud computing environments. By obtaining full visibility into the types of traffic traversing through the network, accurate traffic classification enables fine-grained management of application performance and balanced utilization of the network resources.

Traditional traffic classifiers are port-based, which simply inspect the transport layer port number fields in packet headers and predict the upper layer application according to the list of well-known and registered port numbers maintained by Internet Assigned Numbers Authority (IANA). This simple approach is efficient since it only requires access to packet headers. However, it has become increasingly inaccurate in recent years due to the violation of port number assignments by more and more newly emerging applications. The alternative approach widely deployed in industry is payload-based, which performs deep packet inspection to either reconstruct and validate application protocol sessions or match protocol signatures against payload contents. These methods are accurate, but they not only impose significantly higher computational complexity, but also require specific knowledge of the target application protocols in advance. The a priori information, such as message

* Corresponding author.
E-mail addresses: y.wang@deakin.edu.au (Y. Wang), yang@deakin.edu.au (Y. Xiang), jun.zhang@deakin.edu.au (J. Zhang), wanlei@deakin.edu.au (W. Zhou), xiebailin96@126.com (B. Xie).

http://dx.doi.org/10.1016/j.jcss.2014.02.008
0022-0000/

formats and protocol signatures, is typically derived from protocol specifications or packet traces manually by network experts and the knowledge database has to be updated from time to time. This means that the approach involves heavy human intervention and cannot handle unknown applications automatically. Moreover, the classifiers fail when the access to payload content is unavailable, such as the case of classifying encrypted traffic.

Due to the limitations of the classic port-based and payload-based approaches, recent research efforts have been dedicated to developing alternative classification schemes. One of the major directions is to exploit packet and flow level characteristics, which capture the distinctive usage and interaction patterns of network applications, typically by applying machine learning (ML) techniques [1]. In this approach, a vector of features is extracted to describe each traffic flow, where the elements are some specific flow statistics such as the mean and variance of packet sizes or inter-packet times. The feature vectors are supplied as input to the machine-learning engine, in which two separate stages are involved. The first stage is offline learning where a classifier (e.g. a probabilistic model or a set of classification rules) is learned from a set of training data, and the second stage is online classifying in which the classifier is used to predict the application class of real-time traffic.

Depending on whether or not the training data set is fully labeled, two general types of learning methods are applicable: supervised learning (or classification) and unsupervised learning (or clustering). In supervised learning the training data is fully labeled and the goal is to find a mapping from input features to output classes. In contrast, unsupervised learning focuses on discovering patterns in unlabeled training data such that the flows with similar characteristics are grouped into clusters without any prior guidance from class labels. The resultant traffic clusters need to be labeled and transformed into a classifier for the online classifying stage. Clustering techniques are important since in practice it is very difficult and labor intensive to obtain a fully labeled data set. Besides, clustering has the potential to discover novel patterns that represent previously unknown applications. However, previous work [2–9] has shown that the application of some classic clustering algorithms, such as K-Means and EM, yielded relatively low accuracy.

In this work, we present a novel semi-supervised traffic clustering approach that generates highly accurate traffic clusters. The motivation is that there exists abundant side information describing partial equivalence relationships across flows in addition to the flow feature observations. For example, all concurrent flows that are connecting to the same destination IP address at the same port (i.e. the same endpoint) are typically using the same network application. This kind of side information can be efficiently derived by observing the 3-tuples of {Destination IP, Destination Port, Protocol} without knowing the actual class labels of the flows, and it can provide valuable guidance in the task of traffic clustering. In order to take advantage of the side information, we introduce the notion of set-based instance level constraint, which specifies that a particular set of equivalence flows have to be put in the same cluster during the clustering procedure. We then formulate the constrained clustering problem with the classic Gaussian Mixture Model (GMM) and apply a variant of the Expectation Maximization (EM) algorithm to fit the model with both the observed data and the equivalence constraints. Moreover, we also investigate the impact of unsupervised feature discretization for traffic clustering. The rationale is that some of the flow features are discrete in essence but they were measured and treated as continuous values in most of the related work. In particular, we use equal-frequency binning to quantize the numeric feature values.

To evaluate the proposed method, we use four real-world Internet traffic data sets taken from three locations around the world, including a packet trace captured in 2006 at the Internet edge of a university campus network, two collected in 2008 and 2009 at a trans-Pacific backbone link and the final trace recorded in a commercial ISP network in Australia in 2010. All of the data sets have either partial or full payload content that allows us to build the ground truth with high confidence. Several findings are made from the experimental results. First, the side information in terms of flow-level equivalence constraints is widely available in network traffic. Over 96.8% of flows across the data sets involve in the constraints. Second, an improvement of up to 8.5% in terms of overall accuracy can be achieved by incorporating the constraints into the clustering procedure. Third, by performing unsupervised discretization on features before clustering we can achieve a further improvement of up to 8.4% of flow accuracy. In short, the proposed approach significantly boosts the accuracy performance of Internet traffic clustering by incorporating the side information and performing feature discretization.

The remaining of the paper is organized as follows. A brief review of the related work on statistics-based traffic classification is presented in the next section. In Section 3, we discuss the proposed semi-supervised approach for Internet traffic clustering. The data sets used in the evaluations are described in details in Section 4, where an analysis of the side information in real-world traffic is also presented. Next is the experimental results given in Section 5. In Section 6, we discuss some practical issues of applying the semi-supervised traffic clustering approach in operational networks. Section 7 concludes this paper.

2. Related work

Clustering techniques have been applied in the context of Internet traffic analysis for a long time. In [2] Hernández-Campos et al. proposed using the triplets of (request data size, response data size, quiet time) for application-level communication modeling, such that taxonomy of important patterns in the traffic mix could be developed using hierarchical clustering methods. Similarly, McGregor et al. applied in [3] the Expectation Maximization (EM) algorithm for clustering packet traces based on a range of flow-level statistical attributes such as packet length and inter-arrival statistics, byte count and connection duration. The resulted clusters were then visualized and interpreted using radar charts. Although

the researches were dedicated to the aspects of traffic modeling and synthetic traffic generation without regard to classification, their interpretations showed that the clusters did comprise homogeneous traffic types such as web, bulk transfer, peer-to-peer and so on.

The effectiveness of utilizing the traffic clusters for the purpose of application classification has been investigated in later studies. Zander et al. applied in [4] the AutoClass program that implements a classic probabilistic clustering method based on mixture models and EM approximation. They reported that the produced clusters were in average 86.5% accurate across traces, according to the intra-cluster homogeneity metric defined as the percentage of flows of the dominating application in the cluster. In [5] Erman et al. took one step further by transforming those AutoClass produced clusters to a Nearest Neighbor (NN) classifier and comparing with the Naive Bayes classifier. The cluster-based classifier achieved up to 91% classification accuracy on a separate testing data set and outperformed its counterpart by about 9%. In [6], the authors continued to make a comparison between three clustering algorithms: K-Means, DBSCAN and AutoClass, where they produced comparable results in terms of cluster accuracy but AutoClass took significant longer processing time (4.5 hours) than its competitors (1–3 minutes). In [7], Bernaille et al. proposed an early TCP flow classification approach based solely on trivial packet level features, i.e. the payload size and direction of the first few packets in the flow (down to 4). They applied three methods including K-Means, EM clustering in the Euclidean space and Spectral clustering on Hidden Markov Models. They proposed a number of assignment and labeling heuristics to transform the learnt clusters into classifiers, which were able to classify known applications with 90% accuracy and identify the new ones with a 60% probability. In [8], the authors also applied their approach to encrypted applications and reported an accuracy rate over 85%.

Erman et al. [9] have proposed a hybrid approach referred to as semi-supervised learning, which fed a large amount of unlabeled flows with a few labeled ones to the learning system. The clustering procedure in their scheme was actually unsupervised (K-Means) and the labels were used in the following step to map the clusters to applications, which is quite a practical solution for cluster labeling. Our approach also falls in the category of semi-supervised learning, but it is totally different because firstly, the semi-supervision comes from background knowledge instead of class labels; and more importantly, the guidance applies to the clustering procedure itself in order to generate better clusters.

Provided a complete and fully labeled training set, it is straightforward to build a traffic classifier using the supervised learning techniques. A variety of methods have been applied to accomplish the task, including Linear Discriminant Analysis, Nearest Neighbor [10,11], Naive Bayes [12,32], neural networks [13], normalized threshold [14], Support Vector Machines [15,16] and decision trees [17,18]. Some efforts have been dedicated to the comparisons between algorithms [19], practical implementation issues [20], and investigations on feature properties, such as cross-site stability [21,22] and value types [23]. The last paper showed that by pre-discretizing the traffic features using a supervised algorithm (Ent-MDL) the classification accuracy can be improved, especially for the relatively weaker classifiers. The finding is quite interesting because most previous studies have treated flow features as continuous variables, although some attributes such as those related to packet length are discrete in essence. In this work, we extend the investigation on the impact of feature discretization to the case of flow clustering, where the supervised discretization methods are not applicable.

