A semi-supervised clustering algorithm for real network traffic with concept drift

IOP Conf. Series: Materials Science and Engineering **569** (2019) 052045 doi:10.1088/1757-899X/569/5/052045

A semi-supervised clustering algorithm for real network traffic with concept drift

Qu Hua[1], Jiang Jie[1], Zhao Jihong[1,2], Zhang Yanpeng[1]

[1]Xi'an Jiaotong University, Shaanxi Xi'an, China.

[2]Xi'an University of Posts & Telecommunications, Shaanxi Xi'an, China.

Abstract: Traffic classification has been widely applied for networking. Previous works paid much attention to static network traffic. In this paper, we propose a new strategy for the semi-supervised clustering algorithm to deal the concept drift in a dynamic network, as well as updating the model incrementally. Moreover, our algorithm can find new clusters and reduce the impact of noises. The results of simulation demonstrate the effectiveness of semi-supervised clustering algorithm.

1. Introduction

With the rapid development of network and information technology, the network traffic classification has drawn plenty of attention and plays an important role in network resource management, intrusion detection and other fields[1]. Over the last few years, traffic classification techniques based on Machine Learning have been widely used to alleviate limitations which imposed by traditional network traffic classification techniques. Park et al.[2]use a Genetic Algorithm in the feature selection and compare the three classifier: the C4.5Decision Tree, the Naïve Bayes with Kernel Estimation, and Reduced Error Pruning Tree. The experiments indicate that the C4.5 has better performance. Singh et al.[3]employ unsupervised K-means and the expectation maximization algorithm to cluster the network traffic. Hyun-Kyo Lim et al.[4]treat the payload of packets as image and use deep learning to classify the network traffic. But none of the above traffic classification methods above takes into account that the actual traffic classification should be a dynamic process and contains outliers.

Concept drift[5]always appears in real network environment, such as the changes of user's interests, new traffic attend and old traffic leave, or delays caused by network congestion, and the distribution of data is time-varying. The most of classification algorithms can't address concept drift problems well, but there are still several methods can deal with concept drift by using time windows or weighted examples. Gang Liu[6]proposed an improving CVFDT method for concept drift in data Stream, but performed bad when data mixed with noise or outlier. Furthermore, the role of outlier and clusters are often exchange cause of the dynamic process. Frank et al.[7]employed an incremental learning decision tree algorithm to handled concept drift. However, decision tree algorithm can't handle data with unlabeled well and can't to discover new classes either.

In this paper, we propose a novel semi-supervised clustering algorithm, called *TscanCluster*, to discoveries clusters of arbitrary shape, tracks evolutional data and identity new classes etc. Firstly, we consider the dynamic clustering problem in the damped window model[8]，in which the weight of each data decreases exponentially with a fading function. Secondly, we define three clustering states: core-cluster, candidate-cluster and outlier-cluster, where the core-cluster confirm with labeled data, candidate-cluster can evolve to core-cluster when meets the conditions, outlier-cluster can also evolve

to candidate-cluster or disappear too. Lastly, if new core-cluster unlabeled and the number of this cluster meets the minimum number of the existing core-clusters, we manually labeled this core-cluster. The main contribution of this work are following:

- We propose a novel semi-supervised clustering algorithm to dynamically discoveries clusters of arbitrary shape and solves the concept drift.
- Evolution of candidate-cluster to discover new classes.
- Reduce the impact of the outlier and increase the stability of the model

The remainder of this paper is organized as follows: Section 2 introduces the basic concepts; Section 3 discusses some detailed techniques of TscanCluster; In section 4, we employ experiments to demonstrate it and we conclude the paper in section 5.

2. Fundamental Concepts of Algorithm

The clustering on evolving data are always computed based on damped model with certain time intervals, in which the weight of each data decrease exponentially with time t via a fading function[9] $f(t)=2^{-\lambda t}$,where $\lambda>0$, and the higher the value of λ , the lower importance of historical. The exponentially fading function is widely used in the sequential model where it is worth to gradually ignore the past behavior.

Definition 1.(core-object, density-area) Core-object [10]is defined as an object, in whose neighborhood the overall weight of data is at least an integer μ . Density-area is defined as the union of the ε neighborhoods of core-object.

