TELKOMNIKA, Vol. 11, No. 8, August 2013, pp. 4317~4326
e-ISSN: 2087-278X

A Framework For Concept Drifting P2P Traffic Identification

Minghao Ai*, Guanghui Yan
School of Electronic and Information Engineering, Lanzhou Jiaotong University, No. 88 West Anning Road, Lanzhou city, Gansu Province 730070, P.R.China
Corresponding author, e-mail: aiminghao@sina.com[1], yangh9805@qq.com[2]

Abstract

Identification of network traffic using port-based or payload-based analysis is becoming increasing difficult with many Peer-to-Peer (P2P) application using dynamic ports, masquerading techniques, and encryption to avoid detection. To overcome this problem, several machine learning technique were proposed to classify P2P traffics. But in the real P2P network environment, new communities of peers often attend and old communities of peers often leave. It requires the identification methods to be capable of coping with concept drift, and updating the model incrementally. In this paper, we present a concept-adapting algorithm CluMC which is based on streaming data mining techniques to identify P2P applications in Internet traffic. The CluMC use micro-cluster structures which contain potential micro-cluster structures and outlier micro-cluster structures to classify the P2P traffic and discover the concept drift with limited memory. Our performance study over a number of real data sets that we captured at a main gateway router demonstrates the effectiveness and efficiency of our method.

Keywords*: P2P traffic classification, concept drift adapting, micro-cluster*

1. Introduction

With the development of Internet technology, accurate classification and identification of internet applications play a very important role in many network tasks like: fault monitoring, network planning, flow prioritization and internet security. Over the last few years, the utilization of peer to peer (P2P) application, such as file sharing, VoIP, and VoD, media streaming is growing dramatically and becoming a significant portion of the whole Internet traffic.

P2P file-sharing network have many advantage over standard client-server approaches to data distribution, include improved robustness, scalability and diversity of available data. Despite the P2P application brings convenience to us, it also gives rise to many problem. P2P technology allows data transmitted between any two peers in P2P network without limiting the flow and bandwidth. This transfer model exacerbates network congestion directly and causes the performance degradation of traditional client-server applications. Furthermore, Internet virus or pirate files can spread rapidly on account of P2P transmission model. So, how to identify P2P flow became an immediate problem of urgent need to solve.

Classification and identification of P2P application has received lots of attention in the last few years constituting an important research area in consequence of its growth. Many papers about P2P identification have published at international conference and academic journal since the year 2000. The initial approach to identify the P2P application relies on mapping application to well-know port numbers and has been very success in the past. Refs [1] analyze P2P traffic that matches the default port numbers the corresponding P2P application use and gain a reliable result. But [2, 3] confirm that it is ineffective now because P2P file sharing use dynamic ports for communication at present.

The second work is payload-based analysis that identifies P2P traffic by searching the characteristic signature of known application's data packet. [4, 5] have employed payload-based analysis in their classifiers to identify P2P applications and gain a better result. This approach, however, faces several technical problems. First, these methods must update its signature list frequently for the purpose of addressing the change of P2P applications. Moreover, payload-based techniques break the principle of privacy because information of the package is read

Received January 15, 2013; Revised April 28, 2013; Accepted May 20, 2013

entirely not only the header. Finally, these techniques fail to detect encrypted traffic and many P2P applications begin to use encryption now.

In the other way, method based on flow connection patterns of P2P traffic is proposed to overcome aforementioned technical problems. Karagiannis T, etc developed a systematic methodology to identify P2P flows at transport layer without relying on packet payload in [6]. They found that there were two obvious characteristics in P2P network transport layer. One was that about 2/3 P2P applications concurrently used both TCP and UDP to transfer data. Another was in every P2P Peer, {IP, port} pairs for which the number of distinct connected IPs is equal to the number of distinct connected ports. This method named PTP is not affect by the payload encryption but only classified traffic into P2P or non-P2P. So, it can not do an accurate classification for specific P2P traffic.

To overcome above-mentioned limitation, Machine Learning as a powerful tool in data analysis is employed to identify P2P applications. Machine Learning usually has two key phrases: firstly, it builds a model by training on a representative data set where the P2P applications are known; the next is using this model to determine the class of unknown flows. There are many maturing Machine Learning algorithms, include Naïve Bates, Decision Tree, SVM, etc, are imported to identify the Internet traffic and each of them can achieve a satisfactory effect in [7-9]. However, identification P2P application based ML methods confront several challenges:

1. In contrast, labeled samples are very scare than unlabeled. The classifiers which traditional supervised learning method produce on few labeled samples do not perform well when they are used to classify unknown samples.