3. Semi-supervised Internet traffic clustering

Fig. 1 illustrates the basic modules involved in deploying an Internet traffic classifier using clustering techniques. First of all, the raw traffic in form of IP packets is aggregated into traffic flows according to the 5-tuples (i.e. {Source IP, Source Port, Destination IP, Destination Port, Protocol}) and the flow characteristics are computed. Next the classifier takes as input the feature vector for each flow and then returns a predicted class label. The online classifying stage is finished at this point and the labels would be used for management or security purpose (e.g. traffic engineering, QoS mapping and auditing). Moreover, we need to record the classified traffic to gather a training data set to update the real-time classifier. Sampling techniques can be utilized to select a proper subset of flows to trace, in order to save storage space and also improve the performance of the learning process (e.g. by avoiding the class imbalance problem). In the offline learning stage, the clustering module takes the recorded traffic flows as input and breaks them down into groups that share similar characteristics. Here the goal is to generate pure clusters, that is, ideally the flows in each cluster are from the same application. If some ground truth is available (e.g. labels obtained by the online classifier or manual inspection of a few flows in each cluster), we can then map the clusters to actual application classes. Finally, the resulted clusters are used to train an updated classifier. In this work, we focus on the clustering procedure highlighted at the bottom of Fig. 1, which is the heart of the whole scheme. In particular, we propose a semi-supervised approach that incorporates side information as constraints into the clustering procedure.

3.1. Flow observations

In Internet traffic classification, the basic data instances are IP flows, which consist of sequences of packets exchanged between pairs of particular endpoints for the purpose of inter-process communication across the Internet. We adopt the formal definition from the survey [1] and we also adopt a 900-second timeout for the flows without explicit tear-down.

Definition 1. ***Unidirectional flows*** are series of packets sharing the same five-tuples consisting of source and destination IP addresses, source and destination port numbers, and the transport layer protocol, i.e. {srcIP, dstIP, srcPort, dstPort, Protocol}.

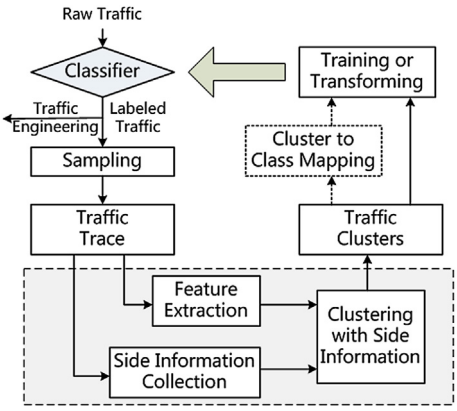

Fig. 1. Internet traffic classification using clustering.

Table 1
Flow feature set.

Characteristics	Descriptions	Count
Duration	Duration of bidirectional flow	1
Packets	Number of packets transferred in each direction	2
Bytes	Volume of bytes transferred in each direction	2
Packet size	Min., Max., Mean and Std. Dev. of packet size in each direction	8
Inter-packet time	Min., Max., Mean and Std. Dev. of inter-packet time in each direction	8
Total		21

Definition 2. ***Bidirectional flows*** (or simply ***flows***) are pairs of unidirectional flows going in the opposite directions between the same source and destination IP addresses and ports.

To allow clustering, each data instance is described by a feature vector $\boldsymbol{x} = (x_1, \ldots, x_d)^T$, which consists of the observed values on a predefined and fixed set of feature attributes. In this work, we define the attribute set with simple flow-level characteristics. In particular, we measure the flow duration, number of packets, volume of transferred bytes, and statistics on packet size and inter-packet time, as listed in Table 1. Except for flow duration, all attributes are separately computed in each direction of the flows. Note that we do not include any attribute related to payload and port numbers. Also, we exclude abundant TCP-specific features such as the statistics of various TCP flags and so on. According to previous studies [19], simple statistical characteristics have sufficient discriminating power and much lower computational overhead.

All the features are measured and treated as numeric values at first, but some of them are discrete in essence, such as those related to packet size. Therefore, we investigate the impact of feature discretization using an unsupervised approach called equal frequency binning. Specifically, for each feature we divide the range of all possible values into a series of intervals, each of which holds the same number of the observed data samples. The resultant intervals are sequentially numbered and the continuous feature values are replaced by the interval numbers. The process is a modification of feature granularity.

3.2. Side information

The observed traffic flows are not isolated. Some of them are initiated by the same sources, and some of them are visiting the same destinations. Therefore, given a set of unlabeled traffic data, we actually possess some side information in addition to the feature observations. In particular, according to the TCP/IP model, if some flows are connecting to the same endpoint on a node, i.e. they share the same 3-tuple of {dstIP, dstPort, Protocol}, then these flows are typically handled by the same application process behind. In other words, they are equivalent in regard to the source application. Fig. 2 gives two examples. In (a), three TCP flows initiated by host A are connecting to host B at port 80 at the same time. Similarly (b) depicts three concurrent flows towards TCP port 1863 of host D, which come from independent hosts. In these examples, we can infer that the correlated flows in each case belong to the same application. In order to model this kind of equivalence information among the flows, we define the equivalent flow sets as follows.

Definition 3. ***Equivalent flow sets*** are sets of bidirectional flows that share the same three-tuples of {dstIP, dstPort, Protocol}.

In this way we can describe the two examples in Fig. 2 as {B, 80, TCP} and {D, 1863, TCP}. Researches in semi-supervised machine learning [24] suggest that such side information could be expressed in the form of instance-level constraints and

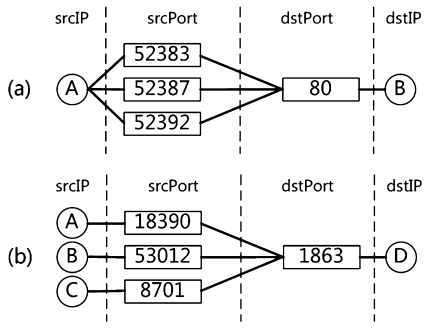

Fig. 2. Examples of equivalent flows.

be incorporated into the clustering procedure. In particular, two kinds of pair-wise constraints are widely adopted in the literature, which are defined as follows.

Definition 4. ***Must-link constraints*** specify that two objects have to be in the same cluster; ***Cannot-link constraints*** specify that two objects must not be placed in the same cluster.

Although the equivalence information among traffic flows can be expressed by must-links, we find it more natural and efficient to introduce a notion of set-based constraint, which specifies that the flows in an equivalence set have to be placed in the same cluster. In this way, an equivalent set consisting of N flow instances can be described using a single set-based constraint instead of $N(N-1)/2$ must-link constraints. Also, the extraction of constraints can be done in the course of identifying flows without introducing much computational cost.

Definition 5. ***Equivalence set constraints*** specify that the flows in each of the equivalent flow sets have to be placed in the same cluster.

Note that similar equivalence information has been used implicitly or explicitly in a number of related studies [25–27] that focus on payload-based or supervised traffic classification. For example, Karagiannis et al. [27] proposed BLINC, a classification system based on host-behavior, in which they introduced graphical application signatures (called graphlets) that capture the relationship between the use of source and destination ports, the relative cardinality of the sets of unique destination ports and IPs as well as the magnitude of these sets. Ma et al. [26] proposed a payload-based method for protocol inference, where they aggregated sessions into equivalence groups (referred to as cells) based on two similar assumptions. The cells are then presented to a hierarchical clustering engine as input. In [25], Canini et al. applied the 3-tuple heuristic to accelerate the flow labeling process for real-time classification. They validated the 3-tuple equivalence in their real-world traffic traces taken over multiple-day periods since 2003 until 2009. The approach to make use of the side information in our study is different from the above work in the following ways. First, our approach is based on flow characteristics instead of host-behavior, payload or port numbers. Second, we do not use the side information to directly generate signatures or rules that classify packets or flows. Instead, we use it to provide partial guidance in the form of constraints throughout the clustering procedure.

3.3. GMM with equivalence constraints

In this part, we model the flow observations and the side information using the classic parametric statistical model for data clustering, i.e. Gaussian Mixture Model (GMM). A GMM is described by:

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \sum_{i=1}^{K} a_i \, p_i(\boldsymbol{x} \mid \boldsymbol{\theta}_i), \tag{1}$$

where a_i is the mixing probability of each component, $p_i(\boldsymbol{x} \mid \boldsymbol{\theta}_i)$ is the respective Gaussian density function, $\boldsymbol{\theta}_i = (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ is the corresponding normal parameter vector, and K denotes the number of components in the model.

We apply the well-known EM algorithm to derive the maximum likelihood estimates of the GMM parameters [28]**.** Given the equivalence information, we know in advance that some data points are from the same source component. Therefore, the search space for the maximum expectation of likelihood is restricted. To derive closed-form solutions for the GMM parameter estimates, we adopt a slightly different data generation model. Specifically, we assume that the equivalence sets are independent and identically distributed (i.e. i.i.d.) with respect to the mixing probabilities and the flow instances within each set are generated i.i.d. according to the Gaussian component density.