Definition 2.(core-cluster) Core-cluster is a defined cluster obtained from the training with labeled samples, at time t is defined as $coc=(w,c,r)$ for the data set $\{d_{i_1},d_{i_2},...,d_{i_m}\}$ with the stamps $\{t_{i_1},t_{i_2},...,t_{i_m}\}$, $w=\sum_{j=1}^{m} f(t-t_{i_j})$ is the weight, and $w\geq\mu$. $c=\frac{\sum_{j=1}^{m} f(t-t_{i_j})d_{i_j}}{w}$ is the center of cluster, $r=\frac{\sum_{j=1}^{m} f(t-t_{i_j})dist(d_{i_j},c)}{w}$ is the radius of cluster, $r<\varepsilon$, and $\mathrm{dist}(d_{i_j},c)$ is the Euclidean distance between d_{i_j} and c .

Definition 3.(candidate-cluster) Candidate-cluster is potential cluster, at time t is defined as $cadoc=(w,c,r,\beta)$. The definition of parameters c,r is equivalent to core-cluster, but $w>\beta\mu$, $0.5<\beta<1$ and $r\leq\varepsilon$

Definition 4.(outlier-cluster) Outlier-cluster is outlier cluster, at time t is defined as $ooc=(w,c,r,t_0,\beta)$, similar to candidate-cluster, but $w<\beta\mu$, and t_0 is the creation time of the outlier-cluster.

Consider candidate-cluster, if new data is merge by $cadoc$ for time interval δ , $w'=w+1$ $m'=m+1$ to update c,r , or $w',c',r'=f(\delta)(w,c,r)$ .Similar procedure to deal with outlier-cluster, such that we can maintain candidate-cluster and outlier-cluster incrementally.

3. Clustering model *TscanCluster* for network traffic

In this section, we describe the *TscanCluster* model for network traffic classification, whose data distributions may have the phenomenon of concept drift. Given a set of data vector $X=\{X_1,X_2,...,X_N\}$, where each vector X_i has the features of network traffic defined as $\{X_{i1},X_{i2},...,X_{ik}\}$, and $\{t_{i1},t_{i2},...,t_{ik}\}$ is corresponding time when X_i arrive. A set of traffic classes $Y=\{Y_{X_1},Y_{X_2},...,Y_{X_N}\}$,where Y_{X_i} is the label of X_i

3.1 The flow of TscanCluster model

In the initialization status, the training samples $X=\{X_1,X_2,...,X_N\}$ is gathered into several core-cluster $C_{l1}=\{C_1,C_{C_2},...,C_{l1}\}$, which is rely on the labeled sample by manual, when $w\geq\mu$ and $r\leq\varepsilon$. The candidate-cluster $C_{l2}=\{C_1,C_2,...,C_{l2}\}$ when $w>\beta\mu$, and $l1\leq l2$. The set of outlier-cluster is null. Further, the set of C_{l1} has its corresponding label set $Y_{l1}=\{Y_{C_1},Y_{C_2},...,Y_{C_{L1}}\}$

When new data X_i arrives at time t_i , the Euclidean distance from X_i to core-cluster、candidate-cluster

and outlier-cluster are calculated, and merge X_i according to the condition. Then calculate weight of current cluster whether it meets w or not. There are three main cases as follow(see Algorithm1 for detail):

1) X_i merge into core-cluster if R_d (the new radius of core-cluster) less than ε as well as w' (the new weight of core-cluster)more than w. In this case, we believe the label of X_i is same of the label of the core-cluster.

2) X_i merge into condition-cluster if R_d (the new radius of core-cluster) less than ε and w' (the new weight of core-cluster)more than βw .At this time, we not label it unless the condition-cluster evolve into core-cluster

3) X_i merge into outlier-cluster if R_d (the new radius of core-cluster) less than ε as well as w' (the new weight of core-cluster) less than βw .Further, create a new outlier-cluster by X_i if R_d more than ε .

