PAPER

Out-of-Sequence Traffic Classification Based on Improved Dynamic Time Warping

Jinghua YAN[†], Xiaochun YUN[††a)], *Nonmembers*, Hao LUO[†], *Member*, Zhigang WU[†], *and* Shuzhuang ZHANG[†], *Nonmembers*

SUMMARY Traffic classification has recently gained much attention in both academic and industrial research communities. Many machine learning methods have been proposed to tackle this problem and have shown good results. However, when applied to traffic with out-of-sequence packets, the accuracy of existing machine learning approaches decreases dramatically. We observe the main reason is that the out-of-sequence packets change the spatial representation of feature vectors, which means the property of linear mapping relation among features used in machine learning approaches cannot hold any more. To address this problem, this paper proposes an Improved Dynamic Time Warping (IDTW) method, which can align two feature vectors using *non-linear* alignment. Experimental results on two real traces show that IDTW achieves better classification accuracy in out-of-sequence traffic classification, in comparison to existing machine learning approaches.

key words: *traffic classification, out-of-sequence, dynamic time warping*

1. Introduction

Traffic classification is crucial for QoS provisioning, traffic scheduling, intrusion detection, and lawful interception of IP data. Traditionally, there are two categories in traffic classification technologies: port number based approaches and Deep Payload Inspection (DPI) approaches. The port number based approaches become inefficient or even misleading since many applications are increasingly using dynamically changed port numbers [1]. The DPI approaches rely on the inspection of the packet payloads and hence can classify traffic more accurately. However, they cannot deal with encrypted and tunneled traffic [2].

To overcome these limitations, a number of researchers apply machine learning techniques to traffic classification [3]. Machine learning techniques extract the statistical features of traffic, such as number of packets, flow duration, packet size or inter-arrival time. The machine learning techniques are divided into two stages: training a model based on features, and then applying machine learning algorithms to classify traffic. The works using machine learning techniques for traffic classification can be categorized into two classes according to to their level of observation:(1) flow-level techniques; and (2) packet-level techniques. The flow-level methods use features such as number of packets, flow duration, mean packet size, etc, while the packet-level methods use packet length, inter-arrival time and the packet direction. Flow-level techniques are less practical since all the features can only be calculated after the flow have completed [4], [5]. On the contrary, the packet-level methods can perform traffic classification in a real-time manner, since it only requires the first few (four or five) packets of a TCP connection [6], [7]. In this paper, we concentrate on packet-level techniques since it can classify traffic flows as fast as possible.

Although machine learning methods have been well studied and have shown good results on traffic classification, things are quite different when it comes to traffic with out-of-sequence packets. Most of the related works just neglected the out-of-sequence packets by removing the traffic flows with out-of-sequence packets. Generally speaking, a packet is said to be out-of-sequence if its sequence number is not equal to its expected sequence number. The network traffic may suffer from packet loss, packet reordering and packet retransmission, all of which will lead to out-of-sequence phenomenon. As previously proposed [8], [9], out-of-sequence is not a pathological phenomenon on the internet and is prevalent at significantly high levels. In recent years, a number of studies have concentrated on measured the out-of-sequence packets in the Internet, all of which have confirmed the prevalence of out-of-sequence in TCP and UDP network [10]–[13].

An intuitive way to classify traffic with out-of-sequence packets is utilizing the existing machine learning techniques. However, we observe that directly adopting these techniques will lead to poor performance, and the reasons are twofold:

1. The out-of-sequence packets alter the representation of feature vectors of traffic. In other words, the linear mapping relation among features no longer exists.
2. Existing machine learning approaches are based on the premise that the features are linear mapping.

Although these problems can be solved by packets reassembly technique, it will cause large processing latency and storage complexity. Moreover, the UDP packets do not contain sequence number information, thus the receiver cannot identify which packet is out-of-sequence. Therefore, the reassembly technique becomes invalid for UDP packets.

In this paper, we aim to address the out-of-sequence

Manuscript received October 26, 2012.
Manuscript revised May 25, 2013.

[†] The authors are with Beijing University of Posts and Telecommunications, Beijing, 100876 China.

[††] The author is with National Computer Network Emergency Response Technical Team, Coordination Center of China, 100029 China.

a) E-mail: yunxiaochun@cert.org.cn

DOI: 10.1587/transinf.E96.D.2354

traffic classification problem in a more efficient manner, and our contributions can be summarized as follows:

1. We propose an Improved Dynamic Time Warping approach named IDTW to overcome the above problems. IDTW aligns two sequences in order to obtain a dissimilarity measure using *non-linear* temporal alignment, and it can deal with the traffic classification problem of all the out-of-sequence situations.
2. We conduct extensive experiments on two real traces to evaluate the proposed method. Experimental results show that our approach can dramatically improve the accuracy of out-of-sequence traffic classification and enhance the robust of traffic classification.

The remainder of this paper is organized as follows: Sect. 2 introduces the related works. Section 3 summarizes different situations of out-of-sequence traffic, and analyzes the reason why they impact the existing machine learning classification methods. We introduce our IDTW approach in Sect. 4. Section 5 presents the experimental results and Sect. 6 concludes the paper.

2. Related Work

In this section, we first summarize the related works on traffic classification, and then introduce the out-of-sequence phenomenon in network traffic.

2.1 Related Works on Traffic Classification

In the last decades, there has been a lot of work for traffic classification using machine learning techniques. Such methods assume that the statistical characters of traffic are unique for different applications and can be used to distinguish them from each other. Then we can classify a traffic flow by different machine learning classification algorithms, such as Decision Trees, Bayesian Networks, Naive Bayes, K-Nearest Neighbor, Support Vector Machines and so on [14].

As mentioned in Sect. 1, the related work can divided into two classes: flow-level methods and packet-level methods. The flow-level methods were proposed in the early stage of traffic classification. Moore and Zuev [4] extracted 248 features based on the statistical properties of the whole traffic flow, then applied the Naive Bayes technique to categorize network traffic. The work was extended with the application of a Bayesian neural network approach [15]. Roughan et al. also introduced the Nearest Neighbor algorithm into traffic classification with flow-level features [16]. The flow-level techniques calculate features over the full flows, which may have thousands of packets. Hence, these techniques become less practical due to the time consuming feature extraction.

Recently, more and more researchers focused on packet-level techniques because it classify flows in a real-time manner. These methods extracted the features such as the packet size and the inter-packet time of the first *n* packets of a flow. Bernaille et al. [6], [17] first proposed a early traffic classification method. In this work, every flow is mapped to an *n*-dimensional space depending on features such as the packet size and the direction of its first *n* packets. Heuristics based on minimum Euclidean distance are used to assign class label to analyzed flows. Similar to approach proposed by Bernaille, Crotti et al. not only made use of packet size and direction as features, but also inter-arrival times, which provided a statistical behavioral description of the corresponding protocol [7]. Este et al. took advantage of the packet size of the first two packets and applied support vector machine algorithm for traffic classification [18]. There has been a number of works on packet-level traffic classification [19]–[23], all of which proved the effectiveness of packet-level traffic classification method.

Although the packet-level traffic classification method can obtain high accuracy, it can not deal with traffic flow with out-of-sequence packets [7], [18]. The reason is that all of the existing machine learning algorithms are based on the hypothesis of right order of packets series and using Euclidean distance to measure similarity among feature vectors. Thus they cannot classify out-of-sequence flows effectively, since Euclidean distance is very sensitive to distortion of feature vectors.