2. New categories may appear as the time increase. Not all types of applications generating sample are know in the training set, and new ones may appear over time. Show that traditional supervised method focus on a mapping of each unlabeled sample into a known class but can do nothing for detecting new types of samples [10].

3. Concept drift of P2P flow can not be neglected in the real P2P flow environment. For P2P applications, new communities of peers often attend and old communities of peers often leave, which make the distribution of samples changing dynamically. So, the optimal classifiers built on old samples may not suitable new samples like Figure 1. In response to this phenomenon, lots of researches about concept drift have emerged in Machine Learning area such as Ensemble of Classifiers, CVFDT, etc. Raahemi, B etc identified P2P flows on use of CVFDT and the concept drift was observed clearly in their experiment result [11]. But almost all existing researches presume a prerequisite that the label of every sample, whatever training set or test set, must be known beforehand. However, in reality it is quite difficult.

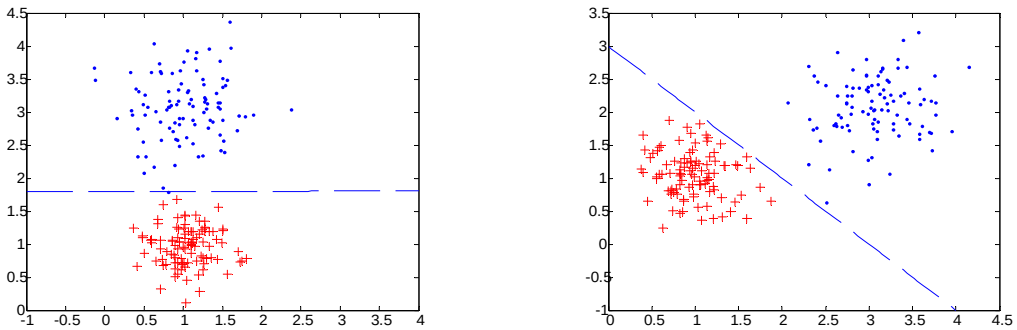

Figure 1. Concept Drift

To cope with aforementioned problem, we proposed an approach CluMC, a novel approach based on a data stream algorithm DenStream which is presented in [12]. DenStream proposed by Cao F in 2006 is used to tackle several problems in the data stream clustering like discovering arbitrary shape of cluster, tracking evolutional data stream etc. A significant feature of the DenStream is introducing core-micro-cluster to cover and summarize the natural cluster, while proposing potential core-micro-cluster and outlier core-micro-cluster to maintain and

distinguish the potential clusters and outliers. But DenStream, as a data stream cluster algorithm, mainly concerned about the distribution of data points in the damped time window, whereas our purpose is to use the old samples to identify the candidate samples. So, CluMC do not need to focus on the some problems such as the function of damped time window, overlap of core-micro-clusters and so on, but put emphasis to build a concise model and increase classification efficiency.

CluMC consists of two main steps. Firstly, the method of micro-cluster is employed to predict the class of unlabeled samples. Secondly, the classic clustering algorithm is used to obtain the label of unknown micro-cluster.

2. Fundamental Concept

The problem of clustering is considered in the damped window model, in which the weight of each data point decrease exponentially with time t via a fading function $f(t)=2^{-\lambda t}$ where $\lambda>0$.

Definition 1. (core object) A core object is defined as an object, in whose neighborhood the overall weight of data points is at least an integer μ

Definition 2. (density-area) A density area is defined as the union of the ε neighborhoods of core objects.

Definition 3. (core-micro-cluster) A core-micro-cluster, abbr. c-micro-cluster, at time t is defined as $CMC(\omega,c,r)$ for a group of close points $p_{i_1}, p_{i_2},..., p_{i_n}$ with time stamps $T_{i_1},\cdots,T_{i_n}$. $\omega=\sum_{j=1}^{n} f(t-T_{i_j})$, $\omega\geq\mu$ is the weight $c=\frac{\sum_{j=1}^{n} f(t-T_{i_j})\,p_{i_j}}{\omega}$ is the center. $r=\frac{\sum_{j=1}^{n} f(t-T_{i_j})\,dist(p_{i_j},c)}{\omega}$, $r\leq\varepsilon$, is the radius, where $dist(p_{i_j},c)$ denotes the Euclidean distance between point p_{i_j} and the center c.

