Short Text Classification and Clustering based Mobile Application Traffic Identification Method

Yuanhao Li, Shuhui Chen, Shuang Zhao
College of Computer, National University of Defense Technology, Changsha, Hunan 410073, China
liyuanhao@nudt.edu.cn

Abstract. With the rapid development of mobile network, mobile traffic accounts for a large proportion of network traffic nowadays, and mobile application traffic identification is becoming increasingly important in network security. For mobile application traffic identification, recent works have focused on proposing supervised classifiers that have shown promising performance. However, it is difficult to obtain labeled traffic in practice and most of the traffic is unlabeled. In this paper, we propose a semi-supervised mobile application traffic identification method based on short text classification and clustering, which requires only a small number of labeled samples to classify traffic. First, plain text in mobile traffic is regarded as short text and its features are extracted using a short text classification algorithm. Then K-Means++ is used to cluster the samples and give the predictions. Experimental results show that the proposed method is more effective than the semi-supervised traffic identification methods that use statistical features.

1. Introduction
With the development of mobile network, mobile devices and apps increase greatly. The Google Play Store had around 2.9 million available apps in March 2020 and an average of 6,140 mobile apps were released through the Google Play store every day according to Statista [1-2]. Besides, the report [3] stated that the number of apps monitored in Chinese local market reached 5.25 million, surpassing the number of Chinese websites. All of the above show that fixed workstation services have been fully migrated to mobile applications. Hence, mobile application traffic identification has become increasingly important.

Mobile traffic identification plays a crucial role in network security and it has been widely used for network monitoring and management, and other technologies [4-5]. For example, network management systems utilize network traffic identification technology to understand traffic distribution, users can be profiled by analyzing user mobile traffic, etc.

Based on port numbers, the traffic can be effectively classified in earlier years. However, with the appearance of dynamic port technology and unknown protocols, this method basically useless. Nguyen et al. [6] pointed out that it is impossible to classify network packets according to port numbers. The payload-based traffic identification method, i.e., DPI (deep packet inspection), can achieve high identification accuracy. Hence current traffic identification methods are mainly based on DPI technology. The disadvantage of DPI is that it cannot be worked in encrypted traffic identification. In addition, DPI technology has shortcomings such as high computational complexity and violation of user privacy [7]. Traffic identification method based on machine learning has made great progress and has shown high accuracy and strong scalability in recent years. Instead of analyzing the payload of the
network traffic, machine learning-based traffic identification methods only use the statistical features of the traffic, such as the number of packets, packet length, etc.

However, mobile application traffic has brought new challenges to current technologies. First of all, DPI method needs a large number of labeled samples to build a feature library. Compared with unsupervised or semi-supervised learning methods, supervised learning methods have higher identification performance, while they also require a large number of labeled samples for learning. In practice, the dataset is usually unlabeled or only a small part of the traffic has labels. For semi-supervised or unsupervised learning methods, related work uses only the statistical characteristics of the traffic to make the method capable of handling encrypted traffic [8-9]. However, given the new characteristics of mobile application traffic, existing methods are not entirely suitable for this scenario. First, mobile traffic is almost transmitted using HTTP/HTTPS protocol and hence application traffic is prone to have similar statistics features. Next, the unencrypted traffic still accounts for a large proportion of mobile application traffic, which contains a wealth of information for identification. In addition to the above two methods, host behavior-based method [10] can be used to identify protocols or services. However, application-level fine-grained identification cannot be achieved through it.

In view of the above deficiencies, we propose a semi-supervised mobile application traffic identification method based on short text classification and clustering in this paper. First, the URLs in HTTP traffic are extracted and further processed to get the bag-of-words space. Then, a short text clustering model is utilized to construct the feature vector of samples. Finally, the samples are clustered by a clustering algorithm. Based on the "simple majority" criterion, the unlabeled traffic in the cluster will be labeled according to a small amount of labeled traffic. To verify the effectiveness of the proposed method, a mobile traffic dataset is collected locally. This dataset contains traffic from more than 91 mobile applications installed on 10 mobile devices. Experimental results show that the proposed method can achieve an accuracy of more than 70%, which is better than the existing semi-supervised traffic identification methods.