Algorithm1: MergeNewData

Input: **new data** $X_i = \{X_{i1}, X_{i2}, ..., X_{ik}\}$ **and** $\{t_{i1}, t_{i2}, ..., t_{ik}\}$
Core-cluster $C_{l1} = \{C_1, C_{C_2}, ..., C_{l1}\}$ **candidate-cluster** $C_{l2} = \{C_1, C_2, ..., C_{l2}\}$
Cluster-label $Y_{l1} = \{Y_{C_1}, Y_{C_2}, ..., Y_{C_{L1}}\}$
Parameters μ, ε, β

1.Try to merge X_i into its nearest core-cluster or candidate-cluster C_c :
2.**IF** R_d (the new radius of C_c) $\le \varepsilon$ **then**
3. **IF** w' (the new weight of C_c) $\ge w$ **then**
4. Merge X_i into core-cluster and update it.
5. **ELSE IF** w' (the new weights of C_c) $\ge \beta w$ **then**
6. Merge X_i into candidate-cluster and update it.
7. **END IF**
8.**ELSE**
Try to merge X_i into outlier-cluster C_o :
9. **IF** R_d (the new radius of C_c) $\le \varepsilon$ **then**
10. Merge d_i into core-cluster and update it.
11. **IF** w' (the new weight of C_c) $\ge \beta w$ **then**
12. Evolve from outlier-cluster to candidate-cluster and create new candidate-cluster
13. **END IF**
14. **ELSE**
15. Create a new outlier-cluster by X_i
16. **END IF**
17.**END IF**

3.2 Maintain candidate-cluster and outlier-cluster algorithm

When an outlier-cluster grows into a candidate-cluster, the distribution of data is different from training set, and relevant data tend to be more similar to each other than to non-relevant data. Thus we employ DBSCAN[11]algorithm to determine the new core-cluster' label. If this new core-cluster is density-reachable for other labeled core-clusters, we believe they have same label. (The detailed procedure described in Algorithm2) For each existing candidate-cluster, if no new data merge into it, it's weight well gradually decrease by t . When the weight is below βw , which means the candidate-cluster degrades to outlier-cluster. Based on this phenomenon, we check the weight of each

candidate-cluster periodically. We define T_p as check interval time according to fading function $f(t)=2^{-\lambda t}$:

$$T_p=\left\lceil \frac{1}{\lambda}\log(\frac{\beta\mu}{\beta\mu-1}) \right\rceil, \quad (1)$$

which ensure the balance of the fading weight and maximal number of candidate-cluster is $\frac{w}{\beta\mu}$.(The detailed procedure described in Algorithm3)

Another problem is that the number of outlier-cluster may continuously increase as the model goes on, which result in memory overflow and algorithm inefficiency. Thus, we define ζ to measure delete the outlier-cluster or not:

$$\zeta=\frac{2^{-\lambda(t-t_0+T_p)}-1}{2^{-\lambda T_p}-1}, \quad (2)$$

which is the measure function of weight, t_0 is the create time of outlier, t is the current time. When $t_0=t$,the $\zeta=1$. When time goes by , $\lim_{t\to\infty}\zeta=\frac{1}{1-2^{-\lambda T_p}}=\beta\mu$. That's obvious that the longer an outlier-cluster existing, the larger weight to have. If the weight of outlier-cluster is below ζ, we believe this outlier-cluster is hardly grows to candidate-cluster and delete it. Further, we employ this function to distinguish noise data from real data.(The detailed procedure described in Algorithm4)

Algorithm2: Core-clusterLabeled

Input: **Core-cluster** $C_{l1}=\{C_1,C_{C_2},...,C_{l1}\}$
New core-cluster C
Parameters μ, ε, β

1.**FOR** each new core-cluster evolve from candidate-cluster **do**
2. Clustering on C based on DESCAN by each core-cluster C_{l1}
3. **IF** C can merge into a existed core-cluster which have a certain label **then**
4. Set C's label as the certain label
5. **ELSE** IF sample size of C meets the minimum size of C_{l1} **then**
6. Consider C as a new class and marked label as *Unknows*
7. Call the professional to identity this new class.
8. **END IF**
9.**END FOR**

Algorithm3: CheckOfCandidate-cluster

Input: **Candidate-cluster** $C_{l2}=\{C_1,C_2,...,C_{l2}\}$
Outlier-cluster $C_{l3}=\{C_1,C_2,...,C_{l3}\}$
Parameters μ, β, t, T_p