Notice the weight of c-micro-cluster must be above or equal to μ and the radius must be below or equal to ε. So the number of c-micro-cluster is much larger than natural, but significantly smaller than the number of data. In addition, we will set the parameter μ and ε in order to make one c-micro-cluster only include data with one class.

Definition 4. (potential c-micro-cluster) A potential c-micro-cluster, abbr. p-micro-cluster, at time t for a group of close point $p_{i_1},\cdots,p_{i_n}$ with time stamps $T_{i_1},\cdots,T_{i_n}$ is defined as $\{\overline{CF^1},\overline{CF^2},\omega\}$. $\omega=\sum_{j=1}^{n} f(t-T_{i_j})$, $\omega\geq\beta\mu$, is the weight. β is the parameter to determine the threshold of outlier relative to c-micro-cluster, where $0<\beta<1$. $\overline{CF^1}=\sum_{j=1}^{n} f(t-T_{i_j})p_{i_j}$, $\overline{CF^2}=\sum_{j=1}^{n} f(t-T_{i_j})p_{i_j}^2$. The center of p-micro-cluster is $c=\frac{\overline{CF^1}}{\omega}$ and the radius is $r=\sqrt{\frac{|\overline{CF^2}|}{\omega}-\left(\frac{\overline{CF^1}}{\omega}\right)^2}$, $r\leq\varepsilon$.

Definition 5. (outlier micro-cluster). A outlier micro-cluster, abbr. o-micro-cluster at time t for a group of close point $p_{i_1},\cdots,p_{i_n}$ with time stamps $T_{i_1},\cdots,T_{i_n}$ is defined as $\{\overline{CF^1},\overline{CF^2},\omega,\ t_o\}$. The definition of $\overline{CF^1}$, $\overline{CF^2}$, ω, center and radius are the creation time of the p-micro-cluster. $t_o=T_{i_1}$ denote the creation time of the o-micro-cluster, where $\omega<\beta\mu$.

Consider a p-micro-cluster $c_p=\{\overline{CF^1},\overline{CF^2},\omega\}$, if no point are merged by c_p for time interval δt, $c_p=\{2^{-\lambda\delta t}\cdot\overline{CF^1},2^{-\lambda\delta t}\cdot\overline{CF^2},2^{-\lambda\delta t}\cdot\omega\}$. If a point p is merged by c_p,

$c_p = \{\overline{CF^1} + p, \overline{CF^2} + p^2, \omega + 1\}$. Similar procedure can prove the property of o-micro-clusters. So, we can see the p-micro-clusters and o-micro-clusters can be maintained incrementally.

3. Algorithmic Framework

Our primary goal is to build a classifiers model that can recognize the characteristic of unlabeled P2P flow, whose distributions may be experiencing concept drift. Formally, the P2P traffic classification problem can be defined as follows: Given a set of flows $X = \{X_1, X_2, \cdots, X_N\}$ where each flow vector X_i is the Characteristic Attributes of P2P traffic $\{x_{i_1}, x_{i_2}, \cdots, x_{i_s}\}$. Set $\{t_{i_1}, t_{i_2}, \cdots, t_{i_n}\}$ is the corresponding time when X arrive. A set of traffic classes $Y = \{Y_{X_1}, Y_{X_2}, \cdots, Y_{X_N}\}$ is known, where Y_{X_i} is the label of X_i. The goal of P2P traffic classification is to build a mapping $f: X \to Y$ such that flow X_i is assigned to only one traffic class.

3.1. Model Building

Training samples $X = \{X_1, X_2, \cdots, X_N\}$ is gathered into several p-micro-clusters $\{P_1, P_2, \cdots, P_m\}$ according to DenStream. Parameters, μ and ε are chosen to guarantee that each p-micro-cluster contains samples have the same label. It is shown in Figure 2. We define that the label of a p-micro-clusters is equals to the label of its interior samples. So, the set of p-micro-clusters $P = \{P_1, P_2, \cdots, P_m\}$ has its corresponding label set $Y = \{Y_{C_1}, Y_{C_2}, \cdots, Y_{C_m}\}$.

We maintain a group of p-micro-clusters and a group of o-micro-clusters in the memory space separately recorded as potential-buffer and outlier-buffer. In the initialization status, the set of p-micro-clusters $P = \{P_1, P_2, \cdots, P_m\}$ is put in potential-buffer and outlier-buffer is null. These two buffers constitute our classification model.