2.2 Related Works on Out-of-Sequence Traffic

Out-of-sequence is a very common phenomenon on network [9], especially in high speed networks where there is high degree of parallelism and different link speeds. During the last decade, a number of studies have measured the prevalence of out-of-sequence in the Internet. As previous proposed [8], packet reordering is prevalent at significantly high levels, and the probability of a session experiencing packet reordering is 90%. Paxson [9] reports that 12% and 36% of all connections, in two different data sets, included at least one reordering event. Jaiswal et al. [10] presented a measurement study for out-of-sequence packets in TCP connections within the Sprint IP backbone. They observed about 5% packets are out-of-sequence, and the percentage of flows experienced any out-of-sequence packets varied between 7.2% and 20.1%. Rewaskar et al. [12] found that the number of out-of-sequence deliveries in seven different traffic traces were between 17% and 51%.

Moreover, the use of UDP as a transport protocol has gained popularity recently [24], more and more P2P applications started to use UDP for their overlay signaling traffic. Zhou et al. analyzed the measurements of the out-of-sequence packets by tracing UDP packets, and pointed out that 56% of the whole flows suffered from out-of-sequence [13]. In view of this, it is highly desirable to design robust approaches to classify traffic with out-of-sequence packets.

Yang et al. [25] proposed a packet-level traffic classification approach and considered the impact of reordering packets in traffic. The classification method is SVM, which takes use of Euclidean distance to measure similarity between features. They simply considered packets with wrong order as loss, and set the values of the missing features to 0. In their simulate experiment, they randomly dropped several packets from the first five packets. The performance of their method decreased almost linearly by increasing loss packets. When there are more than two loss packets in a flow, the accuracy will be less than 45%. Their solution was relative simple since they just considered out-of-sequence packets as loss, while packets can arrive out-of-sequence for other reasons such as packet reordering and packet retransmission.

Nguyen et al. also measured the accuracy of traffic classification with packet loss [26]. They extract flow-level features of a sub-flow, and each sub-flow contains at least 25 packets. They used Naive Bayes and C4.5 decision tree to classify ET and VoIP traffic. They only considered packet loss phenomenon as well as Yang [25], and they chose 5% as packet loss rate. Their experimental results demonstrated that for ET traffic, the packet loss degraded Recall and Precision of both classifiers by less than 0.5%. For VoIP traffic, the packet loss did not produce noticeable degradation of the Naive Bayes classifiers Recall and Precision. However, it degraded the C4.5 Decision Tree classifiers Recall and Precision by 8.5% and 0.1%, respectively. Their measurement is not comprehensive since it only take into account of packet loss, moreover, they extract flow-level features which is time consuming.

In summary, none have been studied out-of-sequence traffic classification comprehensive so far. In this paper, we focus on construct a robust classifier in the presence of out-of-sequence packets.

3. Problem Statements

Out-of-sequence packets are very common in network, which mainly result from packet loss, looping, reordering, or duplication in the network. The well-recognized definition of out-of-sequence packets can be described as follows:

Definition 1: A packet is out-of-sequence if its sequence number is not equal to the expected sequence number.

A flow is typically defined by a 5-tuple, including *srcIP*, *dstIP*, *srcport*, *dstport* and *protocol*. IP packets that have the same 5-tuple are therefore considered to belong to the same flow. Consequently, the out-of-sequence flow can be described as follows:

Definition 2: If there is one or more out-of-sequence packets exist in the first *n* packets of a flow (in this paper, we set *n* to 5), this flow is out-of-sequence.

Jaiswal et al. [10] summarized that out-of-sequence packets can be caused by three different events: (1) *Retransmission*: In this case, a packet has been lost and would be retransmitted. (2)*Network duplication*: In this case, the retransmission of a packet not from the sender is observed. This may be because the monitoring point is within a routing loop, or the network creates a duplicate of the packet. (3) *Network-reordering*: In this case, the network changes the order of two packets of a connection, which may be because of parallelism within a router or a route change.

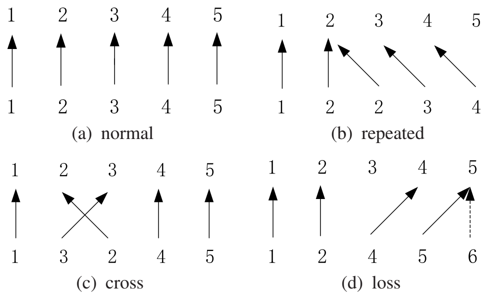


Fig. 1 Four mapping situations of normal, repeated, cross and loss.

Suppose we passively capture the network packets at a monitoring point between client and server, and record packet size and direction of the first 5 packets of a flow as feature. According to the above categories described by Jaiswal et al, we can summarize four mapping situations for traffic feature, as stated below:

1. ***normal***: this situation indicates that there is no out-of-sequence packet (Fig. 1 (a));
2. ***repeated***: this situation is caused by three reasons: (1) network duplication; (2) retransmission when a packet lost after monitoring point; (3) unnecessary retransmission when the ack packet lost (Fig. 1 (b));
3. ***cross***: this situation is also caused by two reasons: network reordering or retransmission when packet lost before monitoring point (Fig. 1 (c));
4. ***loss***: this situation means that a packet has been lost but no retransmission occurred (Fig. 1 (d));

In summary, the out-of-sequence situations include cases 2),3) and 4), and all of them will break the linear mapping property among features. It is worth noting that for the ***cross*** case, we hypothesise that the packet lag is 1. The packet lag refers to the number of packets, with a sequence number greater than the out-of-sequence packet, that are seen before the out-of-sequence packet itself. Wang et al. [27] studied internet packet reordering, and observed that 86.5% of reordering packets have a packet lag= 1, 95.3% of reordering packets have a packet lag≤ 2, and 78.8% of retransmitted packets have a packet lag≤ 3. Thus this hypothesis is relatively reasonable.

4. Improved DTW for Out-of-Sequence Traffic Classification

In this section, we first introduce the classic DTW algorithm [28], and then describe our imporved DTW method, namely IDTW for short. Finally, we introduce the construction of IDTW templates using K-Means.

4.1 Classic DTW

The DTW algorithm can handle the non-linear mapping problem, which has been proved to be superior to Euclidean distance for classification and clustering of time series [29].

DTW algorithm can be described as follows: Suppose there are two time series, a template S of length N and an input signal I of length M, where $S = s_1, s_2, s_3,\ldots,s_N$, and $I = i_1, i_2, i_3,\ldots,i_M$. To compare the similarity of these two time series using DTW, one can construct an N-by-M distance matrix D, and use $d(x,y)$ to represent the Euclidean distance between s_x and i_y, that is

$$d(x,y) = \left\| s_x - i_y \right\| \quad for \quad 1 \le x \le N; 1 \le y \le M \quad (1)$$

A warping path $W = w_1, w_2, w_3,\ldots,w_k,\ldots,w_K$, is a contiguous set of matrix element D that defines a mapping between the template S and input I. The k-th element of W is defined as $w_k = d(i_k, j_k)$ and $max(m,n) \le K \le m + n - 1$. The construction of the warping path W is subjected to the following constraints:

1. *Boundary constraint*: $w_1 = d(1,1)$ and $w_K = d(N,M)$, this requires the warping path W to start at (1,1) and end at (N,M).
2. *Continuity constraint*: Given $w_k = d(a,b)$, then $w_{k+1} = d(a',b')$, where $a' - a \le T$ and $b' - b \le T$. In practice, $T = 1, 2, 3$. If $T = 1$, the allowable steps in the warping path can only be between adjacent cells. If $T > 1$, the steps can be skipped.
3. *Monotonicity constraint*: Given $w_k = d(a,b)$, then $w_{k+1} = d(a',b')$, where $a' - a \ge 0$ and $b' - b \ge 0$. This restricts points in W to be monotonically spaced in time.