The rest of the paper is organized as follows. Section 2 surveys related work; section 3 proposes the method; section 4 gives the experimental evaluation; section 5 gives a brief discussion; section 6 concludes the paper.

2. Related work

Since this paper focuses on the semi-supervised identification method, the related work will be introduced from the aspects of semi-supervised and unsupervised traffic identification.

2.1. Semi-supervised traffic identification

Erman et al. [11] proposed a semi-supervised traffic classification method using traffic statistical features and K-Means algorithm. The work claimed that this is the first work that uses semi-supervised learning techniques for traffic classification. Furthermore, Wang et al. [12] used the method proposed by Erman et al. [11] to construct a semi-supervised binary classification model and verified it by datasets with unknown traffic. Similarly, Zhang et al. [13] extended the method of Erman et al [11] in order to deal with the unknown traffic. Experiments show the proposed method outperforms the existing methods. Glennan et al. [14] made further improvements based on work [13] by mapping clusters to classes to locate targets that were previously difficult to classify and the accuracy had increased by 17%. Then Zhang et al. [15] proposed a robust semi-supervised traffic classification system based on their previous work [13]. The experiment results show their work is greater than state-of-the-art when zero-day applications are present. Lin et al. [16] proposed an unknown network protocol classification method based on semi-supervised learning. Li et al. [17] proposed a semi-supervised traffic classification method based on K-Means and k-nearest neighbor algorithms. The results show that their method can achieve better performance when compared with C4.5 and Native Bayes. Chen et al. [18] proposed a semi-supervised learning method that combines identification space technology with data gravity theory. Experiments show that the highest purity of the cluster exceeds 94%. Iliyasu et al. [19] applied Deep Convolutional Generation Adversarial Networks (DCGAN) to
classify encrypted traffic and verified their method on two datasets. When datasets containing a small number of labeled samples (10% of the dataset), the accuracy was 89% and 78% respectively.

2.2. Unsupervised traffic identification
Erman et al. [20] used three unsupervised algorithms to classify traffic. The experiments show that the accuracy of the AutoClass and the K-Means exceeds 85%. Zhang et al. [21] proposed a novel method, which can classify the traffic according to the application that generates the traffic. Experimental results show the effectiveness and robustness of the proposed method. Luo et al. [22] proposed an unsupervised method based on the affinity propagation mechanism. Experiments proved that the proposed method is greater than EM and K-means clustering algorithms. Alizadeh et al. [23] applied Gaussian mixture model (GMM) to generate a model for each application based on flow statistical features. Their model can be applied not only to the traffic classification, but also to the abnormal traffic detection. Hochst et al. [24] proposed an autoencoders based unsupervised traffic classification method using flow statistical features and clustering. Experimental results show that the detection speed of the method is faster. Its average precision and recall can achieve 80% and 75% respectively.

3. Methodology
In this paper, HTTP bidirectional flow defined by the 5-tuple \(<\text{SrcIP}, \text{DstIP}, \text{SrcPort}, \text{DstPort}, \text{Protocol}>\) is used as the basic classification units. The overall process of the proposed method is shown in Fig. 1. First, URLs are extracted from the request packets in HTTP flows and are segmented to construct the bag-of-words. Based on the BTM model which is a topic model used for short text clustering, the feature vectors of URLs are generated. BTM model can automatically generate keywords in each topic and give the probability that each new short text belongs to each topic after specifying the number of topics. Finally, K-Means++ with JS divergence distance is utilized to divide samples into clusters. Within each cluster, the "simple majority" principle is used to label unlabeled samples.

3.1. Preprocessing
After analyzing the mobile application traffic, we found that URLs are closely related to the use of mobile applications. URLs generated by the same app under the same function will have extremely high similarity. In contrast, URLs generated by different applications under different functions may show obvious differences. As shown in Fig. 2(a), there are 5 URLs from WeChat, of which the first three are from short message functions, the fourth is from payment function, and the fifth is from...
WeChat applet. Obviously, the URLs from the short message have a similar format but are different from the other two URLs. In addition, as shown in Fig.2(b), there are 4 URLs, of which the first two are from WeChat, and the last two are from UC browser and MOMO respectively (UC browser is a browser app and MOMO is an app for chatting and making friends). Obviously, the similarity of URLs from WeChat is significantly higher than the URLs from different applications. Therefore, the URL is not a string of random characters. It usually has a certain meaning and can be regarded as a meaningful short text.