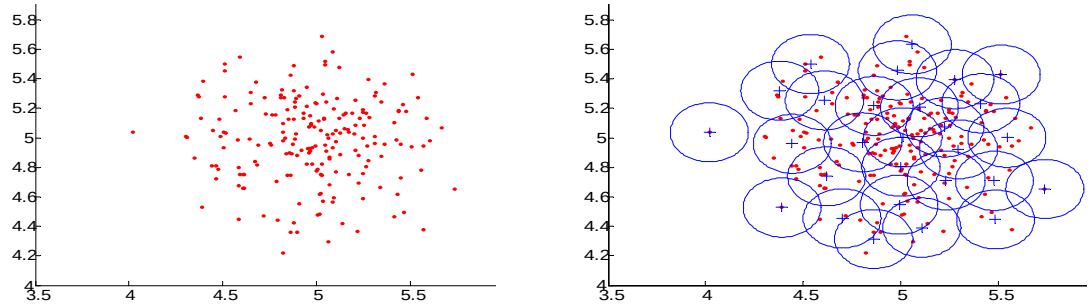

Figure 2. Samples are Covered by p-micro-clusters

3.2. Classification Algorithm Description

CluMC can be divided into two parts: (1) Online part is predicting the label of the new data. (2) Offline part is coping with concept drift and appearance of new class by cluster.

3.2.1. Online Part

When a new data X_t arrives at time t, the distances from X_t to every center of micro-cluster, both p-micro-clusters and o-micro-clusters, are calculated, and if it short than the radius of a micro-cluster C_t, we call the X_t is absorbed by C_t. About C_t, there are three possible cases as follow (see Algorithm 1 for detail):

1. C_t is a labeled p-micro-cluster, so we believe that the label of X_t is the same to the label of C_t. It is shown in Figure 3.

2. C_t is an o-micro-cluster but its weight ω great than or equal to $\beta\mu$ when it absorbs X_t. In this case, we consider that the C_t has grown into a p-micro-cluster. We insert C_t to p-micro-cluster buffer P and use DBSCAN on P in order to determine C_t is an evolved micro-cluster or an outlier as Figure 4, the pink circle represent a new p-micro-cluster. This will be described in more detail at offline part in blow.
3. C_t is an unlabeled p-micro-cluster. It means that C_t represent a new class but we do not know its accurate name temporarily. Although we cannot label X_t immediately, we do not discard it as the noise data. Instead, we gather it into a micro-cluster and wait for the professionals to recognize it.
4. C_t is an o-micro-cluster but its weight ω still less than $\beta\mu$ after it absorbs X_t. It show that the C_t do not grow to be a p-micro-cluster. We reserve it into the o-micro-cluster buffer and do nothing before its weight great than the $\beta\mu$ or less than ξ.

Algorithm 1: Online part

Input：p-micro-cluster buffer $P=\{P_1,P_2,\cdots,P_m\}, Y=\{Y_{C_1},Y_{C_2},\cdots,Y_{C_m}\}$;

o-micro-cluster buffer $O=\{\mathrm{O}_1,\mathrm{O}_2,\cdots,\mathrm{O}_\mathrm{k}\}$;

Parameters μ and ε;

Parameters β;

For *t=1,2,3, …* as long as new data arrive) do

1. Receive the new arriving data X_t and find its nearest micro-cluster C_t

If *r* (The radius of new C_t) < ε then

2. Merge X_t into C_t;

If C_t belongs to P then

3. Classify X_t to the label of C_t, $Y_{X_t}=Y_{C_t}$;

Else (C_t belongs to O)

If ω (the new weight of C_t) > $\beta\mu$ then

4. Insert C_t to P;
5. Send a clustering requirement;

End If

End If

Else (d > ε, X_t can not absorbed by C_t)

6. Create a new o-micro-cluster by X_t

and insert it to the o-micro-cluster buffer O;