In DTW, $dtw(S,I)$ denotes the minimum warping cost of S and I. There are many warping paths that meet the above constraints. Since we are only interested in the path that preserves the minimum warping cost of S and I, we have:

$$dtw(S,I) = min\left(\sqrt{\sum_{k=1}^{K} w_k}\right) \quad (2)$$

To determine the minimum cost warping path, one can test every possible warping path between S and I. Such a procedure, however, will lead to a computational complexity that is exponential in the lengths N and M. So dynamic programming approach is introduced to address this problem. That is, the DTW algorithm constructs a matrix γ with dimension of N-by-M, the element of (x,y) in γ defines the cumulative distances of the warping path W from position $(1,1)$ to positive (x,y). The minimum of the cumulative distance is represent by $\gamma(x,y)$ as:

$$\gamma(x,y) = d(x,y) + min\{\gamma(x-1,y-1), \gamma(x-1,y), \gamma(x,y-1)\} \quad (3)$$

After creating the matrix γ, the value $\gamma(N,M)$ is the minimum cumulative distances of the DTW between the template S and the input I, that is, $dtw(S,I) = \gamma(N,M)$.

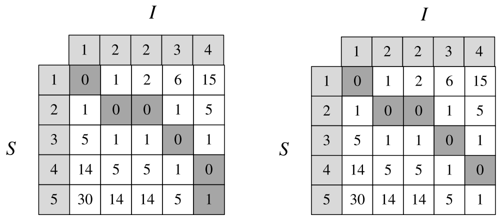
4.2 Improved DTW

In order to achieve better performance on out-of-sequence traffic classification, we design an improved DTW (IDTW) algorithm. Specifically, we relax the constraints of the classic DTW algorithm. Now let $idtw(S,I)$ denotes the distance in IDTW, we detail how to calculate $idtw(S,I)$ as below:

1. ***Relaxed boundary constraint***:
We relax the boundary constraint, that is, the last point of I is not necessary to be aligned to the last point of S. Therefore, $idtw(S,I)$ can be modified as: $idtw(S,I) = min\{\gamma(1,M), \gamma(2,M), \ldots, \gamma(N,M)\}$.
2. ***Relaxed monotonicity and continuity constraints***:
We relax the monotonicity and continuity constraints in our scheme by adding admissible step patterns. Equation (3) illustrates the admissible step patterns are $\{\gamma(x-1,y-1), \gamma(x-1,y), \gamma(x,y-1)\}$. Specifically, we add $\gamma(x-1,y+1)$ and $\gamma(x-1,y-2)$ to the set of step patterns of classic DTW to relax monotonicity and continuity constraints respectively, that is: $\gamma(x,y) = d(x,y) + min\{\gamma(x-1,y-1), \gamma(x-1,y), \gamma(x,y-1), \gamma(x-1,y-2), \gamma(x-1,y+1)\}$. Figure 3(a) and Fig. 3(b) visualize the step patterns of classic DTW and IDTW.

The objective of relaxing the boundary constraint is to address the ***repeat*** situation. As can be seen from Fig. 1, the training vector is 1,2,3,4,5 and the test vector is 1,2,2,3,4. Figure 2(a) and Fig. 2 (b) illustrate the matrix γ and warping path W of classic DTW and IDTW. The matrix is built column by column, from left to right and from top down for each column. We can find after relaxing the boundary constraint, $idtw(S,I) = \gamma(4,5) = 0$, which is the right distance between the two feature vectors.

The target of relaxing monotonicity and continuity constraint is to handle the ***cross*** and ***loss*** situation. For the cross situation, the latter packet may appear before the former one. Hence, we can conclude that the warping path is not monotonically and continuously increasing any more. Now take a look at Fig. 1 again, the training vector and test vector are 1,2,3,4,5 and 1,3,2,4,5 respectively. Figure 3(c) and Fig. 3(d) depict the warping path of classic DTW and IDTW for cross situation. We can observe $idtw(S,I) = 0$, and the warping path reflects the true mapping relationship of training and test feature vectors. We can handle loss situation by relaxing continuous constraints as well. Assume the training vector and test vector are 1,2,3,4,5 and 1,3,4,5,6. We observe that the second element of test vector should mapping to the third element of training vector but not the second,



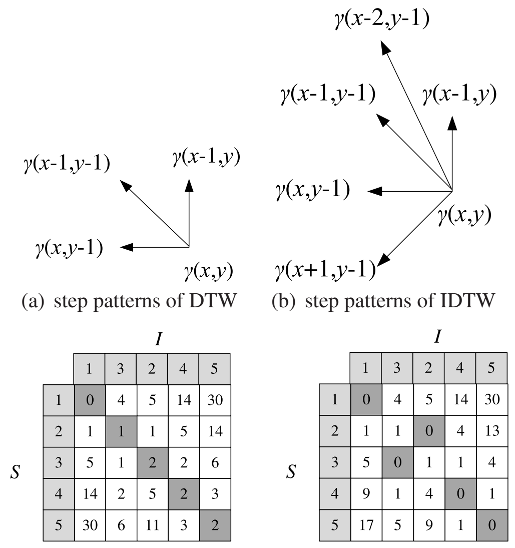
	1	2	2	3	4
1	0	1	2	6	15
2	1	0	0	1	5
3	5	1	1	0	1
4	14	5	5	1	0
5	30	14	14	5	1

(a) before relax boundary constraint

	1	2	2	3	4
1	0	1	2	6	15
2	1	0	0	1	5
3	5	1	1	0	1
4	14	5	5	1	0
5	30	14	14	5	1

(b) after relax boundary constraint

Fig. 2 Boundary relaxed DTW.



(a) step patterns of DTW (b) step patterns of IDTW

	1	3	2	4	5
1	0	4	5	14	30
2	1	1	1	5	14
3	5	1	2	2	6
4	14	2	5	2	3
5	30	6	11	3	2

	1	3	2	4	5
1	0	4	5	14	30
2	1	1	0	4	13
3	5	0	1	1	4
4	9	1	4	0	1
5	17	5	9	1	0

(c) before relax monotonicity and continuity constraint (d) after relax monotonicity and continuity constraint

Fig. 3 Monotonicity relaxed DTW.

which implies that there exists skip pattern in warping path.

4.3 Construction of DTW Template

An application may have different kinds of behaviors. Since IDTW is based on template matching, we adopt the K-Means algorithm to group similar flows into clusters and use the cluster centers to construct templates of IDTW in our scheme.

Let $T = t_1, t_2, \ldots, t_n$ be the training set. Each flow $t_i \in T$ is described by a feature vector $P = p_1, p_2, \ldots, p_m$. Let p_j be the payload size of packet j in a flow. We use the direction of the packet to determine whether p_j is positive or negative. When a packet j is sent by TCP client, then p_j is a positive value. Conversely, if j is sent by the TCP server, then p_j is negative. As for traffic classification, let us assume a set of M network applications $L = l_1, l_2, \ldots, l_M$, where each l_i corresponds to the label of each application, such as http, pop3, etc. We then cluster the training set using the K-Means algorithm. Suppose there are K clusters: $C = c_1, c_2, \ldots, c_K$, let $O = o_1, o_2, \ldots, o_K$ represent the corresponding centers. For each cluster we maintain a tuple $\{c_k, l_k\}$, where c_k denotes center and l_k denotes label. Let $X = x_1, x_2, \ldots, x_J$ denote the test set of out-of-sequence flows, our objective is to compute the similarity distance between x_j and all sets of the templates, then assign the label $\widehat{l_j}$ having the minimum similarity distance to x_j.