Since URL appears in unencrypted flows, only HTTP flows are used in this method. One HTTP flow usually includes request messages and response messages. The request message consists of the request method, URL, and the header. The header contains lots of information, such as the type of the resource requested, page encoding type, HTTP connection status, etc. The response message contains the status information and the response body. The status information includes the HTTP version and a response code. The response entity contains information about the response server, such as the server software version, connection status, etc. However, the server-side information is not suitable for distinguishing different applications. The main reason is that CDN (Content Distribution Network) and cloud technology are widely used in mobile applications so that one server can provide services for different mobile applications at the same time. Therefore, URLs are selected and are regarded as a short text as the source data to classify mobile traffic.

URLs will be pre-processed as follows. First, the resource path is extracted from the HTTP request packet and the host field of the header is spliced with it to form a complete URL. Next, the URL is segmented to multiple words using "/", "?", "&", "=" and "". Since random numbers composed of pure numeric characters and words composed of pure alphabetic characters are not highly effective for clustering, these words are filtered. Besides, words whose character length is less than or equal to a preset threshold would be removed. The final obtained bag-of-words will be used in the BTM model to construct the feature vector of URLs. The preprocessing procedure is shown in Algorithm 1.

**Algorithm 1: URLs preprocessing**

**Input:** URLs

**Output:** Bag of words

1. for url in URLs do
2.    bow ← url.split("/" and "?" and "&" and "=" and ")
3.    for word in bow do
4.      if word.length ≤ Threshold then
5.        remove word from bow
6.      else if word.isdigit == true then
7.        remove word from bow
8.      else if word.isalpha == true then
9.        remove word from bow
10. end if
11. return bow
3.2. Feature extraction

Traditional text clustering models are typically used for document or long text. Different from long text clustering, due to the short length of the URL, a URL would contain few words after preprocessing. However, the bag-of-words space formed by all the URLs will be large. Therefore, the traditional text clustering models cannot be applied in our method since it will cause the feature explosion, such as one-hot, BOW [25], and continuous bag-of-words model (CBOW) [26].

In fact, all short text clustering would encounter the above problem when using conventional text clustering models. Hence BTM (Biterm Topic Model) proposed by Yan et al. [27] is chosen in this paper as the text clustering method. BTM is a topic model based on "biterm". After specifying the number of topics, BTM will generate the keywords in each topic and the probability of a new text belonging to each topic. Compared with the traditional text clustering, which uses a single word for feature extraction, BTM can model the word co-occurrence patterns.

BTM model calculates parameters based on Gibbs sampling. First, the joint probability is calculated through the Dir-Multi conjugate distribution. Then the conditional probability is obtained based on the joint probability. Finally, Gibbs sampling is performed. The conditional probability distribution $P(z|z_b, B, \alpha, \beta)$ for each biterm $b = (w_i, w_j)$ is computed as follows:

$$P(z|z_b, B, \alpha, \beta) \propto \left( \frac{n_{w|z} + \beta}{\sum_{w} n_{w|z} + M \beta} \right)^z$$

where $z_b$ denotes the topic assignments for all biters except $b$, $B$ is the global biterm set, $n_z$ is the number of times of the biterm $b$ assigned to the topic $z$, and $n_{w|z}$ is the number of times of the word $w$ assigned to the topic $z$.

The following formulas are used by Gibbs sampling for updating parameters.

$$\phi_{w|z} = \frac{n_{w|z} + \beta}{\sum_{w} n_{w|z} + M \beta}, \quad (2)$$

$$\theta_z = \frac{n_z + \alpha}{|B| + K \alpha}, \quad (3)$$

where $|B|$ is the total number of biters.

For the URL in Fig.2(a), the process of the feature extraction is illustrated in Fig.3. 5 WeChat URLs are segmented and are inputted into BTM model. Afterward, the probability that each URL belongs to each topic is outputted.