End If

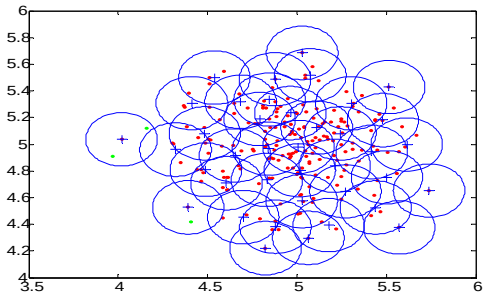

Figure 3. New Samples are in the Interior of the Labeled p-micro-cluster

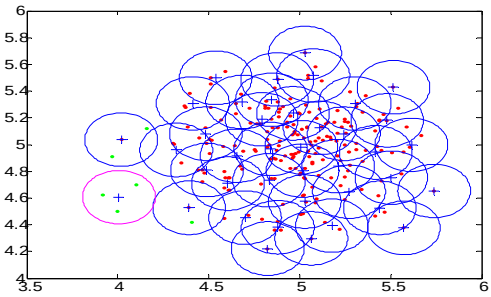

Figure 4. C_t is an Evolved Micro-cluster

3.2.2. Offline Part

There are two tasks in the offline part: dealing with the new p-micro-cluster and managing micro-clusters. In this part, we consider that this problem follows the cluster hypothesis which states that relevant data tend to be more similar to each other than to non-relevant data. So, we employ the DBSCAN algorithm to determine the new micro-cluster's label.

1. When an o-micro-cluster grows into a p-micro-cluster, the distribution of data is different than the training set. It means that the original model is no longer applicable. Then, the DBSCAN will be called on the p-micro-cluster buffer. DBSCAN is a clustering algorithm which relies on a density-based notion of clusters. When a clustering request arrives, a variant of DBSCAN algorithm is applied on the p-micro-cluster buffer to get the final result of clustering. Each p-micro-cluster is regarded as a virtual point located at the center. If this new p-micro-cluster is density-reachable from other labeled clusters, they can be gathered into a cluster. So, we can determine the label of this p-micro-cluster according to the cluster hypothesis. Else, we consider that this p-micro-cluster belong to a new class.

2. In the data stream environment, we only care about the current data. For each existing p-micro-cluster P_i, if no new point is merged into it, its weight will decay gradually. If the weight is below $\beta\mu$, we can delete P_i, because P_i has became an outlier. Thus, we need to check the weight of each p-micro-cluster periodically. Here, we use T to denote this interval. Another problem is the number of o-micro-cluster may continuously increase as data stream proceed. In fact we cannot wait infinite time to observe whether an o-micro-cluster could become a p-micro-cluster or not. But we should provide opportunity for an o-micro-cluster to grow into a p-micro-cluster. So, we define $\xi = \frac{2^{-\lambda(t-t_o+T)}-1}{2^{-\lambda T}-1}$, which is a function of t (i.e., current time) and t_o (i.e., the creation time of the o-micro-cluster). Each o-micro-cluster is checked at every T time periods. If the weight of an o-micro-cluster O_i is lower than ξ, we consider O_i is hardly possible to grow into a p-micro-cluster and we delete it from the o-micro-cluster buffer. All of the detailed procedure is described in Algorithm 2.

Algorithm 2: Offline part

Input : p-micro-cluster buffer $P=\{P_1,P_2,\cdots,P_m\}, Y=\{Y_{C_1},Y_{C_2},\cdots,Y_{C_m}\}$;

o-micro-cluster buffer $O=\{\mathrm{O}_1,\mathrm{O}_2,\cdots,\mathrm{O}_k\}$;

time interval T;

new p-micro-cluster buffer C_t ;

For *t=1,2,3,…*

If *t* mod T = 0 then

For each p-micro-cluster P_i do

If ω_{P_i} (the weight of P_i) < $\beta\mu$ then

1. Delete P_i;

End if

End for

For each o-micro-cluster O_i do

2. $\xi = \frac{2^{-\lambda(t-t_o+T)}-1}{2^{-\lambda T}-1}$

If ω_{O_i} (the new weight of O_i) < ξ then

3. Delete O_i

End if

End for

If a clustering request arrives then

4. Clustering on *C* based on DBSCAN by the virtual center of every P_i;

If C_t can be gathered with a cluster which have a certain label then

5. Y_{C_t} is equal to this label;
   Else
6. The label of C_t is considered as a new class and marked $Y_{C_t} = Unknown_t$;
7. Call the Professional to identity this new class;
   End If
   End If

3.3. Parameter Debate

For ε, if it is too large, it may mess up different cluster. If it is too small, it requires a corresponding smaller μ which will result in a larger number of micro-cluster. Therefore, we set $\varepsilon = \alpha d_{\min}$, where α between *0* and *1*, is fixed by the requirement of precision. $d_{\min}$ is the minimal Euclidean distance between the nearest pair of samples in different clusters.

Theorem: Each p-micro-cluster contains samples have the same label under this ε.