1.**IF** $t \bmod T_p=0$ **then**
2. **FOR** each candidate-cluster **do**
3. **IF** w'(the new weight of C) $<\beta w$
4. Delete C in candidate-cluster and add C into outlier-cluster
5. **END IF**
6. **END FOR**

Algorithm4: CheckOfOutlier-cluster

Input: **Outlier-cluster** $C_{l3} = \{C_1, C_2, ..., C_{l3}\}$

Parameters μ, β, t, T_p

1.**IF** $t \bmod T_p = 0$ **then**
2. **FOR** each outlier-cluster **do**
3. Calculate ζ (2)
4. **IF** w'(the new weight of C) $< \zeta$
5. Delete C in outlier-cluster
6. **END IF**
7. **END FOR**

4. Experiment

4.1. Data for experiment

In the simulation experiment, the train dataset is established manually[12][13]. It is collected by the high-performance network monitor, which is intended to provide a wide variety of features to characterize flows[14]. We use a total of 273061 packets at different times in a day, as well as get some disturbance term to simulate noises and new classes. All of packets can be concluded into 8 attributes[15], which is consisting of source IP, destination IP, protocol, length, source port, destination port, time of arrival and the label. Further, We also extract the characteristic values from packets as follow:

1) First-arrive: First arrive time of packet.
2) Min-inter: Minimum interval of packet.
3) Mean-inter: Average interval od packet.
4) Max-inter: Maximum interval of packet.
5) Var-inter: Variance in packet inter-arrive time.
6) Min-length: Minimum length of packet.
7) Mean-length: Average length of packet.
8) Max-length: Maximum length of packet.
9) Total-bytes: total bytes of packet.

The detailed classification of data set is shown in table 1、Fig.1 and Fig.2..

Table 1：The detail classification of data source

Application Type	Count	Percentage	Label
Http, Https	56014	20.5%	WWW
Imap, Pop, Smtp	70116	25.6%	MAIL
Ftp-Control, Ftp-Pasv	51202	18.7%	BULK
Postgres, Sqlnet Oracle	76216	27.9%	DATABASE
KaZaA, BitTorrent	19513	7%	P2P
Ntp, DNS, Player	6000	1%	OTHER

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	172.141.85.131	122.157.149.230	TCP	64	1474 → 80 [ACK] Seq=1 Ack=1 Win=65535 [TCP CHECKSUM INCORRECT] Len=0
2	0.000005782	70.191.29.201	99.176.135.53	UDP	115	41173 → 48443 Len=65
3	0.000015915	70.191.28.53	173.180.83.84	UDP	222	11546 → 38860 Len=172[Packet size limited during capture]
4	0.000244975	172.141.77.19	75.38.242.23	TCP	74	51641 → 80 [ACK] Seq=1 Ack=1 Win=65535 [TCP CHECKSUM INCORRECT] Len=0
5	0.000330687	122.157.191.122	70.191.27.254	TCP	1422	993 → 53501 [ACK] Seq=1 Ack=1 Win=156 Len=1348[Packet size limited during capture]
6	0.000352204	122.157.191.122	70.191.27.254	TCP	1422	993 → 53501 [ACK] Seq=1349 Ack=1 Win=156 Len=1348[Packet size limited during capture]
7	0.000376582	172.141.82.107	82.151.253.10	UDP	573	4898 → 50309 Len=523[Packet size limited during capture]
8	0.000386715	172.141.84.243	116.236.123.214	TCP	598	26696 → 6890 [ACK] Seq=1 Ack=1 Win=4299 Len=536[Packet size limited during capture]
9	0.000391662	172.141.89.181	122.134.147.104	TCP	74	38344 → 3925 [ACK] Seq=1 Ack=1 Win=505 [TCP CHECKSUM INCORRECT] Len=0
10	0.000392437	172.141.90.177	186.101.60.113	ESP	230	[Packet size limited during capture]
11	0.000434280	122.157.191.122	70.191.27.254	TCP	1422	993 → 53501 [ACK] Seq=2697 Ack=1 Win=156 Len=1348[Packet size limited during capture]
12	0.000470400	122.157.191.122	70.191.27.254	TCP	720	993 → 53501 [PSH, ACK] Seq=4045 Ack=1 Win=156 Len=646[Packet size limited during cap…
13	0.000479341	194.184.132.187	172.141.75.199	TCP	150	33044 → 2082 [PSH, ACK] Seq=1 Ack=1 Win=65467 Len=88[Packet size limited during capt…
14	0.000588358	56.47.149.224	70.191.27.130	TCP	1442	80 → 1045 [ACK] Seq=1 Ack=1 Win=65151 Len=1380[Packet size limited during capture]
15	0.000616491	172.141.82.27	114.150.100.71	TCP	64	3776 → 9339 [ACK] Seq=1 Ack=1 Win=64240 [TCP CHECKSUM INCORRECT] Len=0
16	0.000652492	56.47.149.224	70.191.27.130	TCP	1442	80 → 1045 [ACK] Seq=1381 Ack=1 Win=65151 Len=1380[Packet size limited during capture]
17	0.000677228	112.92.17.129	172.141.77.248	TCP	68	22 → 2650 [ACK] Seq=1 Ack=1 Win=65535 [TCP CHECKSUM INCORRECT] Len=0[Packet size lim…
18	0.000789166	102.44.240.3	172.141.89.63	TCP	750	60002 → 22611 [PSH, ACK] Seq=1 Ack=1 Win=64713 Len=688[Packet size limited during ca…
19	0.000950873	122.247.59.120	70.191.24.243	TCP	1514	27340 → 58821 [ACK] Seq=1 Ack=1 Win=64769 Len=1452[Packet size limited during captur…
20	0.000991285	70.191.24.250	122.157.149.148	TCP	74	36014 → 80 [ACK] Seq=1 Ack=1 Win=2149 [TCP CHECKSUM INCORRECT] Len=0