![Fig.3: The process example of the BTM model](image)

3.3. K-Means++ clustering

K-Means is a widely used clustering method and its basic idea is to classify similar samples into same cluster and different samples are divided into different clusters. Compared with other clustering methods, K-Means is more efficient in processing large datasets. The number of clusters $K$ in K-Means needs to be set in advance and it is usually determined by experience. However, the choice of the initial clustering center will greatly affect the clustering result. The widely used random selection is prone to cause a locally optimal solution. K-Means++ is an improved version of K-Means which introduces a new way to initial clustering centers. The initial clustering center selection of K-Means++
is as follows. First, a point is randomly picked as the first cluster center. Then the point with the largest distance in the shortest distance from the selected cluster center is chosen each time as the new cluster center until K cluster centers are decided. Since K-Means++ can solve the initial clustering center selection to some extent, it is used in this paper as the clustering algorithm.

After extracting the feature vectors of URLs through BTM, the calculation of the similarity between two URLs translates into the calculation of the similarity between the corresponding feature vectors. Since the features obtained from topic model are presented in the form of probabilities, KL divergence (Kullback-Leibler Divergence) is generally used in this case. The symmetric version of the KL divergence is called JS divergence (Jensen-Shannon Divergence), which is used in this paper and its calculation is as follows:

\[ D_{JS}(d_i, d_j) = \frac{1}{2} D_{KL}(d_i||m) + \frac{1}{2} D_{KL}(d_j||m) \]  

where \( m = \frac{1}{2} (d_i + d_j) \), and \( D_{KL}(p||q) = \sum_i p_i \ln \frac{p_i}{q_i} \) is the Kullback–Leibler divergence.

Finally, the principle of "simple majority" is used to label the unlabeled samples in the clusters.

4. Evaluation

4.1. Dataset collection

Since mobile traffic contains a lot of private data, there is no public mobile traffic dataset available currently. Therefore a local dataset is collected and the data collection method proposed by Zhao et al. [28] is used. We collected mobile traffic generated by 10 different mobile devices for more than two months. Involved mobile devices come from different manufacturers, including Xiaomi, Huawei, and Samsung. The collected dataset contains more than 40,000 flows from 91 applications.

Then necessary preprocess is performed on the dataset. First, flows not containing HTTP request messages are removed from the dataset. Then, flow samples have only one word after segmentation and filtering is removed. Finally, 8264 flows from more than 60 applications are picked out as shown in Table 1, which is the final dataset we used in our experiment.

| Table 1. Overview of the Dataset |
|----------------------------------|
| Time of data collection | TCP flow size | Number of apps |
| Candidate Dataset | 2019.11.15-2020.01.17 | 47469 | 91 |
| Final Dataset | 2019.11.15-2020.01.17 | 8264 | 65 |

4.2. Evaluation Metrics

Three metrics, including precision, recall, and accuracy is used in this paper to evaluate the proposed method. Accuracy is the ratio of the number of correctly classified samples to the total number of samples. Recall and precision are used to evaluate the identification results for each class. For class C, TP refers to the number of samples correctly classified as C, FP refers to the number of samples incorrectly classified as C. FN refers to the number of samples incorrectly classified as non-C. The calculation methods of precision and recall are as follow:

\[ \text{precision} = \frac{TP}{TP + FP}, \quad \text{recall} = \frac{TP}{TP + FN} \]

Since the dataset contains dozens of applications, the average of the precision and the recall will be shown instead of individual value. It is worth mentioning that TP, FN, and FP for a class maybe 0 in experiments. If TP is 0, the precision and recall would be 0. Similarly, if TP and FP are both 0, the accuracy and recall cannot be calculated. If the mentioned cases happen, these invalid values would be ignored and only the remaining values will participate in the calculation.

4.3. Influence of the proportion of labeled samples

In order to verify the influence of the proportion of labeled samples on the identification performance of the proposed method, a certain proportion of the samples of each application are randomly selected for labeling. The number of clusters is set to 60 considering that at least one application should be
clustered into one cluster. The proportion of labeled samples is set between 5%-50% and the step size is 5%. For different parameters, the test repeat 10 times and the average results are shown. The results of the experiment are given in Table 2.