Let $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ denote N observed unlabeled data points and let $X = \{X_1, \ldots, X_M\}$ denote M equivalence sets, where each set X_s consists of N_l data points $\{\boldsymbol{x}_1^s, \ldots, \boldsymbol{x}_{Ns}^s\}$ such that $\sum_{s=1}^{M} N_s = N$. Let $Y = \{y_1, \ldots, y_N\}$ denote the source assignments of the respective data points (i.e. $y_i \in \{1, \ldots, K\}$) and let Y_s denote the source assignment of equivalence set X_s. With the side information, the space of values Y can take on becomes:

$$\Omega = \left\{ Y \;\middle|\; \left(y_1^s = \ldots = y_{Nl}^s = Y_s\right),\ s = 1, \ldots, M \right\}. \tag{2}$$

Thus the expectation of the constrained complete-data log-likelihood is given by:

$$\begin{aligned} Q^c(\boldsymbol{\theta}, \boldsymbol{\theta}^g) &= E\left[\log p(X, Y \mid Y \in \Omega, \boldsymbol{\theta}) \;\middle|\; X, Y \in \Omega, \boldsymbol{\theta}^g\right] \\ &= \sum_{\boldsymbol{y} \in \Omega} \log p(X, \boldsymbol{y} \mid \boldsymbol{y} \in \Omega, \boldsymbol{\theta}) P\left(\boldsymbol{y} \;\middle|\; X, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g\right), \end{aligned} \tag{3}$$

where $\boldsymbol{\theta}^g$ is the current parameter estimate, Ω is the space of value $\boldsymbol{y}$ can take on subject to the constraints, and $\boldsymbol{\theta}$ is the new parameters to be optimized so as to increase the expectation.

In the E-step, we calculate the expected value given in Eq. (3) [29,30]. According to the data generation model, the constrained complete-data log-likelihood can be computed as follows:

$$\begin{aligned} \log p(X, y \mid y \in \Omega, \theta) &= \log p(y \mid y \in \Omega, \theta) p(X \mid y, y \in \Omega, \theta) \\ &= \sum_{s=1}^{M} \log a_{Ys} + \sum_{s=1}^{M} \log p(X_s \mid Y_s, y \in \Omega, \theta), \end{aligned} \tag{4}$$

and the marginal probability distribution for the hidden data can be computed as follows:

$$\begin{aligned} P\left(\boldsymbol{y} \;\middle|\; X, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g\right) &= \frac{P(\boldsymbol{y} \in \Omega \mid X, \boldsymbol{y}, \boldsymbol{\theta}^g) P(\boldsymbol{y} \mid X, \boldsymbol{\theta}^g)}{P(\boldsymbol{y} \in \Omega \mid X, \boldsymbol{\theta}^g)} \\ &= \frac{\prod_{s=1}^{M} \delta_{Ys} P(Y_s \mid X_s, \theta^g)}{\sum_{Y1} \cdots \sum_{YM} \prod_{j=1}^{M} \delta_{Yj} P(Y_j \mid X_j, \theta^g)}, \end{aligned} \tag{5}$$

where δ_{Yj} is defined as:

$$\delta_{Yj} = \begin{cases} 1, & \text{if } y_1^j = \cdots = y_{Nj}^j, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Based on Eqs. (4) and (5), Eq. (3) can be transformed into the following expression after some calculations:

$$\begin{aligned} Q^c(\boldsymbol{\theta}, \boldsymbol{\theta}^g) &= \sum_{s=1}^{M} \sum_{l=1}^{K} P\left(Y_s = l \;\middle|\; X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g\right) \sum_{n=1}^{N_s} \log p_l\left(\boldsymbol{x}_n^s \;\middle|\; \boldsymbol{\theta}_l\right) \\ &\quad + \sum_{s=1}^{M} \sum_{l=1}^{K} P\left(Y_s = l \;\middle|\; X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g\right) N_s \log a_l, \end{aligned} \tag{7}$$

where the posterior probability of each equivalence set is defined as:

$$\begin{aligned} P\left(Y_s = l \;\middle|\; X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g\right) &\equiv P\left(y_1^s = l, \ldots, y_{Ns}^s = l \;\middle|\; X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g\right) \\ &= \frac{\prod_{n=1}^{N_s} [a_l^g p_l(\boldsymbol{x}_n^s \mid \boldsymbol{\theta}_l^g)]}{\sum_{j=1}^{K} \prod_{n=1}^{N_s} [a_j^g p_j(\boldsymbol{x}_n^s \mid \boldsymbol{\theta}_j^g)]}. \end{aligned} \tag{8}$$

In order to maximize $Q^c(\boldsymbol{\theta}, \boldsymbol{\theta}^g)$, we can take its derivatives with respect to each parameter and obtain the following update rules for the M-step of the constrained EM algorithm:

$$\boldsymbol{\mu}_i = \frac{\sum_{s=1}^{M} P(i \mid X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g) \sum_{n=1}^{N_s} \boldsymbol{x}_n^s}{\sum_{s=1}^{M} P(i \mid X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g) N_s}, \tag{9}$$

$$\boldsymbol{\Sigma}_i = \frac{\sum_{s=1}^{M} P(i \mid X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g) \sum_{n=1}^{N_s} (\boldsymbol{x}_n^s - \boldsymbol{\mu}_i)(\boldsymbol{x}_n^s - \boldsymbol{\mu}_i)^T}{\sum_{s=1}^{M} P(i \mid X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g) N_s}, \tag{10}$$

$$a_i = \frac{1}{M} \sum_{s=1}^{M} P\left(l \;\middle|\; X_s, \boldsymbol{y} \in \Omega, \boldsymbol{\theta}^g\right). \tag{11}$$

Table 2
Traffic traces.

Trace	Date	Time	Duration	Network type	Link type	Packet	Byte	Payload
KEIO	2006-08-06	19:43	0.5 hour	Campus	Edge	27.0 M	16.99 G	40 bytes
WIDE08	2008-03-18	12:00	5 hours	Organization	Backbone	321.0 M	197.2 G	40 bytes
WIDE09	2009-03-31	12:00	5 hours	Organization	Backbone	361.4 M	224.2 G	40 bytes
ISP	2010-11-27	00:00	7 days	ISP	Edge	1270 M	665.7 G	Full

In summary, the constrained EM (Cons-EM) algorithm restricts the search space for the maximum expectation of likelihood, so as to find a GMM model that fits both the flow observations and the side information well. The algorithm begins with a random guess for the parameters and then iterates between the E-step and M-step until converges to a local maximum.

4. Side information in real-world traffic

4.1. Packet traces

In order to evaluate the availability of the side information and its value for traffic clustering, we use four real-world Internet traffic data sets that are carefully chosen regarding the following aspects. First of all, they are captured from three different network positions located around the world, such that the sampling points are heterogeneous in terms of link type and capacity. Secondly, the target networks have a diverse range of educational, organization and home users, which in turns generate different network usage patterns. Thirdly, the collection time of these traces ranges from 2006 to 2010, covering five recent years in which the Internet has grown and evolved rapidly. Lastly, either partial or full packet payload is preserved in these anonymized traces. This allows us to build the ground truth with high confidence using deep packet inspection techniques. Table 2 summaries the details of the data sets.

The KEIO and WIDE traces are all provided by the public traffic data repository maintained by the MAWI Working Group of the WIDE Project [31]. The KEIO trace is captured at a 1 Gbps Ethernet link in Keio University Shonan-Fujisawa campus in Japan and the collection date is the sixth of August in 2006 respectively. The WIDE08 and WIDE09 traces are taken at a US–Japan trans-Pacific backbone line (150 Mbps Ethernet link) that carries commodity traffic for WIDE member organizations. The original traces collected as part of the 'a Day in the Life of the Internet' project last 72-hour-long on 2008/03/18–20 and 96-hour-long on 2009/03/30–04/02. We use 5-hour subsets of them in our work. Forty bytes of application layer payload are kept for each packet while all IP addresses are anonymized in both KEIO and WIDE traces. The ISP data set is a full payload trace we captured at a 100 Mbps Ethernet edge link of a small-medium size Internet Service Provider (ISP) located in Australia. In this network, there are hundreds of regular home users and several internal servers that host web, mail and name services. The trace is seven days long starting from November 27 of 2010. For evaluation purpose, we focus on TCP traffic in these traces exclusively and consider only the flows that have successful TCP handshakes in the following analysis.

In order to evaluate the performance of the clustering algorithms, we establish the ground truth in the data sets using a customized deep packet inspection (DPI) tool that matches regular expression signatures against flow payload content. Two sets of payload signatures are developed based on previous experiences and some well-known tools such as l7-filter, Tstat and NeTraMark. The first set is designed to match against full flow payload and it is used to process the ISP traces. For the rest traces in which only 40 bytes of payload are available for each packet, we tune the second set of signatures to match against early message keywords. For the SSL encrypted flows, we classify them into HTTPS (443), POP3S (993), IMAPS (995) and SSL over other ports according to the standard server port number assignment. Moreover, a number of new applications are found by manual inspection of the unidentified traffic. Fig. 3 presents the application breakdown of TCP traffic in the traces. We identify fourteen most common application classes in these links: HTTP, BitTorrent, SSH, Razor, POP3, FTP, IMAP, DNS, SMTP, MSN, SMB, XMPP, HTTPS, and SSL. The OTHER class consists of the flows from the rest minor applications (such as NNTP, RTP, RSTP, RSTP over HTTP, RTMP, MSN over HTTP, AIM, IRC, X11 and RDP), while the UNKNOWN class represents the amount of traffic that could not be recognized.