Proof: Here we use the *reductio ad absurdum*. We can assume that there is a p-micro-cluster contains samples $\{X_1, X_2\}$, where $X_1 = \{x_1, x_2, \cdots, x_l\}$ belongs to a class Y_1 and $X_2 = \{x_{l+1}, x_{l+2}, \cdots, x_n\}$ belongs to another class Y_2. We define $d(x_i, x_j)$ is Euclidean distance of x_i and x_j, where $x_i \in X_1$ and $x_j \in X_2$. Because of $\{X_1, X_2\}$ is contained by a p-micro-cluster, so $d(x_i, x_j)$ must less than the radius of this p-micro-cluster and less than the ε. But the $\varepsilon = \alpha d_{\min}$ must less than the minimal Euclidean distance which between the nearest pair of samples in different clusters. Obviously, this result is a contradiction. Therefore, the initial assumption must be false and hence the claim-each p-micro-cluster contains samples have the same label under this ε.

After the setting of ε, we gain some p-micro-cluster by this ε on the training dataset. μ can be set by the average number of points in each clusters. λ, β and T can be estimated by the character of data. We can use our experience to set them.

4. Experiment

4.1. Data for experiment

The training datasets are established manually. It is obtained from several hosts running P2P applications including: PPLive, PPStream, Bittorrent, Xunlei and Skype. Packets of these hosts running different application are firstly captured on a Gbps Ethernet link. We collected a total of 429767 packets at different times in a day. All of packets can be analyzed into 8 attributes consist of ID, time of arrival, source IP, destination IP, protocol, length, source port and destination port. It is shown in Figure 5.

Then, these packets are converted into some flows according to the five tuples (srcIP, desIP, Port, srcPort, desPort). If some packets have the same five tuples and adjacent time intervals of them are less than 60s, they are considered to belong to a same flow. We put all the packets into 21085 flows and Table 1 shows the composition of our training dataset.

Table 1. Composition of P2P application

ID	Application	Class	Number of samples	Percentage
1	PPLive	P2P Streaming Media	4857	23.04%
2	PPStream	P2P Streaming Media	4143	19.65%
3	Bittorrent	P2P File Download	3984	18.89%
4	Xunlei	P2P File Download	4366	20.70%
5	Skype	P2P IM	3735	17.72%

Table 2. Predict and actual matrix

		Predicted	
		Y	N
Actual class	Y	TP	FN
	N	FP	TN

```
10746 42.916223   192.168.52.209      125.46.55.194       UDP    83    Source port: 12372  Destination port: 62412
10747 42.917398   222.182.172.107     192.168.52.209      UDP    1103  Source port: 44455  Destination port: 12372
10748 42.917618   192.168.52.209      222.182.172.107     UDP    83    Source port: 12372  Destination port: 44455
10749 42.917724   192.168.52.209      222.182.172.107     UDP    83    Source port: 12372  Destination port: 44455
10750 42.917811   192.168.52.209      222.182.172.107     UDP    83    Source port: 12372  Destination port: 44455
10751 42.919378   123.131.151.57      192.168.52.209      UDP    1103  Source port: 9690  Destination port: 12372
10752 42.919581   192.168.52.209      123.131.151.57      UDP    83    Source port: 12372  Destination port: 9690
10753 42.919989   123.131.151.57      192.168.52.209      UDP    1103  Source port: 9690  Destination port: 12372
10754 42.920225   192.168.52.209      123.131.151.57      UDP    83    Source port: 12372  Destination port: 9690
10755 42.920360   210.83.64.206       192.168.52.209      UDP    1118  Source port: 40934  Destination port: 12372
10756 42.920572   192.168.52.209      210.83.64.206       UDP    83    Source port: 12372  Destination port: 40934
10757 42.920675   192.168.52.209      210.83.64.206       UDP    83    Source port: 12372  Destination port: 40934
10758 42.920763   192.168.52.209      210.83.64.206       UDP    83    Source port: 12372  Destination port: 40934
10759 42.921188   192.168.52.209      101.66.93.32        UDP    181   Source port: 12372  Destination port: 4312
10760 42.921252   192.168.52.209      113.239.204.144     UDP    181   Source port: 12372  Destination port: 43914
10761 42.921283   192.168.52.209      116.25.239.42       UDP    181   Source port: 12372  Destination port: 5823
10762 42.922147   192.168.52.209      210.83.64.206       UDP    83    Source port: 12372  Destination port: 40934
10763 42.922246   192.168.52.209      210.83.64.206       UDP    83    Source port: 12372  Destination port: 40934
10764 42.937122   222.182.172.107     192.168.52.209      UDP    1103  Source port: 44455  Destination port: 12372
10765 42.937384   192.168.52.209      222.182.172.107     UDP    83    Source port: 12372  Destination port: 44455
10766 42.948473   117.85.144.178      192.168.52.209      UDP    1103  Source port: 46402  Destination port: 12372
10767 42.948670   192.168.52.209      117.85.144.178      UDP    83    Source port: 12372  Destination port: 46402
10768 42.952659   110.16.11.162       192.168.52.209      UDP    1103  Source port: 31876  Destination port: 12372
10769 42.952881   192.168.52.209      110.16.11.162       UDP    83    Source port: 12372  Destination port: 31876
10770 42.953264   110.16.11.162       192.168.52.209      UDP    1103  Source port: 31876  Destination port: 12372
10771 42.953446   116.21.28.170       192.168.52.209      UDP    123   Source port: 35119  Destination port: 12372
10772 42.953529   192.168.52.209      110.16.11.162       UDP    83    Source port: 12372  Destination port: 31876
10773 42.953678   192.168.52.209      116.21.28.170       UDP    1181  Source port: 12372  Destination port: 35119
10774 42.954666   110.16.11.162       192.168.52.209      UDP    1103  Source port: 31876  Destination port: 12372
10775 42.954880   192.168.52.209      110.16.11.162       UDP    83    Source port: 12372  Destination port: 31876
10776 42.956023   116.21.28.170       192.168.52.209      UDP    123   Source port: 35119  Destination port: 12372
10777 42.956242   192.168.52.209      116.21.28.170       UDP    1181  Source port: 12372  Destination port: 35119
```