The existing classification methods based on Euclidean distance will assign the closet cluster label to it, that is:

$$\widehat{l_j} = label\left(\arg\min_k d\left(x_j, o_k\right)\right) \tag{4}$$

Where d is the standard Euclidean distance which is sensitive to noise and misalignments, thus may impact the performance of existing classification methods.

To overcome the drawbacks of Euclidean distance metric, we use our IDTW distance *idtw* described above, rather than the Euclidean distance, to compute the label, as stated below:

$$\widehat{l_j} = label\left(\arg\min_k idtw\left(x_j, o_k\right)\right) \tag{5}$$

5. Performance Evaluation

This section presents the performance evaluation of our IDTW algoirthm versus 5 machine learning approaches and classic DTW on classifying two real sets of traffic trances, where one is UNIBS traffic traces [30], and the other one is LBNL traffic traces [31].

5.1 Dataset Description

In our experiments, we use two datasets to test classification methods. Both of the two traces consist of normal flows and out-of-sequence flows which contain all of the out-of-sequence situations listed in Sect. 3. We select all the normal flows as training set, and all the out-of-sequence flows as test set. To avoid a biased analysis due to unbalanced representation of classes, we randomly sample 2000 flows for each of the application (If an application consist of less than 2000 flows, choose all of its flows). We adopt K-Means method to cluster the training data. To tradeoff clustering quality and the scale of the clusters, we empirically set the number of clusters to be 400.

UNIBS:This trace was collected at the edge router of the campus network of University of Brescia on three consecutive working days in 2009. The traffic are categorized into the following classes: Web (http and https), Mail (pop3, pop3s), VoIP (skype) and P2P (bittorrent, edonkey).

Table 1 shows the detail composition of the UNIBS data set. There are 2777 out-of-sequence flows, accounting for 5.4% of all the flows. The overall flow number of the repeat, cross and loss situations are 2049, 529,199, and correspondingly, their ratios are 73.8%, 19.0%, 7.2% respectively. The repeat situation is the majority, followed by cross and loss situation. A closer look at Table 1 reveals that the out-of-sequence flows are quite common in bittorrent and edonkey applications, which is conform to reality.

LBNL:The LBNL traffic traces were collected at the Lawrence Berkeley National Laboratory under the Enterprise Tracing Project. The traffic traces are completely anonymized, so all the packets do not have payload. Therefore, we label each flow according to its TCP destination port number. The composition of LBNL data set is reported in Table 2. There are 5922 out-of-sequence flows, accounting for 14.2% of all the flows. The flow number of repeat, cross and loss situations are 4454, 1143, 325, and their ratios are 75.2%, 19.3%, 5.5%. We observe the ratios are similar to the UNIBS dataset, which reflects the repeat situation is the most common situation as well.

Table 1 Composition of traffic of UNIBS data set.

Application	Normal flows	Out-of-sequence flows	repeat flows	cross flows	loss flows
pop3s	3504	22	15	6	1
http	26425	554	158	269	127
skype	857	6	5	1	0
bittorrent	3276	950	837	76	37
pop3	843	31	19	9	3
https	995	10	2	8	0
edonkey	12962	1204	1013	160	31

Table 2 Composition of traffic of LBNL data set.

Port	Expected application	Normal flows	Out-of-sequence flows	repeat flows	cross flows	loss flows
80	http	22664	4425	3355	827	243
139	netbios	3690	326	260	45	21
25	smtp	4998	164	123	30	11
443	https	6424	825	551	227	47
993	imaps	2771	43	37	5	1
110	pop3	341	91	81	8	2
22	ssh	187	16	15	1	0
995	pop3s	523	32	32	0	0

5.2 Experimental Results

We compare our IDTW approach with five standard machine learning approaches on the weka platform [32]: Decision Trees (J48), Bayesian Networks (BN), Naive Bayes (NB), K-Nearest Neighbor (KNN, here we use 1-NN), Support Vector Machines (SVM), and classic DTW algorithm. All of the five machine learning approaches are based on the linear alignments between two feature vectors. The IDTW and DTW can classify data by nonlinear mapping.

Since we choose flows for each application as training set randomly, the classification result may be unstable. To avoid the injustice, we select training samples and repeat each experiment for 10 times, and then report the average result on all runs.

We measure the performance of a given algorithm in terms of the following three metrics:

- *Overall accuracy* - the ratio of all flows correctly classified. This metric is used to measure the accuracy of a clasifier on the whole dataset.
- *Recall* - the ratio of flows from a given class that are properly attributed to that class. Recall is used to evaluate the per-class performance.
- *Precision* - the ratio of flows correctly attributed to a class over the total flows attributed to that class. Precision is used to evaluate the per-class performance as well.

5.2.1 Overall Performance

Table 3 Overall accuracy of UNIBS dataset.

J48	BN	NB	KNN	SVM	DTW	IDTW
61.0%	61.7%	55.2%	51.3%	64.3%	68.5%	**92.3%**

Table 4 Overall accuracy of LBNL dataset.

J48	BN	NB	KNN	SVM	DTW	IDTW
70.8%	72.7%	54.4%	73.5%	68.5%	78.5%	**93.4%**

The overall performance is evaluated according to overall accuracy. Table 3 and Table 4 show the overall accuracy on the two datasets obtained by IDTW, DTW and other five methods.

First, a general observation is that IDTW outperforms all the other methods in term of overall accuracy. From Table 3 it can be seen that IDTW is higher than the rest methods by approximately 24% to 41% percent for UNIBS dataset, and Table 4 demonstrates that IDTW is higher than other methods by 15% to 40% for LBNL dataset. All the traditional machine learning methods degrade sharply. Now let's analysis the results of J48 and SVM for instance. As we know, the out-of-sequence packets will change the time serial characteristics of feature vectors, thus will affect the representation of them. However, the J48 method can't change the paths within the tree adaptively to handle this problem. For SVM classification method, the feature vectors of out-of-sequence flows may be mapped to the wrong space, which will lead to wrong classification result. Our proposed IDTW method shows better performance, the reason is that: IDTW can adapt to all the out-of-sequence situations by relaxing constraints of classic DTW, thus achieve high accuracy.

Second, we find that the result of the classic DTW algorithm is slightly better than the other five machine learning algorithms, but the result is still not quite acceptable. The reason is that although the classic DTW algorithm can handle non-linear mapping problem, it is not quite adapt to the various out-of-sequence situations.

5.2.2 Per-Class Performance

We use recall and precision to measure the per-class performance of all the methods on the two datasets.

Figure 4 illustrates the recall of each method on the UNIBS dataset, which shows that none of the existing machine learning classification methods and DTW can keep high accuracy for all of the applications. For example, NB obtains better performance on http flow classification in comparison to other methods, however, it can not classify https, pop3 and skype at all. SVM and J48 have the same problem with NB. KNN, BN and DTW can classify all of the applications more or less, but they could not always achieve the best performances. In contrast, IDTW almost achieves better recall than other methods for all of the applications.