Table 2. Average precision (%), average recall (%) and accuracy under different random sampling proportion

| Proportion | Precision | Recall | Accuracy |
|------------|-----------|--------|----------|
| 5%         | 52.26     | 57.44  | 63.50    |
| 10%        | 56.35     | 61.83  | 66.11    |
| 15%        | 55.76     | 62.02  | 66.64    |
| 20%        | 56.67     | 62.39  | 66.81    |
| 25%        | 57.76     | 62.78  | 67.10    |
| 30%        | 57.06     | 62.45  | 67.06    |
| 35%        | 58.24     | 62.66  | 67.25    |
| 40%        | 57.56     | 62.39  | 67.13    |
| 45%        | 58.25     | 62.37  | 67.01    |
| 50%        | 58.31     | 63.40  | 67.48    |

Table 3. Average precision (%), average recall(%) and accuracy at different K

| K  | Precision | No. Of class | Recall | No. of class | Accuracy |
|----|-----------|--------------|--------|--------------|----------|
| 30 | 53.46     | 14.5         | 61.21  | 14.5         | 54.97    |
| 40 | 61.68     | 19.3         | 63.59  | 19.3         | 60.31    |
| 50 | 58.38     | 22.8         | 63.10  | 22.8         | 63.20    |
| 60 | 56.67     | 26.8         | 62.39  | 26.8         | 66.81    |
| 70 | 58.35     | 29.2         | 62.19  | 29.2         | 69.55    |
| 80 | 58.84     | 29.7         | 60.69  | 29.7         | 69.58    |
| 90 | 59.64     | 30.8         | 59.49  | 30.8         | 70.30    |
| 100| 59.85     | 30.8         | 59.88  | 30.8         | 70.27    |
| 110| 59.97     | 31.5         | 59.06  | 31.5         | 70.71    |
| 120| 62.20     | 32.2         | 58.35  | 32.2         | 70.84    |

It can be seen that the accuracy, precision, and recall all show a slight upward trend with the increase of the proportion of labeled samples. This result makes sense since the higher the proportion of labeled samples the more information is known. In general, more information in the dataset is known, more accurate the analysis can be. Besides, the "simple majority" principle is used in the label stage, which is beneficial to the class with more labeled samples.

4.4. Influence of the different K

In this section, the proportion of labeled samples is fixed to 20%. Then different values are set for the initial cluster number to verify the impact of different K on the algorithm performance. The range of K is 30 to 120 and the step size is 10. For each K, the test is repeated 10 times. The experimental results are shown in Table 3. The column ‘No. of class’ represents the number of apps involved in the calculation. The larger of the ‘No. of class’, the better the calculated metric can represent the entire dataset.

Experimental results show that the overall trend of average precision is slightly upward with the increase of K. However, when K is 40 and 50, the precision shows a relatively large increase. One reason is that the number of classes involved in the calculation is small which results in classes with high precision affecting the average class precision. Similarly, the recall has the same tendency. In general, precision and accuracy increase due to the increase of the number of clusters but the recall
decreases after K=100, hence the better cluster number for the dataset is 90 and 100. Besides, the larger K would result in a longer running time of the algorithm, hence 90 is selected as the best cluster number.

4.5. Comparison with other methods

At present, there is no semi-supervised or unsupervised method for mobile application traffic identification. Therefore, Erman [11] and Li [19] is chosen to compare with our methods since they use k-means to classify traffic. They both use statistical features to classify the traffic. Erman's method uses 11 flow features and Li's method uses 10 flow features. The difference between them is that Erman and Li use "simple majority" and kNN respectively when labeling unlabeled samples.

With the labeled sample proportion fixes to 20%, three methods are compared. Fig.4 shows the precision, recall and accuracy of the three methods on different K respectively.