Figure 5. Information of P2P Network Packet

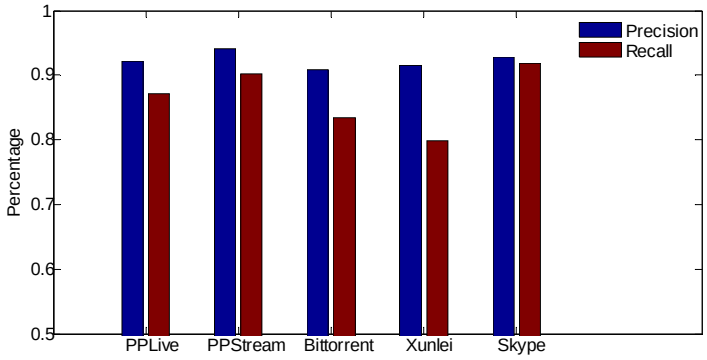

Figure 6. Comparison of P2P Applications

We extract the characteristic values from flows as follow:

1. min-IAT Minimum interval of packets arrival in a flow
2. mean-IAT Average interval of packets arrival in a flow
3. max-IAT Maximum interval of packets arrival in a flow
4. min-LP Minimum length of packets in a flow
5. mean-LP Average length of packets in a flow
6. max-LP Maximum length of packets in a flow
7. bandwidth
8. source port
9. destination port

We can build the p-micro-clusters model by this characteristic values and use our algorithm to predict the P2P flow samples of the new arrival.

4.2. Precision of the Classifier

To test the method's effectiveness, we use Precision and Recall to measure the performance of our algorithm. A sample has 4 different prediction outputs, two are correct, namely True Positive (TP) where an example is actually P2P and it is classified as P2P and True Negative (TN) where an example is actually NonP2P and it is classified as NonP2P. Accordingly, there could be two false predictions, namely False Positive (FP) where an example is classified as P2P but actually it is NonP2P and False Negative (FN) where an example is classified NonP2P but actually it is P2P. They are visually represented in the confusion matrix of Table 2.

According to the four different prediction outputs, Precision and Recall can be defined as:

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

We apply our model to the test samples and each 2000 samples will be used as a test point. The result is shown as Figure 6.

Beside Precision and Recall, we define CluMC overall success rate Correctness as:

$$Correctness = \frac{Success_{out}}{Total_{in}} \quad (3)$$

Where $Total_{in}$ is the total number of test dataset and $Success_{out}$ is the number of samples which are successful prediction. Correctness value is in the range of *(0, 1)*. Figure 7 show the Correctness of the CluMC algorithm.