We also observe an interesting phenomenon, that is, all the approaches can classify http with relatively high accuracy. Further investigation shows that most of http flows are relative persistent to out-of-sequence packets. For example, a sequence of the first 5 packets of a normal flow is 504, −1460, −1460,−1460,−1460. Assume the third packet re-

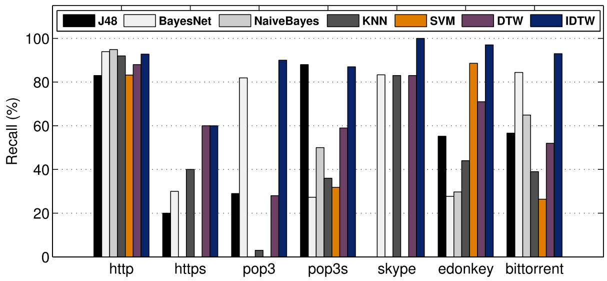


Fig. 4 Recall of UNIBS dataset.

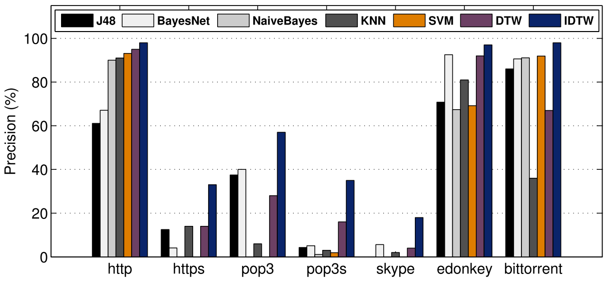


Fig. 5 Precision of UNIBS dataset.

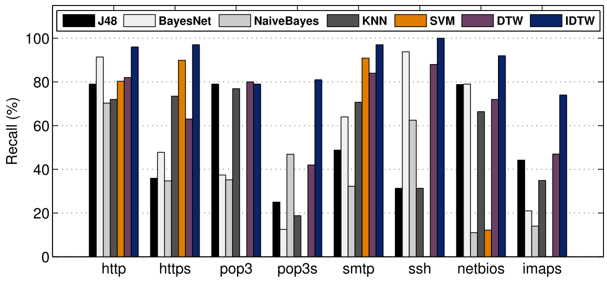


Fig. 6 Recall of LBNL dataset.

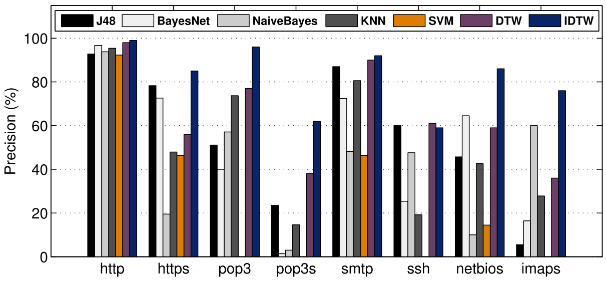


Fig. 7 Precision of LBNL dataset.

transmits or reorder, then the out-of-sequence flow is still 504, −1460, −1460,−1460,−1460, which does not change the mapping relationship, thus the accuracies of existing classification methods still remain at a high level. Pop3, pop3s, skype, edonkey and bittorrent are not quite easy to classify for existing machine learning methods. For these applications, IDTW can greatly improve the classification results. It contrast, it is hard to classify https for all the methods including IDTW, since https flows have similar patterns of http flows.

Figure 5 shows the precision of each classifier on the UNIBS dataset. We find that the precisions of classifying https, pop3, pop3s and skype are lower than that of other applications. We think the possible reason is the test set of these applications contain less out-of-sequence flows(Table 1 shows that the test set only contains 10 https flow, 31 pop3 flows, 22 pop3s flows and 6 skype flows in the test set). Therefore, even if only a small amount of other flows are misclassified as them, precision will decline greatly. Nevertheless, IDTW achieves best precision for all applications. Although some classifiers obtain high precision with an application, but the recall rate is low, thus precision does not make sense. For example, BayesNet classifier achieves good precision of edonkey, but a lot of edonkey flows are classified as http, which reduces the recall of edonkey and precision of http.

Figure 6 and Fig. 7 depict the recall and precision on LBNL dataset respectively. Similar to the situation of UNIBS dataset, the recall and precision of http are satisfactory for all the existing machine learning methods, and thus the improvement space for IDTW is small. For other applications such as https, pop3, pop3s, smtp, ssh, netbios and imaps, the recall and precision of IDTW are consistently higher than other approaches. In sum, the results show IDTW has a superior performance comparing with other existing machine learning methods.

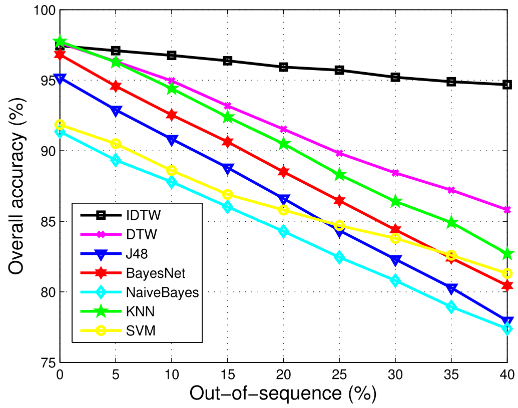


Fig. 8 Overall accuracy of different out-of-sequence ratio of UNIBS dataset.

5.2.3 Classification of Both Normal and Out-of-Sequence Flows

To further validate the effectiveness of IDTW, we classify the data contains both normal flows and out-of-sequence flows. We evaluate on UNIBS dataset, choose 1/2 of the normal flows as the training set, and 5000 normal flows plus different number of out-of-sequence flows as the test set. The ratio of the out-of-sequence flows increases from 0% to 40%. Figure 8 depicts the overall accuracies of different approaches versus IDTW approach on UNIBS dataset.

As shown in Fig. 8, with the growth of the ratio of the out-of-sequence flows, the accuracies of existing machine learning classifier decline sharply, which decrease about 15% on average. On the contrary, the accuracy of IDTW declines quite slowly (from 97.44% to 94.68%), which is no

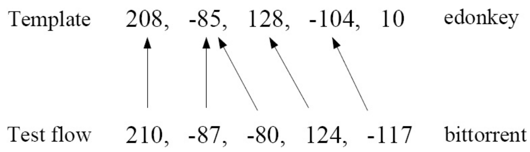


Fig. 9 A special case of classification of normal flow by IDTW.

more than 3%. We can also observe that the slope of IDTW curve is flattest, which means that IDTW is the most robust out-of-sequence traffic classification approach.

It is noteworthy that when there are no out-of-sequence flows, the accuracy of IDTW is small than that of DTW and KNN. As shown in Fig. 8, the accuracies of IDTW, DTW and KNN are 97.44%, 97.56% and 97.64% respectively. We observe that the accuracy of IDTW is 0.12% and 0.2% lower than that of DTW and KNN. We analyze the classification procedure of IDTW in depth to study this phenomenon. Suppose there is a normal flow x which belonging to application l_1, and the most similar training template is o_1. Suppose by computing the Euclidean distance based on linear mapping, KNN finds the most similar template is o_1 and obtains the right label l_1. While IDTW calculates *idtw* distance which is based on nonlinear mapping and relaxation of classic DTW. Thus it may consider the most similar training template is o_2 and assigned flow x a label of l_2. Therefore IDTW fails to get the right classification result. For instance, suppose there is a normal bittorrent flow (210, −87, −80, 124, −117). KNN selects the corresponding template is (177 , −48, −135, 123, −103), and the label of this template is “bittorrent”. While IDTW finds the most similar template is (208, −85, 128, −104, 10), and the label of this temple is “edonkey”. Therefore IDTW classifies a bittorrent flow as edonkey mistakenly. This special case is demonstrated in Fig. 9.