Fig.1 Information of Network Packet

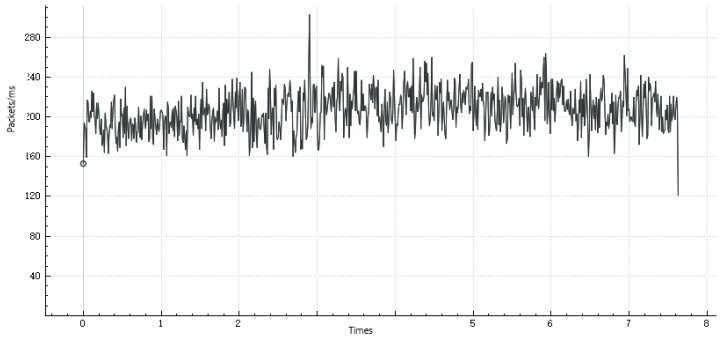

Fig.2 The traffic of packets per 10ms

4.2 The measure of the classifier

We use the Precision、Recall and Receiver Operating Characteristic(ROC) to measure the performance of our algorithm. According to outputs of the confusion matrix(showed as Table 2), Precision and Recall can be defined as:

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{3}$$

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{4}$$

Table 2：The definition of confusion matrix

Real situation	Predict situation	
	Positive	Negative
Positive	TP	FN
Negative	FP	TN

The ROC curve is useful in studying the dependency of sensibility and specificity on a particular classification method, as well as in concept drift. The ROC curve remains constant as the data changes over time. The closer the curve to the upper left corner, the greater the performance. Range of area under the ROC curve also known as area under the curve(AUC) lies om between 0 to 1.

4.3 The experiment of TscanCluster model

4.3.1 Parameters of TscanCluster model

The input parameters of *TscanCluster* model are discussed in the following:

1)For ε, if it is too large, it may fuse different clusters. If it is too small, it produce lots of clusters and requires a corresponding smaller μ .Thus we employ grid search[16] to confirm better ε .

2) After the setting of ε, μ can be heuristically set by the average number of data in the ε

neighborhood of each clusters. Generally, we apply to a smallest acceptable μ .

3) After the setting of ε, μ , λ, β can be selected by grid search.

Based on discussion above, we conduct simulation experiment and Fig.3 shows the precision of *TscanCluster* model with grid search for parameters. We can draw a conclusion from the experimental results that the algorithm performance better when ε=20, μ=27, β=0.95, λ=2.1and experiment with the optimal parameters to get the results showed as Fig.4.