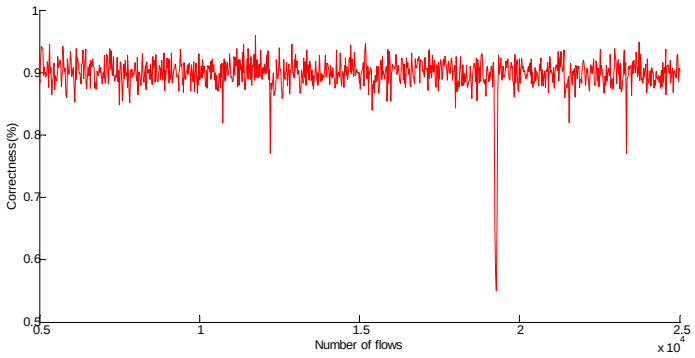

Figure 7. Correctness of the CluMC Algorithm

Through the Figure 7, we can obverse that Correctness drop to 0.5501 when the number of test samples increase to 19274. After checking characteristic values of misclassified flows, we find that the interval of packets arrival in these flows become shorter especially in Bittorrent and Xunlei. This leads to the change of the distribution of samples and it means the concept drift is appeared at this period, which resulted in the drop of Correctness. However the experiment results also shown that the Correctness quickly increased to 0.8720 and 0.9406. This demonstrates that the CluMC detected the concept drift, and quickly adjusted the trained model. Therefore, our method is feasible and effective.

5. Conclusion

This paper proposes and evaluates CluMC, an effective and efficient method for P2P application identification. Comparing with previous machine learning methods, this method has two main advantages: First, the structures of p-micro-cluster and o-micro-cluster are used to not only cover cluster of arbitrary shape in the database, but also limit the calculate and memory consumption with precision guarantee. Second, this method employ two parts, online and offline, to handle concept drift of unlabeled P2P flow samples. This is tested and verified in a real dataset with more than 50 thousands samples, which is captured at our campus gateway. But in our algorithm, the parameters are pinned. It is difficult to adapt to the high-speed dynamic data stream environment. Beside, we use the Euclidean distance to measure the distance

between points and micro-clusters and it is the lack of basis. How to solve these two problems is in the future work.

References

[1] S Saroiu, KP Gummadi, RJ Dunn, SDGribble, HM Levy. *An analysis of Internet content delivery systems*. Proceeding of the 5th symposium on Operating systems design and implementation. 2002; 315-327.

[2] T Karagiannis, A Broido, M Faloutsos, K claffy. *Transport Layer Identification of P2P Traffic*. In IMC'04, Taormina, Italy. 2004; 25-27.

[3] S Sen, O Spatscheck, D Wang. *Accurate, Scalable In-Network Identification of P2P Traffic Using Application Signatures*. In WWW2005, New York, USA. 2004; 17-22.

[4] A Spognardi, A Lucarelli, RD Pietro. *A methodology for P2P file-sharing traffic detection*, in Proceedings of the Second International Workshop on Hot Topics in Peer-to-Peer Systems Volume 00 HOT-P2P'05. 2005; 52-61.

[5] H Bleul, EP Rathgeb, S Zilling. *Evaluation of an efficient measurement concept for P2P multiprotocol traffic analysis.* Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Application. 2006; 414-423.

[6] T Karagiannis, A Brodio. *Transport layer Identification of P2P Traffic,* in Proceeding of 4th ACM SiGCOMM conference on Internet Measurement. Sicily, Italy. ACM. 2004: 121-134

[7] AW Moore, D Zuev. *Internet traffic classification using Bayesian analysis techniques*. ACM SIGMETRICS Performance Evaluation Review. 2005; 33(1): 50-60.

[8] AW Moore, D Zuev. *Discrimination for use in flow-based classification.* Technical report, Intel Research, Cambridge. 2005.

[9] F Hernandez, AB Nobel, FD Smith, K Jeffay. *Statistical Clustering of Internet Communication Patterns.* In Proceedings of Symposium on the Interface of Computing Science and Statistics. 2003.

[10] J Erman, A Mahanti, C Williamson, M Arlitt. *Internet Traffic Identification using Machine Learning.* In GLOBECOM' 06, San Francisco, USA. 2006.

[11] Watanabe O. *Simple Sampling Techniques for Discovery Science*. IEICE Transactions on Information and Systems. 2000; E83-D(1): 19-26.

[12] F Cao, M Ester, W Qian, A Zhou. *Density-based clustering over an evolving data stream with noise.* In SIAM Conference on Data Mining (SDM). 2006; 326-337.