We list all misclassification cases of normal flows on UNIBS dataset in Table 5. The first two columns declare the test flow and the corresponding label. The third column states the classification methods: IDTW, KNN, DTW. The forth columns lists the template flow chosen by these methods. For IDTW, we add a indicator to state the specific out-of-sequence situation which is mistaken for by IDTW. For instance, (***r***) means IDTW considers the normal test flow as repeated flow, (***c***) and (***l***) represent the cross and loss situation respectively. The last column represents the corresponding predicted label obtained by the above methods. If the predicted label is different from the original label, we mark it with italic type, which means the result is wrong. From Table 5, we find that due to the non-linear mapping properties and relaxation constraints of classic DTW, IDTW may considers normal flows as out-of-sequence flows. The DTW method may meet the similar situations as IDTW because of its non-linear mapping characteristics. While it should subject to boundary constraint, continuity constraint and monotonicity constraint. Thus it won’t classify normal flows as out-of-sequence flows as frequently as IDTW.

Table 5 Misclassification cases on UNIBS dataset.

Test flow	Label	Method	Template flow	Predicted label
210, −87, −80, 124, −117	bittorrent	IDTW	208, −85, 128, −104, 10 (***r***)	*edonkey*
		KNN	177, −48, −135, 123, −103	bittorrent
		DTW	208, −78, 67, 62, −108	*edonkey*
144, −126, 6, 41, 148	https	IDTW	68, −131, −357, −5, 1448 (***l***)	*bittorrent*
		KNN	155, −48, −135, 123, −103	https
		DTW	155, −48, −135, 123, −103	https
88, 94, −94, −78, 109	bittorrent	IDTW	93, −92, 128, −91, 98 (***c***)	*edonkey*
		KNN	123, −38, 17, −117, 106	bittorrent
		DTW	129, −129, −83, 111, 131	*edonkey*
281, −224, −78, −78, −78	pop3	IDTW	329, −211, 910, −83, −83 (***l***)	*http*
		KNN	296, −328, −78, −78, −65	pop3
		DTW	296, −328, −78, −78, −65	pop3
155, −122, 6, 37, 506	skype	IDTW	76, −31, −436, 20, 460 (***l***)	*bittorrent*
		KNN	149, −132, 5, 45, 638	skype
		DTW	149, −132, 5, 45, 638	skype
260, −83, −121, 122, −99	bittorrent	IDTW	252, −83, 127, −108, 70 (***c***)	*edonkey*
		KNN	177, −48, −135, 123, −103	bittorrent
		DTW	177, −48, −135, 123, −103	bittorrent
68, −97, −724, −7, 17	bittorrent	IDTW	102, −67, −1448, −669, 139 (***l***)	*pop3s*
		KNN	74, −41, −599, 54, 104	bittorrent
		DTW	74, −41, −599, 54, 104	bittorrent
160, −122, 6, 37, 1460	https	IDTW	68, −131, −357, −5, 1448 (***l***)	*bittorrent*
		KNN	155, −128, 48, 82, 1292	https
		DTW	155, −128, 48, 82, 1292	https
272, −98, −73, 124, −97	bittorrent	IDTW	254, −85, 129, −103, 49 (***r***)	*edonkey*
		KNN	177, −48, −135, 123, −103	bittorrent
		DTW	235, −100, 126, −107, −15	*edonkey*
151,−106,−237, 124, −101	bittorrent	IDTW	129, −121, 44, −215, −122 (***c***)	*edonkey*
		KNN	283, −75, −130, 123, −104	bittorrent
		DTW	283, −75, −130, 123, −104	bittorrent

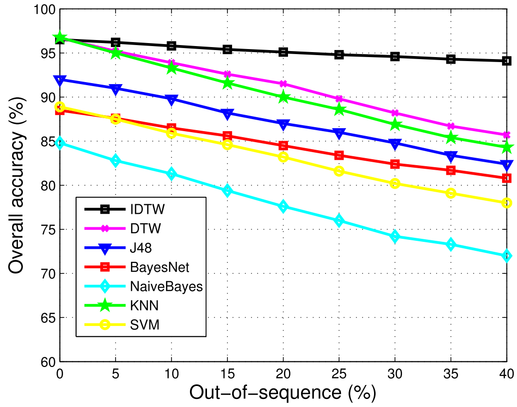


Fig. 10 Overall accuracy of different out-of-sequence ratio of LBNL dataset.

To further confirm the effectiveness of IDTW, we carry out the the same experiment on LBNL dataset. The experimental result is is depicted in Fig. 10. From Fig. 10, we can also find that with the increase of out-of-sequence ratio, the accuracies of existing machine learning methods degrade significantly, while the accuracy of IDTW declines slowly (from 96.52% to 94.11%). When the out-of-sequence ratio is 0%, the accuracies of IDTW, DTW and KNN methods are 96.52%, 96.7% and 96.76% respectively. We find that accuracy of IDTW is 0.18% and 0.24% lower than DTW and KNN. The reason of this inversion is as same as that of UNIBS dataset. We list the misclassification cases of normal flows on LBNL DATASET in Table 6.

Table 6 Misclassification cases on LBNL dataset.

Test flow	Label	Method	Template flow	Predicted label
186, −948, −948, −948, −140	http	IDTW	142, −968, −968, −151, 204(***r***)	*imaps*
		KNN	264, −998, −998, −998, −272	http
		DTW	264, −998, −998, −998, −272	http
255, 605, −17, −1181, −468	http	IDTW	125, −80, 782, −1351, −514(***c***)	*netbios*
		KNN	46, 514, −269, −1368, −421	http
		DTW	46, 514, −269, −1368, −421	http
850, −708, 689, −1460, −1460	http	IDTW	880, −684, 688, −1380, −724(***r***)	*pop3s*
		KNN	825, −910, 842, −1437, −1420	http
		DTW	825, −910, 842, −1437, −1420	http
176, −128, 48, −512, −512	imaps	IDTW	110, −114, 40, 834, −437(***l***)	*https*
		KNN	240, −512, −170, −512, −512	imaps
		DTW	94, −146, 27, 325, −590	*https*
4, −4, 20, −26, 16	netbios	IDTW	−15, 33, −21, 12, −21(***r***)	*telnet*
		KNN	25, −4, 25, −10, 27	netbios
		DTW	25, −4, 25, −10, 27	netbios
−120, −138, 624, −612, −527	smtp	IDTW	−120, −138, 1445, 547, −595(***l***)	*pop3*
		KNN	−54, −61, 639, −410, −412	smtp
		DTW	−98, −139, 25, 518, −628	*pop3*
1073, 184, −112, −1460, −1348	http	IDTW	1138, −201, 118, −1372, −10(***c***)	*netbios*
		KNN	1040, −43, −160, −1428, −1348	http
		DTW	971, 352, −61, −181, −1393	http
137, −177, 210, −452, 370	pop3	IDTW	102, −146, 290, 343, −337(***c***)	*https*
		KNN	130, −141, 310, −297, 371	pop3
		DTW	130, −141, 310, −297, 371	pop3
111, −882, 139, 47, −47	pop3s	IDTW	104, −63, −903, 208, −6(***l***)	*https*
		KNN	109, −1014, 119, 52, −50	pop3s
		DTW	109, −1014, 119, 52, −50	pop3s
120, −146, 735, −517, −1380	https	IDTW	125, −80, 782, −1351, −514(***c***)	*netbios*
		KNN	119, −128, 730, −350, −1360	https
		DTW	119, −128, 730, −350, −1360	https
958, −4, −8, −1368, −173	netbios	IDTW	971, 353, −61, −181, −1393(***c***)	*http*
		KNN	1138, −201, 118, −1372, −10	netbios
		DTW	994, 115, −1426, −1440, −131	*https*
408, −239, 667, −1460, −1460	https	IDTW	516, −300, 552, −1448, −256(***r***)	*http*
		KNN	395, −173, 466, −1413, −1421	https
		DTW	395, −173, 466, −1413, −1421	https