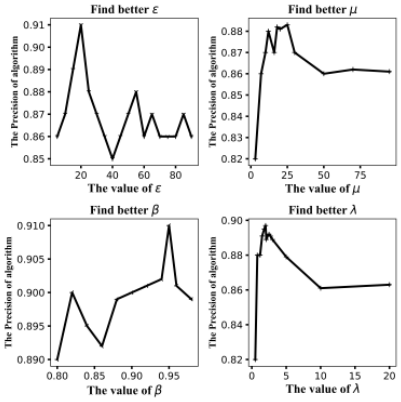

Fig.3 Grid search for parameters

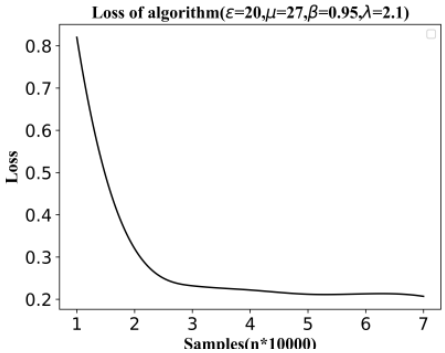

Fig.4 The loss of algorithm

4.3.2 TscanCluster model for traffic classification

We can build the *TscanCluster* model and use our algorithm to predict the samples of the new arrival with concept drift. On the other hand, we also execute Density-cluster[17]and CVFDT[6]as control groups. Fig.5 shows the Precision and Recall of the *TscanCluster* mode、Density-cluster and CVFDT. Through the contrast experiment, we can obverse that the *TscanCluster* model performance better. Further, shown as Fig.6 we can obverse that the precision rapidly drop to 0.7 when the number of samples increase about 10000 and it means the intense concept drift is appeared at this period. However the experiment results also shown that the precision quickly increase to 0.85 which demonstrates the TscanCluster model can detect concept drift and quickly recovery.

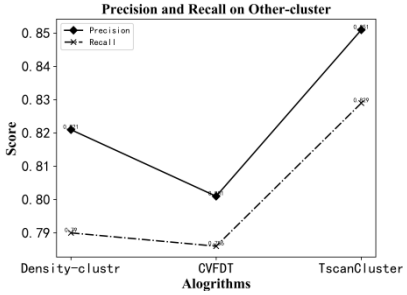

Fig.5 Precision and Recall on Other label.

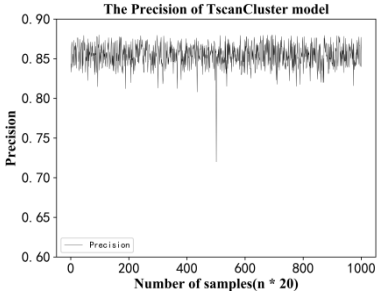

Fig.6 The detailed Precision of TscanCluster

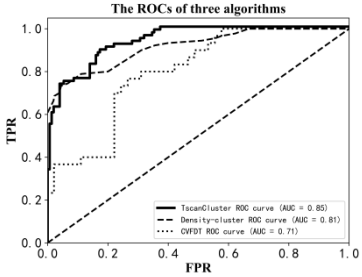

Fig.7 The ROCs of algorithms

Another experiment carry out on the concept drift and convergence time of each algorithm. The results of concept drift shows in Fig.7. By checking the AUC of each algorithm, we find the that the TscanCluster get 0.85, Density-cluster get 0.81 and CVFDT get 0.71. According the function of ROC, as a measure when datasets changes over time, we believe that the TscanCluster have better performance on datasets with concept drift or noises. However, by checking the convergence time of each algorithm, we observe that the TscanCluster takes the most time and the experimental results show as Table 3.

Table 3. The experimental result of three agorithms

	Precision	Recall	Convergence time
Density-cluster	0.83	0.82	363s
CVFDT	0.76	0.72	492s
TscanCluster	0.88	0.85	702s

5. Conclusions

In this paper, we have proposed a novel semi-supervised clustering model. The model can discover a new cluster of arbitrary shape, reduce the impact of noises and have a desirable performance on concept drift and the experiment demonstrates the effectiveness of our model. However, our proposed algorithm takes much time in experiments and how to speed up the convergence is in our future work.