4.2. Analysis of side information

Before applying the constrained approach to traffic clustering, the first question to answer is that how much side information exists in real-world Internet traffic. We measure the amount of side information across the traces by calculating the statistics of services and must-links. The results are listed in Table 3. In particular, we record the number of TCP flows, equivalence sets and those with more than two flows. The average and maximum equivalence set sizes (i.e. the number of flows in the sets) are also presented. The last two columns indicate the amount of must-links and the percentage of flows that can be linked to any other flow. For simplicity, we use a randomly selected two-hour subset here to represent the original one-week-long ISP trace.

From Table 3 we can make the following observations. First, abundant side information exists in all data sets. Millions of must-link constraints and thousands of set-based constraints are recorded in each trace. Second, a large proportion of flows in the traffic are involved in some correlation. The percentage of linked flows is 96.8%–98.9% across the traffic traces. In other words, only less than 3.2% of the flows show no relationship with others. Third, some sets are extremely large in size.

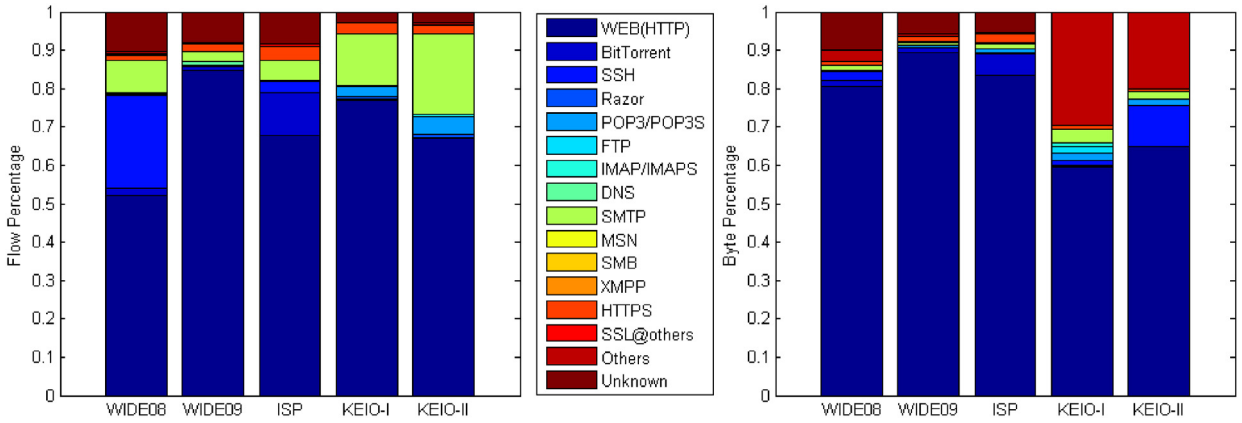

Fig. 3. Application breakdown.

Table 3
Side information in traffic traces.

Trace	Flow #	Eq. set #	Eq. set # (2 + flows)	Eq. set size (average)	Eq. set size (maximum)	# Must-link (million)	Linked flow (%)
KEIO	170 019	8241	5070	20.63	10 461	150.2	98.1%
WIDE08	2 701 771	47 607	18 366	56.75	89 233	19 874	98.9%
WIDE09	1 772 781	32 539	12 616	53.87	105 572	43 792	98.9%
ISP-2H	165 274	11 321	6041	14.60	6373	50.9	96.8%

Table 4
Side information per application in KEIO trace.

Protocol	Flow #	Eq. set #	Eq. set size (average)	Eq. set size (maximum)	Linked flow (%)
bt	640	132	4.8	19	95.6%
http	138 296	5584	24.8	10 461	98.7%
imap	432	16	27	210	99.3%
pop3	4439	243	18.3	988	98.9%
razor	1021	3	340.3	945	100%
smtp	23 172	1604	14.4	6247	95.6%
others	2019	659	3.1	124	83.4%

For example, in WIDE09 there exist a set of more than 100 thousand flows that are connecting to the same destination IP address at TCP port 80. Such a large constraint set generates 5 billion pair-wise constraints of must-link. In summary, the equivalence information of flows is widely available in network traffic, which is valuable to exploit.

Moreover, Table 4 provides a further analysis on the equivalence set distributions over various Internet applications obtained in the KEIO traffic trace and Fig. 4 shows the corresponding empirical cumulative distribution of the equivalence set size. Significant difference across classes can be noticed from them. First, all the RAZOR flows in the network were connecting to only 3 servers, which provide regular updates for the spam filtering client applications. Moreover, we can see that over 900 of the flows fall in one set (which could be the major server) and the other two sets hold only dozens of flows. Second, the two dominant protocols, i.e. HTTP and SMTP, in the trace exhibit similar patterns. They have large equivalence sets that comprise thousands of flows, and they also have a lot of flows visiting some unique services such that forming a lot of sets with only one sample. The other two traditional protocols, i.e. IMAP and POP3, have less traffic traversing across the networks, but the equivalence set patterns are actually quite similar to HTTP and SMTP as their average service sizes are moderate (range from 14 to 27). Finally, the peer-to-peer protocol BitTorrent shows quite a different behavior, as its traffic spreads widely to a lot of peers and each peer typically responds to a limited number of flows.

The last analysis of side information focuses on its utility for traffic clustering. Specifically, we investigate whether the unsupervised clustering methods can learn the knowledge by themselves, in other words, whether the clustering results are compatible with the side information. We use a metric called Inconsistency [24] to obtain quantitative measurements. Given a data set, we first present the equivalence sets as pair-wise must-link constraints. Then we generate partition on the data set using classic clustering algorithms including K-Means and EM, and calculate the fraction of the unsatisfied pair-wise constraints. Fig. 5 presents the results, which are obtained using training data sets randomly sampled from each trace (described in the next section). We can see that both K-Means and EM partitions show significant conflict against the side information. In particular, when we partition the data into ten clusters, 30%–50% of the must-link pairs of flows end up in different clusters, and the figure goes up to 50%–75% with a larger number of clusters ($K = 100$). These results lead to a

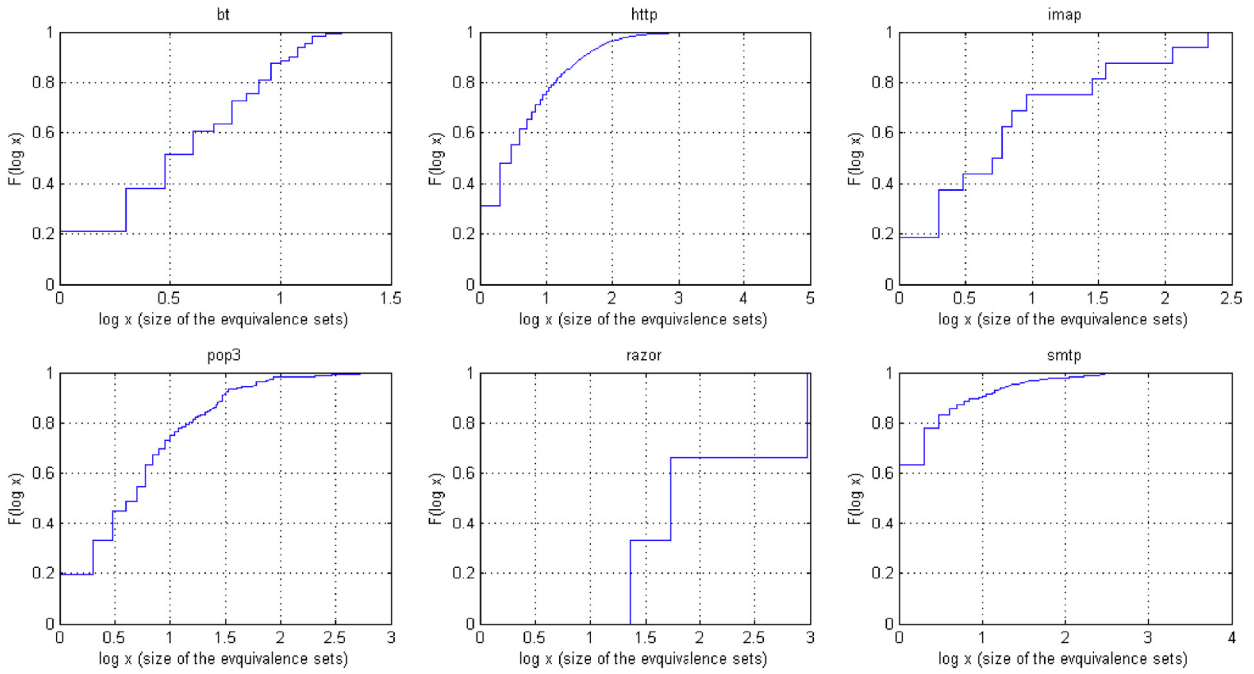

Fig. 4. Empirical CDF of the equivalence set sizes for each protocol in the KEIO trace.