By summarizing the above experimental results, we find that the accuracy of IDTW is slightly lower than other methods such as KNN and DTW when there are no out-of-sequence flows. The difference is quite small since the special situations which lead to this phenomenon are not very common. While in the cases when there exist out-of-sequence flows, especially the ratio of out-of-sequence flows should not be ignored, IDTW is significantly better than other approaches. The whole analysis described here confirms that the IDTW approach proposed in this paper is very effective.

It is worth noting that, although the IDTW approach proposed in this paper is based on the analysis of TCP traffic, it can be applied to UDP traffic classification as well. Generally speaking, UDP protocol is an unreliable protocol, thus the out-of-sequence situations will happen quite often. Therefore, IDTW method is an effective solution for robust UDP traffic classification as well.

5.3 Discussion

In this section, we provide some discussions on computational complexity of our algorithm and the influence of number of packets to classification accuracy.

5.3.1 Computational Complexity of Proposed Algorithm

Suppose Running our classification method requires extracting feature of test flow x, and computing the similarity distance between x and training flow t. Since our method only extracts the packet size of the first N packets of a flow, the computational complexity for feature extracting is $O(N)$. For an input feature size of N and template size of N, the complexity of classic DTW algorithm is $O(N{\times}N)$ [29], [33], and it is not difficult to see that the computational complexity of our IDTW is the same as the classic DTW algorithm. The classical machine learning method based on Euclidean distance have a cost of linear complexity, which is $O(N)$. We can conclude the total computational complexity of IDTW algorithm is $O(N{+}N{\times}N)$, and the classical machine learning method based on Euclidean distance is $O(N + N)$.

Although the cost of our algorithm is higher than classical machine learning method based on Euclidean distance, our algorithm can still run fast enough since it is based on packet-level classification method. Now let's analysis the computational complexity of flow-level classification method. Computing flow-level features such as mean packet size often may require memory complexity $O(L)$ (where L is the total number of packets of a flow). To collect K features of the whole flow, the computational complexity of feature extraction is $O(K \times L)$. Suppose we still use classical machine learning method based to classify a flow, the whole cost is $O(K \times L + K)$. For the most situations $N \ll L$, $N \approx K$. In this paper we only use the first 5 packets, which is a very small value. Compared with flow-level classification method, our method is still superior on computational complexity.

The IDTW algorithm presented in this paper is fast and effective, thus it suitable to implement in customized hardware. Such devices only need to collect the first n number of a flow, and compute the *idtw* distance with several templates. It is hoped to straightforwardly implemented in high-speed routers.

5.3.2 Influence of Number of Packets to Classification Accuracy

In this subsection, we will make discussion about the effects of the number of packets to classification accuracy. We calculate the accuracies of the above classification methods as the number of packets increases from 1 to 10.

Figure 11 presents the classification accuracies with respect to the changes of number of packets in UNIBS dataset. This figure shows the IDTW method outperforms all the existing machine learning approaches. As we see, all existing machine learning classifiers present relative low and unstable accuracies, which are between 60% to 75%. This means that they can not classify out-of-sequence flows satisfactorily no matter how many packets are used. Figure 11 shows that the accuracy of IDTW improves obviously as the number of packets increases from 1 to 5, while the accuracy becomes stable when more than 5 data packets are considered. From Fig. 11, we can find that using the first 5 packets can achieve sufficient high accuracy for the IDTW method.

To further study the influence of number of packets,

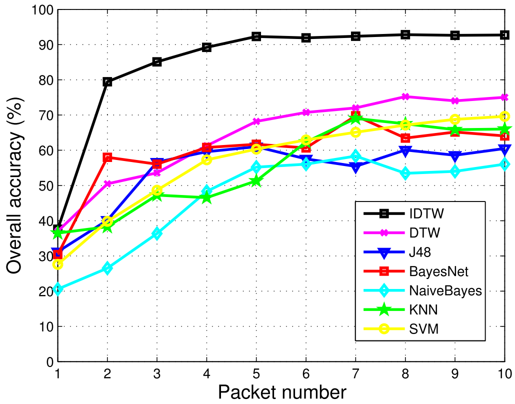


Fig. 11 Influence of the number of packets on classification accuracy of UNIBS dataset.

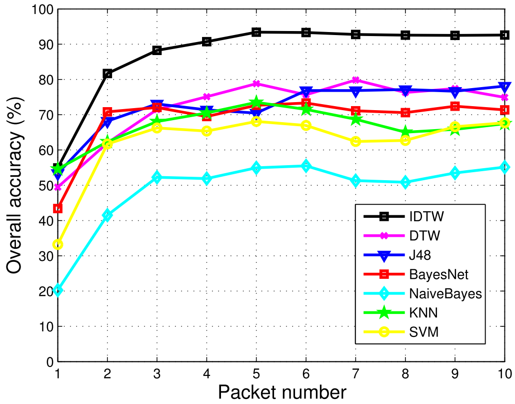


Fig. 12 Influence of the number of packets on classification accuracy of LBNL dataset.

we conduct the same experiment on LBNL dataset, and the result is demonstrated in Fig. 12. From this figure we obtain similar conclusion with Fig. 11: Using the first 5 packets gives satisfactory performance for IDTW algorithm, while adding more packets doesn't improve accuracy significantly.

As described in Sect. 5.3.1, the number of packets determines the computational complexity of IDTW algorithm. Therefore, we choose the number of packets which obtains the best trade-off between accuracy and computational complexity. Based on the above experimental results, we only use the first 5 data packets of a flow for classification in our experiment.

6. Conclusions

In this paper, we addressed the traffic classification problem with out-of-sequence packets and proposed an improved DTW algorithm. To our best knowledge, this is the first comprehensive work in online traffic classification with out-of-sequence packets. The result on two real traces showed that IDTW performs better than other approaches with comparable quality. Specifically, IDTW obtained an overall accuracy of 92.3% on UNIBS dataset and 93.4% on LBNL dataset. Compared with existing machine learning methods, the IDTW approach is more effective and robust to classify traffic with out-of-sequence packets. We attribute this advantage to the fact that the IDTW can find the nonlinear mapping relationship between traffic packet series and adapt to various out-of-sequence situations.

There are numerous research approaches that can be used to improve the speed of DTW, by using lower bounding techniques and global constraint techniques [34]. In our future work, we intend to study how to speed up IDTW calculation by utilizing these techniques.

Acknowledgments

This work is supported by National High Technology Research and Development Program of China (Grant No.2012AA013001), and National Science and Technology Support Program of China (Grant No.2012BAH37B02, 2012BAH46B02), and National Information Security Program of China(Grant No.2012A54, 2012A51, 2012A62).