Acknowledgements

This work was supported in part by the National Major Project under Grant No.2018ZX03001016, the National Natural Science Foundation of China under Grant No.61531013.

References

[1] Ji-hye Kim, Sung-Ho Yoon and Myung-Sup Kim, "Study on traffic classification taxonomy for multilateral and hierarchical traffic classification," 2012 14th Asia-Pacific Network Operations and Management Symposium (APNOMS), Seoul, 2012, pp. 1-4.

[2] J. Park, H.-R. Tyan, and K. C.-C. Jay Kuo, "GA-Based Internet Traffic Classification Technique for QoS Provisioning," in Proc of the 2006 International Conference on Intelligent Information Hiding and Multimedia Signal Processing, Pasadena, California, December 2006.

[3] 5. M. Shafiq, X. Yu, D. Wang, "Network traffic classification using machine learning algorithms", Advances in Intelligent Systems and Computing, vol. 686, pp. 621-627, 2018.

[4] H. Lim, J. Kim, J. Heo, K. Kim, Y. Hong and Y. Han, "Packet-based Network Traffic Classification Using Deep Learning," 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Okinawa, Japan, 2019, pp. 046-051.

[5] A. Liu, G. Zhang and J. Lu, "Fuzzy time windowing for gradual concept drift adaptation," 2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE), Naples, 2017, pp. 1-6.

[6] G. Liu, H. Cheng, Z. Qin, Q. Liu and C. Liu, "E-CVFDT: An improving CVFDT method for concept drift data stream," 2013 International Conference on Communications, Circuits and Systems (ICCCAS), Chengdu, 2013, pp. 315-318.

[7] S. Wang, J. Wang, X. Gao and X. Wang, "Pool-based active learning based on incremental decision tree," 2010 International Conference on Machine Learning and Cybernetics, Qingdao, 2010, pp. 274-278.

[8] R. Gopikaramanan, R. Praburaja, S. Panneerselvam and S. Dhanasekaran, "Novel Fourier Multiple orthogonal window to gauge damping and its vindication on real time info," 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM), Chennai, 2015, pp. 554-561.

[9] W. M. Jang, "Quantifying Performance in Fading Channels Using the Sampling Property of a Delta Function," in IEEE Communications Letters, vol. 15, no. 3, pp. 266-268, March 2011.

[10] Yan G , Ai M . A Framework For Concept Drifting P2P Traffic Identification[J]. Telkomnika Indonesian Journal of Electrical Engineering, 2013, 11(8):1313-21.
[11] M. Chen, X. Gao and H. Li, "Parallel DBSCAN with Priority R-tree," 2010 2nd IEEE International Conference on Information Management and Engineering, Chengdu, 2010, pp. 508-511.
[12] T. Auld, A. W. Moore and S. F. Gull, "Bayesian Neural Networks for Internet Traffic Classification," in IEEE Transactions on Neural Networks, vol. 18, no. 1, pp. 223-239, Jan. 2007.
[13] Ying Huang, "Research on sampling collecting and predicting for IP network traffic," Proceedings of 2011 International Conference on Computer Science and Network Technology, Harbin, 2011, pp. 1354-1357.
[14] Z. Liu and R. Wang, "Mobilegt: A system to collect mobile traffic trace and build the ground truth," 2016 26th International Telecommunication Networks and Applications Conference (ITNAC), Dunedin, 2016, pp. 142-144.
[15] Ji-hye Kim, Sung-Ho Yoon and Myung-Sup Kim, "Study on traffic classification taxonomy for multilateral and hierarchical traffic classification," 2012 14th Asia-Pacific Network Operations and Management Symposium (APNOMS), Seoul, 2012, pp. 1-4.
[16] F. Xue, D. Wei, Z. Wang, T. Li, Y. Hu and H. Huang, "Grid searching method in spherical coordinate for PD location in a substation," 2018 Condition Monitoring and Diagnosis (CMD), Perth, WA, 2018, pp. 1-5.
[17] Z. Heng and W. Jie, "Determination Method of Piecewise Linear Membership Function Based on the Interval Density Cluster," 2012 International Conference on Industrial Control and Electronics Engineering, Xi'an, 2012, pp. 1134-1137.