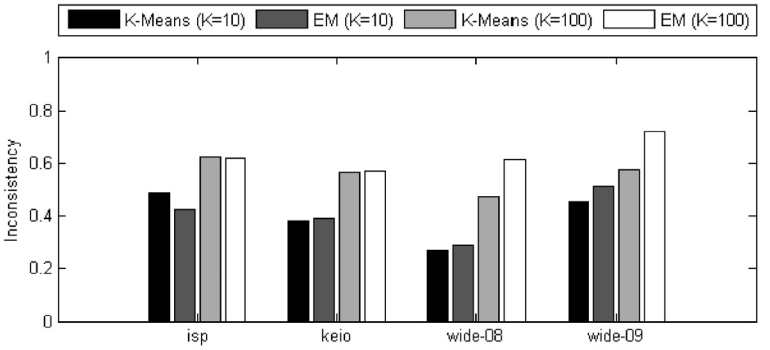

Fig. 5. Inconsistency between side information and unsupervised clustering results.

conclusion that the background knowledge can provide extra information, which cannot be completely learnt from the data itself by the unsupervised clustering algorithms.

5. Evaluation

5.1. Methodology

To evaluate the constrained clustering algorithm, we extract training sets from the traces using the following approach. For each trace, we randomly sample 1000 flows for each of the majority applications (those comprise vast majority flows in the traces). In order to be fully confident with the ground truth, we exclude encrypted traffic in the following analysis. In ISP trace, this sampling rule results in a training set consisting of 11 classes (i.e. HTTP, BitTorrent, DNS, FTP, IMAP, MSN, POP3, SMB, SMTP, SSH, and XMPP) and 11000 flows. The KEIO data sets have 6 thousand flows covering HTTP, BitTorrent, Razor, POP3, IMAP, and SMTP. The WIDE08 data sets have 8 thousand flows from HTTP, BitTorrent, Razor, POP3, FTP, IMAP, SMTP, and SSH. Finally, the WIDE09 data sets have 6 thousand flows representing HTTP, BitTorrent, POP3, IMAP, DNS, and SMTP. In order to achieve a greater confidence, we repeat the random sampling process for ten times to generate ten independent training sets for each trace.

The clustering algorithms in our evaluation take the number of clusters K as an input parameter. We run a series of clustering experiments on the training data using a range of values from ten to two hundred as follows: $K = 10, 20, \ldots, 190, 200$, and the accuracy results are reported as a function of K. For each setting, we repeat the clustering for ten times

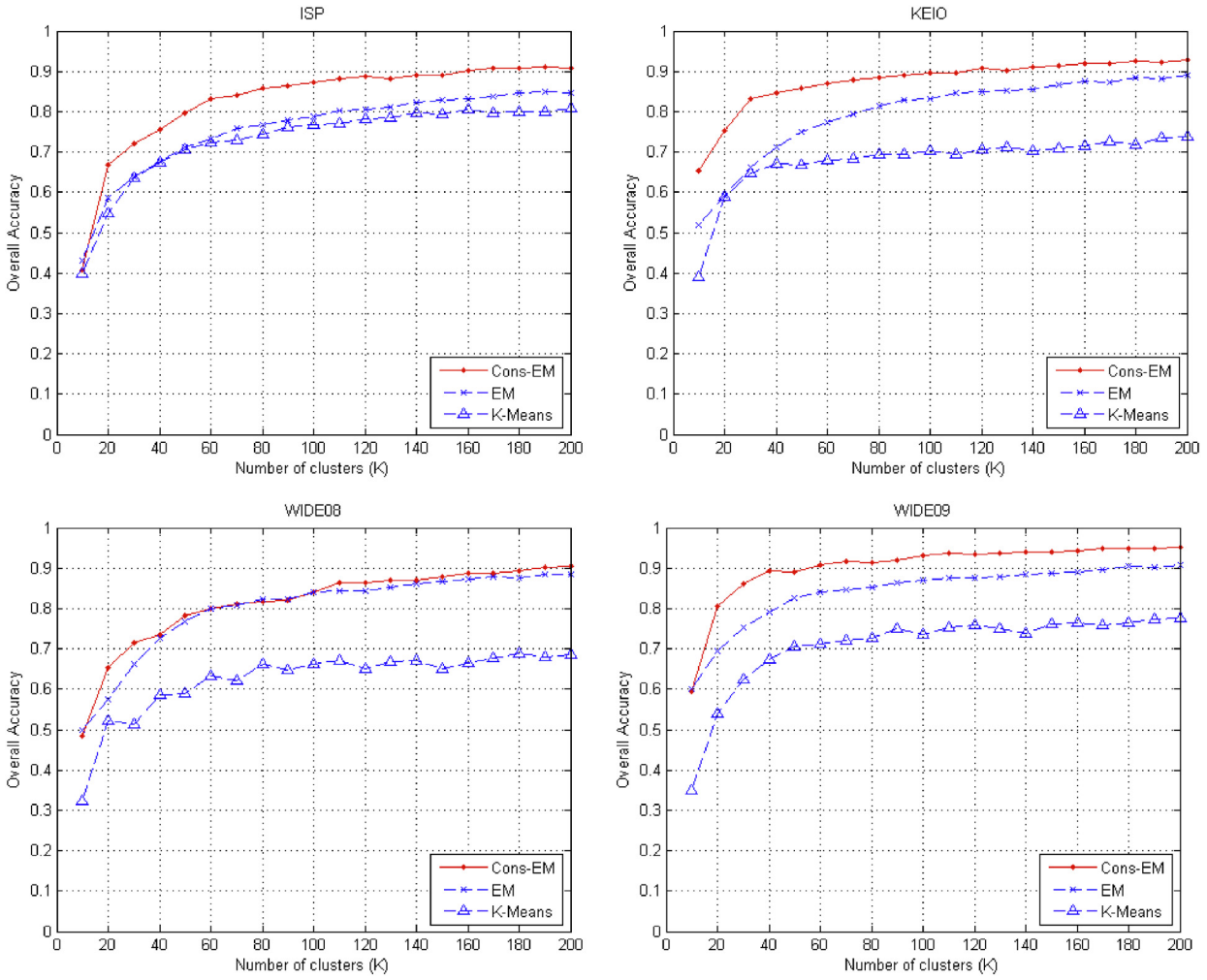

Fig. 6. Overall Accuracy results obtained in continuous feature space.

with different random seeds. Recall that we have selected ten random subsets from each trace, such that the results reported in the following sections are averaged on 100 runs.

In order to assess the clustering results, we use the majority heuristic to label the clusters based on the ground truth, i.e. we label the clusters with the dominant application presented in them. After labeling we measure the clustering results by the overall accuracy of the whole data set, and the per-class metric F-measure. Overall accuracy determines how pure the resultant clusters are. It is calculated as the ratio of the number of correctly labeled flows to the total number of flows in the whole data set. For each particular class, true positives (TP) is the number of its members that are correctly classified as belonging to it; false negatives (FN) is the number of its rest members that are incorrectly classified as belonging to other classes; and false positives (FP) is the number of members from other classes that are mistakenly classified as belonging to the present class. Then Precision is the percentage of the flows classified to the present class that are indeed its members, which is computed by $TP/(TP + FP)$; and Recall is the fraction of flows of the present class that are correctly classified as belonging to itself, which is computed by $TP/(TP + FN)$. Finally, F-measure appreciates a balance between precision and recall by taking the harmonic mean of them, i.e. $2 \times Precision \times Recall/(Precision + Recall)$.

5.2. Clustering in continuous feature space

Fig. 6 illustrates the overall accuracy results obtained in the continuous feature space. With the number of clusters setting to a small value (i.e. K = 10), the overall accuracy is 32.3%–39.9%, 42.9%–60.2% and 40.9%–65.4% across the data sets for K-Means, EM and Cons-EM respectively. The average accuracy for all algorithms rapidly rises as the value of K increases. This fast growing trend continues until K reaches a larger value (e.g. K = 100 in ISP data sets and K = 60 in WIDE09 data sets), where the overall accuracy for K-Means is between 66.1% and 76.7% across the four traces, and that of EM and Cons-EM is 78.8%–86.9% and 84.2%–93.2% respectively. Since then the improvement of accuracy for all algorithms