References

[1] T. Karagiannis, A. Broido, N. Brownlee, K.C. Claffy, and M. Faloutsos, "Is P2P dying or just hiding?," IEEE GLOBECOM 2004, vol.3, pp.1532–1538, 2004.
[2] S. Sen, O. Spatscheck, and D. Wang, "Accurate, scalable in-network identification of p2p traffic using application signatures," WWW 2004, pp.512–521, 2004.
[3] T.T.T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," IEEE Communications Surveys & Tutorials, vol.10, no.4, pp.56–76, 2008.
[4] A.W. Moore and D. Zuev, "Internet traffic classification using bayesian analysis techniques," ACM SIGMETRICS Performance Evaluation Review, vol.33, pp.50–60, 2005.
[5] J. Erman, A. Mahanti, and M. Arlitt, "Internet traffic identification using machine learning," IEEE GLOBECOM 2006, pp.1–6, 2006.
[6] L. Bernaille, R. Teixeira, and K. Salamatian, "Early application identification," CoNEXT 2006, p.6, 2006.
[7] M. Crotti, M. Dusi, F. Gringoli, and L. Salgarelli, "Traffic classification through simple statistical fingerprinting," ACM SIGCOMM Computer Communication Review, vol.37, no.1, pp.5–16, 2007.
[8] J.C.R. Bennett, C. Partridge, and N. Shectman, "Packet reordering is not pathological network behavior," IEEE/ACM Trans. Netw., vol.7, no.6, pp.789–798, 1999.
[9] V. Paxson, "End-to-end internet packet dynamics," ACM SIGCOMM Computer Communication Review, vol.27, pp.139–152, 1997.
[10] S. Jaiswal, G. Iannaccone, C. Diot, J. Kurose, and D. Towsley, "Measurement and classification of out-of-sequence packets in a TIER-1 IP backbone," IEEE INFOCOM 2003, vol.2, pp.1199–1209, 2003.
[11] L. Gharai, C. Perkins, and T. Lehman, "Packet reordering, high speed networks and transport protocol performance," ICCCN 2004, pp.73–78, 2004.
[12] S. Rewaskar, J. Kaur, and F.D. Smith, "A passive state-machine approach for accurate analysis of tcp out-of-sequence segments," ACM SIGCOMM Computer Communication Review, vol.36, no.3, pp.51–64, 2006.
[13] X. Zhou and P. Van Mieghem, "Reordering of IP packets in Internet," PAM 2004, pp.237–246, 2004.
[14] H. Jiawei and M. Kamber, Data mining: Concepts and techniques, Morgan Kaufmann, San Francisco, CA, 2001.
[15] T. Auld, A.W. Moore, and S.F. Gull, "Bayesian neural networks for internet traffic classification," IEEE Trans., Neural Netw., vol.18, no.1, pp.223–239, 2007.

[16] M. Roughan, S. Sen, O. Spatscheck, and N. Duffield, "Class-of-service mapping for QoS: A statistical signature-based approach to ip traffic classification," Proc. 4th ACM SIGCOMM conference on Internet measurement, pp.135–148, 2004.

[17] L. Bernaille and R. Teixeira, "Early recognition of encrypted applications," Passive and Active Network Measurement, pp.165–175, 2007.

[18] A. Este, F. Gringoli, and L. Salgarelli, "Support vector machines for tcp traffic classification," Comput. Netw., vol.53, no.14, pp.2476–2490, 2009.

[19] A. Dainotti, F. Gargiulo, L.I. Kuncheva, A. Pescape, and C. Sansone, "Identification of traffic flows hiding behind TCP port 80," 2010 IEEE International Conference on Communications (ICC), pp.1–6, 2010.

[20] C. Beşiktaş and H. Mantar, "Real-time traffic classification based on cosine similarity using sub-application vectors," Traffic Monitoring and Analysis, pp.89–92, 2012.

[21] D. Nechay, Y. Pointurier, and M. Coates, "Controlling false alarm/discovery rates in online internet traffic flow classification," IEEE INFOCOM 2009, pp.684–692, 2009.

[22] C. Barakat and M. Jaber, "Enhancing application identification by means of sequential testing," IFIP Lecture Notes in Computer Science (LNCS), 5550, pp.287–300, 2011.

[23] V. Carela-Español, P. Barlet-Ros, M. Solé-Simó, A. Dainotti, W. de Donato, and A. Pescapé, "K-dimensional trees for continuous traffic classification," Traffic Monitoring and Analysis, pp.141–154, 2010.

[24] M. Zhang, M. Dusi, W. John, and C. Chen, "Analysis of udp traffic usage on internet backbone links," IEEE SAINT 2009, pp.280–281, 2009.

[25] B. Yang, G. Hou, L. Ruan, Y. Xue, and J. Li, "Smiler: Towards practical online traffic classification," Proc. 2011 ACM/IEEE Seventh Symposium on Architectures for Networking and Communications Systems, pp.178–188, 2011.

[26] T.T.T. Nguyen, G. Armitage, P. Branch, and S. Zander, "Timely and continuous machine-learning-based classification for interactive IP traffic," IEEE Trans. Netw., vol.10, no.4, pp.56–76, 2012.

[27] Y. Wang, G. Lu, and X. Li, "A study of internet packet reordering," Information Networking. Networking Technologies for Broadband and Mobile Networks, pp.350–359, 2004.

[28] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," IEEE Trans. Acoust., Speech Signal Process., vol.26, no.1, pp.43–49, 1978.

[29] C.A. Ratanamahatana and E. Keogh, "Three myths about dynamic time warping data mining," Proc. SIAM International Conference on Data Mining (SDM05), pp.506–510, 2005.

[30] The unibs internet traces. http://www.ing.unibs.it/ntw/tools/traces

[31] Lbnl/icsi enterprise tracing project.
http://www.icir.org/enterprisetracing

[32] Weka. http://www.cs.waikato.ac.nz/ml/weka/

[33] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh, "Querying and mining of time series data: experimental comparison of representations and distance measures," Proc. VLDB Endowment, vol.1, no.2, pp.1542–1552, 2008.

[34] E. Keogh and C.A. Ratanamahatana, "Exact indexing of dynamic time warping," Knowledge and Information Systems, vol.7, no.3, pp.358–386, 2005.

Jinghua Yan was born in 1986. She received the B.S. and M.S. degrees in communication engineering from Communication University of China, Beijing, China in 2007 and 2009. She is pursuing the Ph.D. degree in Beijing University of Posts and Telecommunications. Her research interests include traffic classification, data mining.

Xiaochun Yun was born in 1971. He received the B.S., M.S. and Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin, China in 1993, 1995, and 1998. He is a professor of the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include network security and information.

Hao Luo was born in 1979. He received the B.S., M.S. and Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin, China in 2000, 2002, and 2006. He is currently an associate professor in Beijing University of Posts and Telecommunications. His research interests include computer network, network security.

Zhigang Wu was born in 1972. He received the B.S., M.S. and Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin, China in 1994, 1996, and 2000. He is currently an associate professor in Beijing University of Posts and Telecommunications. His research interests include computer network, network security.

Shuzhuang Zhang was born in 1982. He received the B.S. and M.S. degrees in computer science from Yan Shan University, Qinhuangdao, China in 2004 and 2007, respectively. He received the Ph.D. degrees in computer science from Harbin Institute of Technology, Harbin,China in 2011. He is currently a postdoctoral fellow in Beijing University of Posts and Telecommunications. His research interests include computer network, network security.