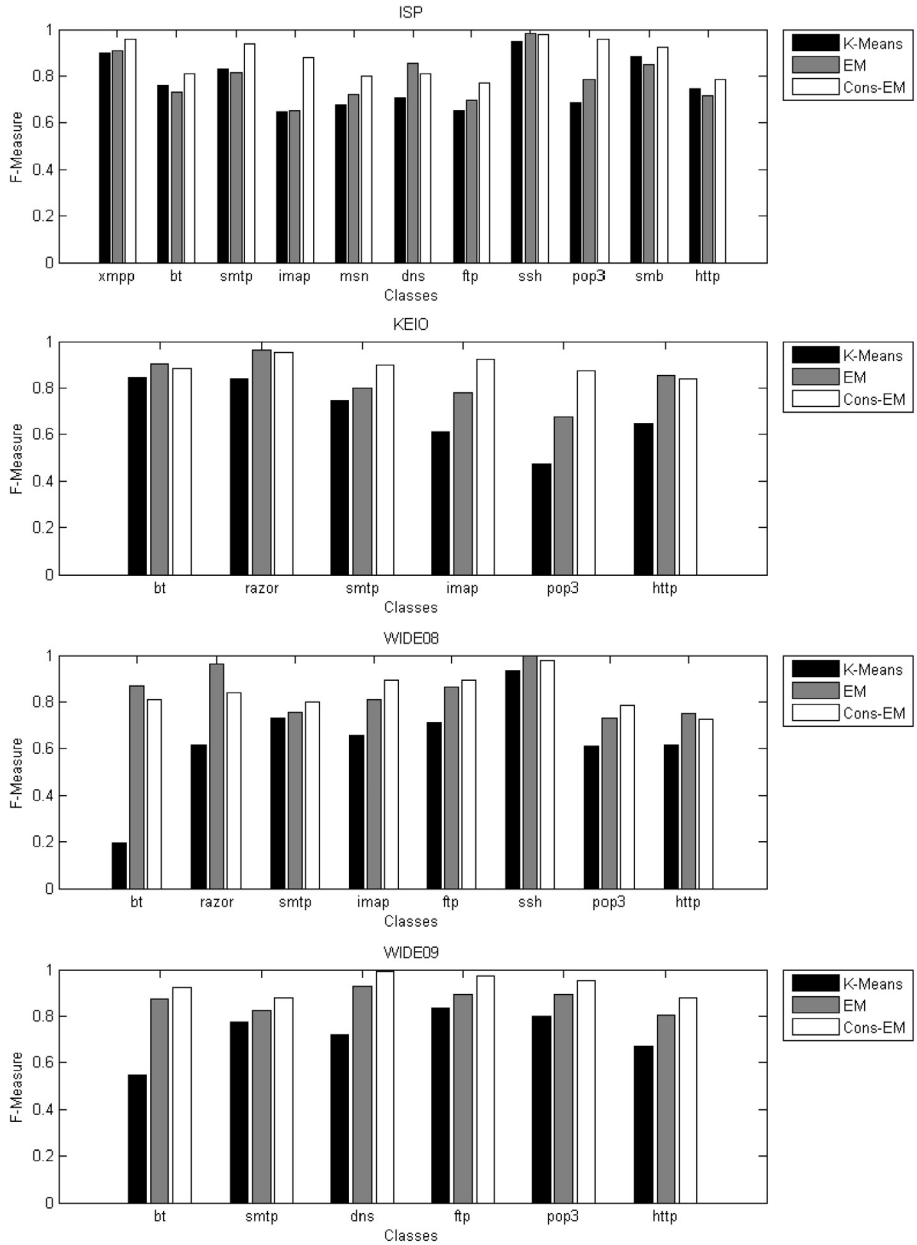

Fig. 7. Per-class F-measure results obtained in continuous feature space with K = 100.

becomes more gradual even if the number of clusters keeps increasing by dozens. Finally, when K = 200, K-means, EM and Cons-EM generate 68.4%–81%, 84.8%–90.7% and 90.5%–95.1% pure clusters respectively. In general, by incorporating the side information into the EM clustering algorithm, we can improve the overall cluster accuracy by 6%–8% (when K = 100), except for in WIDE08 trace where the improvement is not obvious (0.2%).

Fig. 7 presents per-class performance in terms of averaged F-measure obtained in the setting of K = 100. We first look at the ISP data set. For Cons-EM algorithm, the F-measure of XMPP, SMTP, IMAP, SSH, POP3, and SMB is better than 90%, while that of the other classes is around 80%. It outperforms the other two classic algorithms in almost all application classes. In KEIO data sets, the constrained clustering method significantly improves the F-measure performance in SMTP, IMAP, and POP3 classes while slightly decreases that in the other three classes. In WIDE08 data sets the Cons-EM and EM methods perform comparably and outperform each other in different classes, and thus they achieve similar accurate rates as shown in Fig. 6. Finally, in WIDE09 data sets, Cons-EM algorithm achieves the best F-measure results and outperforms the second best algorithm EM by about 10% in every application class.

In short, the results show that in the task of clustering based on the continuous flow features, the proposed constrained clustering method Cons-EM can generate more accurate traffic clusters than the classic K-Means and EM algorithms.

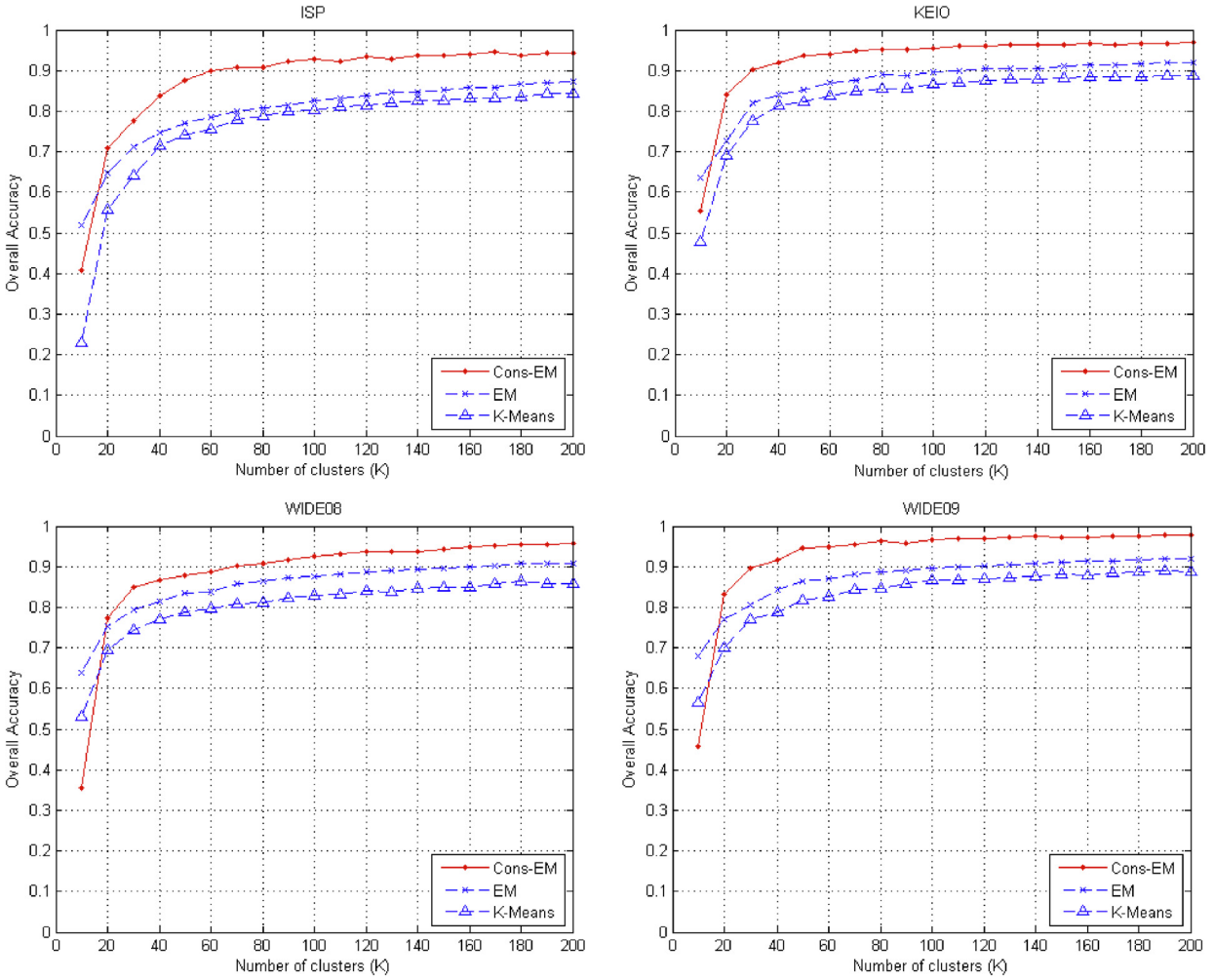

Fig. 8. Overall Accuracy results obtained in discrete feature space.

5.3. Clustering in discrete feature space

Fig. 8 gives the overall accuracy results obtained in the discrete feature space. Regarding the equal frequency binning discretization, we set the number of data samples each interval holds to be 50 in this part. More results of tuning this parameter will be reported in the next section.

We can see that the clustering accuracy rate achieved by all algorithms is better than that in the continuous feature space, while the convergence trend of accuracy with respect to the number of clusters is similar. In the case of clustering into a small number of clusters (i.e. $K = 10$), the overall accuracy rates of K-Means, EM and Cons-EM are 22.9%–56.7%, 51.8%–61.0% and 35.5%–55.5% across the data sets. All three algorithms become more accurate as the value of K increases. In the case of clustering into one hundred clusters, the overall accuracy rates can be raised to 80.2%–87.0%, 82.6%–90.1%, and 92.6%–97.8% for K-Means, EM and Cons-EM respectively. Further increase of the number of clusters does not help much in accuracy improvements. For example, raising the value of K from 100 to 200 can only enhance the accuracy of K-Means, EM and Cons-EM by 1.8%–4.1%, 1.9%–4.5%, and 1.1%–3.3% across the data sets.

In particular, we highlight two aspects of the accuracy results obtained by the proposed algorithm Cons-EM with discrete flow features. First, it is able to achieve very high cluster purity with the number of clusters setting to a relatively small value. For example, the overall accuracy reaches 90% when K increases to only 60 in ISP and WIDE08 data sets, and 30 in KEIO and WIDE09 data sets. Second, for the tasks of clustering to a larger number of clusters (e.g. $K = 200$), the average accuracy can be up to 94.2%–97.8% across traces, which is significantly higher than that achieved by the unsupervised approaches proposed in previous work.

Fig. 9 presents the average per-class F-measure results obtained in discrete feature space and with the number of clusters setting to be 100. We can see that in this case the proposed Cons-EM algorithm outperforms the other algorithms in every application class in all data sets. As an example, in ISP data sets Cons-EM achieves over 90% F-measure for most application

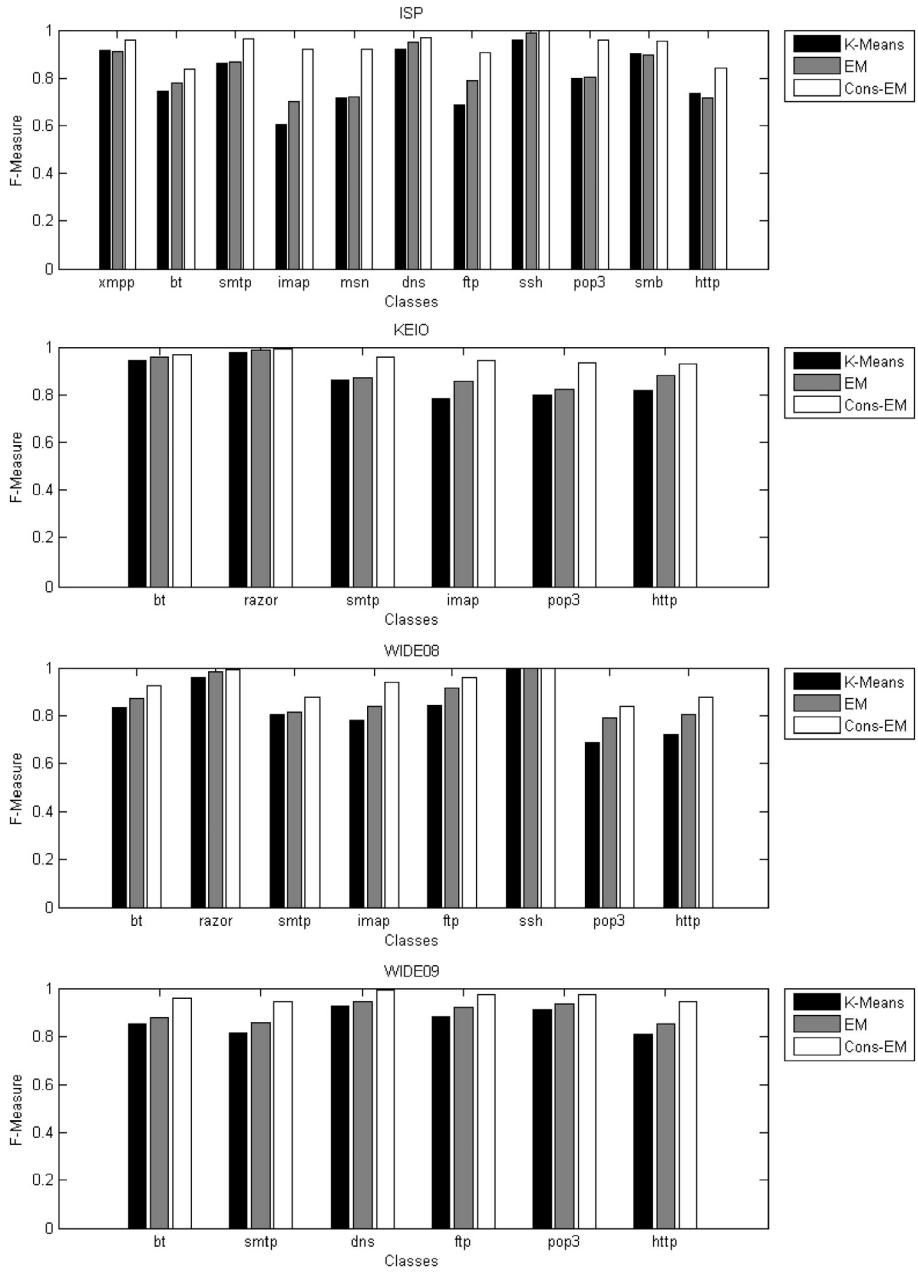

Fig. 9. Per-class F-measure results obtained in discrete feature space with K = 100.

classes except for BitTorrent and HTTP (around 80% and 85%), and the improvements are over 10% in the classes of SMTP, FTP, POP3, HTTP, IMAP, and MSN compared with the other algorithms.

5.4. Impact of feature discretization

In the above experiments, we use 50 as the default interval size in equal frequency binning, and the results show that this feature quantizing step does improve the performance of the clustering algorithms. In this section, we further investigate the impact of this parameter, which has been tuned to be 50, 100, 150 and 200. The results are presented in Fig. 10. Specifically, we show the differences of overall accuracy between clustering with discrete features and with continuous features, with respect to different parameters and different algorithms. Due to space limitation, we only present the results in KEIO and WIDE09 data sets.

Several interesting results can be found in Fig. 10. First, different values of the parameter show no noticeable difference on the improvements for Cons-EM and EM algorithms. For the case of K-Means, larger interval size tends to generate

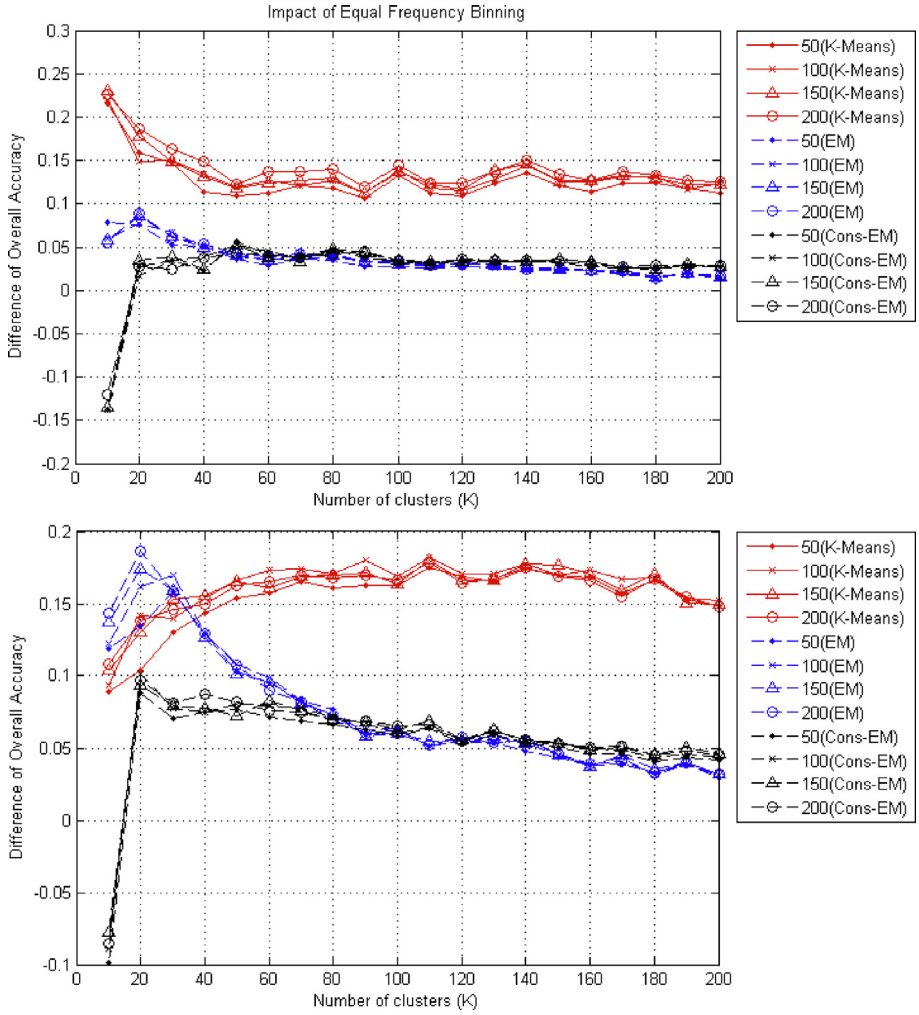

Fig. 10. Impact of feature discretization in WIDE09 (upper) and KEIO (lower) data sets.

slightly better performance, where an increase of 1–5% on overall accuracy can be seen by using 200 instead of 50. Second, the improvement is also related to K, the number of clusters. When the value of K is small (and the clustering accuracy is thus very low as shown in Fig. 6 and Fig. 8), feature discretization has greater impact on the results. On the other hand, as the value of K becomes larger (and the overall accuracy is higher and tends to converge), the influence of feature quantization is diminished and becomes stable. For example, in KEIO trace, feature discretization improves the accuracy of EM and Cons-EM by 6%, and that of K-Means by 17% when K equals to 100; these figures become 4%, 5% and 15% when K equals to 200.

In general, we summarize the following findings from the experimental results. On one hand, by quantizing flow features using the unsupervised equal frequency binning method, it is able to improve the overall accuracy of traffic clustering. On the other hand, by incorporating the side information into the clustering procedure, the proposed Cons-EM algorithm is able to achieve significantly better overall accuracy than previous methods including K-Means and EM. Take clustering the data sets into 100 clusters as an example, Cons-EM outperforms K-Means by 10.6%–19.8% and 8.9%–12.6% accuracy across traces in continuous and discrete feature space respectively; and the accuracy improvements against EM are 0.2%–8.5% and 5.1%–10.2% across traces in the continuous and discrete feature space respectively.

6. Remarks

In this study, we mainly focus on extending the network traffic clustering techniques to allow the incorporation of some background knowledge about the flow relationships. In the evaluations, we adopt the common experimental method [4–9], that is, to treat the number of clusters as a parameter and report the overall accuracy as a function of the value. In practice, the number of clusters is unknown in advance, and thus the best choice of which is a practical issue. It is widely believed that the number of clusters should be larger than the number of actual applications presented in the data, since it is

common for some applications to generate traffic with diverse patterns. There is a trade-off in the choice of an appropriate value. On one hand, having more clusters representing each application can provide a fine-grained view to its traffic and increase the overall clustering accuracy. On the other hand, a larger number of clusters also increase the computational complexity and the workload of post-processing such as labeling the clusters. From the experimental results given in Fig. 6 and Fig. 8, we can see that the overall accuracy tends to converge with the number of clusters increases to above 60. In the future work, we plan to explore the automatic estimation of the best value for cluster number with considerations of the side information.

To allow validating and extending, the data sets used in the current work will be made available to the researchers. Please visit the following link for more information: http://anss.org.au/tc.

7. Conclusions

In the task of Internet traffic clustering, it can be observed that there exists a large amount of side information indicating partial equivalence information among the traffic flows. In light of the observation, we have introduced a notion of set-based constraint to express the specific type of side information and proposed a semi-supervised clustering method with a constrained variant of the EM algorithm. Moreover, we propose to quantize the flow features by using the unsupervised equal frequency binning method in a preprocessing step to further improve the quality of clusters. Experimental results obtained from a number of real-world Internet traces have shown that the proposed method is able to significantly improve the performance of Internet traffic clustering in terms of both overall accuracy and per-class F-measure.

Acknowledgments

This work is partly supported by the Australian Research Council Discovery Project (Grant No. DP1095498), the National Natural Science Foundation of China (Grant No. 60970146 and No. 61202271) and the Guangdong Natural Science Foundation (Grant No. S2012040007184).

References

[1] T. Nguyen, G. Armitage, A survey of techniques for internet traffic classification using machine learning, IEEE Commun. Surv. Tutor. 10 (4) (2008) 56–76.
[2] F. Hernández-Campos, F. Donelson Smith, K. Jeffay, A.B Nobel, Statistical clustering of internet communication patterns, in: Proceedings of Symposium on the Interface of Computing Science and Statistics, 2003.
[3] A. McGregor, M. Hall, P. Lorier, J. Brunskill, Flow clustering using machine learning techniques, in: Proc. Passive and Active Measurement Workshop (PAM '04), Antibes Juan-les-Pins, France, April 2004.
[4] S. Zander, T. Nguyen, G. Armitage, Automated traffic classification and application identification using machine learning, in: IEEE 30th Conference on Local Computer Networks (LCN '05), Sydney, Australia, November 2005.
[5] J. Erman, A. Mahanti, M. Arlitt, Internet traffic identification using machine learning techniques, in: Proc. of 49th IEEE Global Telecommunications Conference (GLOBECOM '06), December 2006.
[6] J. Erman, M. Arlitt, A. Mahanti, Traffic classification using clustering algorithms, in: MineNet '06: Proc. 2006 SIGCOMM Workshop, pp. 281–286.
[7] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, K. Salamatian, Traffic classification on the fly, SIGCOMM Comput. Commun. Rev. 36 (2) (2006).
[8] L. Bernaille, R. Teixeira, Early recognition of encrypted applications, in: Proceedings of the Eighth Passive and Active Measurement Conference (PAM'07), April 2007.
[9] J. Erman, A. Mahanti, M. Arlitt, I. Cohen, C. Williamson, Semi-supervised network traffic classification, SIGMETRICS Perform. Eval. Rev. 35 (1) (2007) 369–370.
[10] M. Roughan, S. Sen, O. Spatscheck, N. Duffield, Class-of-service mapping for QoS: A statistical signature-based approach to IP traffic classification, in: Proc. ACM/SIGCOMM Internet Measurement Conference (IMC), 2004, Sicily, Italy, October 2004.
[11] Jun Zhang, Yang Xiang, Yu Wang, Wanlei Zhou, Yong Xiang, Yong Guan, Network traffic classification using correlation information, IEEE Trans. Parallel Distrib. Syst. 24 (1) (January 2013) 104–117.
[12] A. Moore, D. Zuev, Internet traffic classification using Bayesian analysis techniques, in: ACM International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS), Alberta, Canada, June 2005.
[13] T. Auld, A.W. Moore, S.F. Gull, Bayesian neural networks for Internet traffic classification, IEEE Trans. Neural Netw. 1 (January 2007) 223–239.
[14] M. Crotti, M. Dusi, F. Gringoli, L. Salgarelli, Traffic classification through simple statistical fingerprinting, SIGCOMM Comput. Commun. Rev. 37 (1) (2007) 5–16.
[15] A. Este, F. Gringoli, L. Salgarelli, Support vector machines for TCP traffic classification, Comput. Netw. 53 (14) (2009) 2476–2490.
[16] D. Schatzmann, W. Muhlbauer, T. Spyropoulos, X. Dimitropoulos, Digging into HTTPS: flow-based classification of webmail traffic, in: Proceedings of the 10th Annual Conference on Internet Measurement (IMC '10), ACM, New York, NY, USA, 2010, pp. 322–327.
[17] Y. Wang, S. Yu, Supervised learning real-time traffic classifiers, J. Netw. 4 (7) (September 2009) 622–629.
[18] Thuy T.T. Nguyen, Grenville Armitage, Philip Branch, Sebastian Zander, Timely and continuous machine-learning-based classification for interactive IP traffic, IEEE/ACM Trans. Netw. 20 (6) (December 2012) 1880–1894.
[19] N. Williams, S. Zander, G. Armitage, A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification, ACM SIGCOMM Comput. Commun. Rev. 36 (5) (October 2006) 7–15.
[20] S. Zander, G. Armitage, Practical machine learning based multimedia traffic classification for distributed QoS management, in: 36th Annual IEEE Conference on Local Computer Networks (LCN 2011), Bonn, Germany, 4–7 October 2011.
[21] M. Pietrzyk, J. Costeux, G. Urvoy-Keller, T. En-Najjary, Challenging statistical classification for operational usage: the ADSL case, in: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference (IMC '09), 2009, pp. 122–135.
[22] A. Este, F. Gringoli, L. Salgarelli, On the stability of the information carried by traffic flow features at the packet level, ACM SIGCOMM Comput. Commun. Rev. 39 (3) (July 2009) 13–18.
[23] Y. Lim, H. Kim, J. Jeong, C. Kim, T. Kwon, Y. Choi, Internet Traffic classification demystified: On the sources of the discriminative power, in: Proceeding of the ACM CoNEXT 2010.

[24] K. Wagstaff, S. Basu, I. Davidson, When is constrained clustering beneficial, and why?, in: Proceedings of the Twenty-Sixth Conference on Artificial Intelligence (AAAI-06), 2006.
[25] M. Canini, W. Li, M. Zadnik, A. Moore, Experience with high-speed automated application-identification for network-management, in: Proceedings of the 5th ACM/IEEE Symposium on Architectures for Networking and Communications Systems, pp. 209–218.
[26] J. Ma, K. Levchenko, C. Kreibich, S. Savage, G. Voelker, Unexpected means of protocol inference, in: IMC '06: Proc. 6th ACM SIGCOMM on Internet Measurement, Rio, Brazil, October 2006, pp. 313–326.
[27] T. Karagiannis, K. Papagiannaki, M. Faloutsos, Blinc: Multilevel traffic classification in the dark, ACM SIGCOMM Comput. Commun. Rev. 35 (4) (October 2005) 229–240.
[28] A. Dempster, N. Laird, D. Rubin, Maximum-likelihood from incomplete data via the EM algorithm, J. R. Stat. Soc. 39 (1977) 1.
[29] J. Blimes, A gentle tutorial on the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models, Tech. Rep. ICSI-TR-97-021, University of California Berkeley, 1998.
[30] N. Shental, A. Bar-Hillel, T. Hertz, Computing gaussian mixture models with EM using equivalence constraints, Adv. Neural Inf. Process. Syst., vol. 16, MIT Press, Cambridge, MA, 2004.
[31] MAWI Working Group Traffic Archive, http://mawi.wide.ad.jp/mawi/.
[32] Jun Zhang, Chao Chen, Yang Xiang, Wanlei Zhou, Yong Xiang, Internet traffic classification by aggregating correlated naive Bayes predictions, IEEE Trans. Inf. Forensics Secur. 8 (1) (January 2013) 5–15.