The experimental results show that our method is better than the other two methods and Li's method is better than Erman's method. Furthermore, the results indicate that the statistical traffic features are more suitable for the classification of the protocol rather than the classification of the mobile application, which verifies our hypothesis. Since the traffic samples used in our paper belong to HTTP, the compared two methods are inferior to the proposed method. In addition, Li's method does not use the "simple majority" method when labeling unlabeled traffic but uses the kNN. Although the way of "simple majority" mapping is simple, it will label all the unlabeled traffic in the cluster with the same label which results in the classification will be worse than the result obtained by kNN. Finally, statistical features selected by Erman's method are relatively simple features such as the number of packets and the number of bytes, and it is difficult to reflect the features of different application’s traffic. In contrast, Li's features include relatively complex statistical features that can better reflect the features of different application flows, such as RTT samples and initial window size, etc.

5. Discussion

Compared with the method that discards the text features in the traffic and only uses the statistical features of the traffic, this method makes good use of the text features in the traffic, especially for mobile application traffic. The experiments further verify the effectiveness of the proposed method. In addition, there are a few problems we encountered in this work worth discussing.

Firstly, the same URL may appear in the traffic generated by different applications which could result from two cases. First, one mobile app can call the services of other applications. For example, when clicking an eBay payment link in Twitter, the generated eBay traffic would be labeled as Twitter since the data collection method only cares about which foreground app generates the traffic. Second, same third-party services can be called by multiple mobile apps. In the latter case, the traffic from the same third-party services can be gathered and be identified as a new class. For the former case, DPI method and other preprocessing are used to filter the traffic in advance.

Secondly, when the traffic is completely unlabeled, our method can be migrated to unsupervised learning algorithm easily after a few modifications. In short, for each cluster generated by clustering algorithm, common substrings and other related information can be extracted from the cluster. In
addition, the relevant keyword fields of apps can be extracted from its installation package. After matching the two types of information, the cluster may be labeled automatically without any labeled sample.

6. Conclusion
In this paper, we propose a semi-supervised mobile application traffic identification method based on short text classification and clustering. The method uses the idea of text clustering to identify the traffic generated by different mobile applications. The experiments show that the proposed method is much better than the traditional method that only uses statistical features. Selecting more distinctive text features is the direction of further optimization of this method. How to improve the classification accuracy and realize the automatic classification of traffic without labeled samples is the research direction of our future work.

Acknowledgments
This work is supported by the National Key R&D Program of China under Grants 2017YFB0802300

References
[1] J. Clement. Number of available applications in the Google Play Store from December 2009 to June 2020 [DB/OL]. https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/.
[2] J. Clement. Average number of new Android app releases per day from 3rd quarter 2016 to 1st quarter 2018[DB/OL]. https://www.statista.com/statistics/276703/android-app-releases-worldwide/.
[3] CAICT. White Paper of App Data Security and Personal Information Protection [EB/OL]. https://www.cebnet.com.cn/upload/resources/file/2019/12/30/78787.pdf (in chinese).
[4] Cai Z , Wang Z , et al. A Distributed TCAM Coprocessor Architecture for Integrated Longest Prefix Matching, Policy Filtering, and Content Filtering[J]. IEEE Transactions on Computers, 2013, 62(3):417-427.
[5] Yu Y, Long J, et al. Network Intrusion Detection through Stacking Dilated Convolutional Autoencoders[J]. Security and Communication Networks, 2017:1-10
[6] Nguyen T T T , Armitage G . A survey of techniques for internet traffic classification using machine learning[J]. IEEE Communications Surveys & Tutorials, 2009, 10(4):56-76.
[7] Cherry, S. The VoIP Backlash[J]. IEEE Spectrum, 2005, 42(10):61-63.
[8] Aceto G , Cuionzo D , et al. Toward Effective Mobile Encrypted Traffic Classification through Deep Learning[J]. Neurocomputing, 2020.
[9] Zhongsheng W , Jianguo W , et al. Traffic identification and traffic analysis based on support vector machine[J]. Concurrency and Computation: Practice and Experience, 2020, 32(2).
[10] M. Illofoitou, P. Pappu, M. Faloutsos, et al. Graption: Automated detection of P2P applications using traffic dispersion graphs (TDGs). In UC Riverside Technical Report, CS-2008-06080, 2008.
[11] Erman J , Mahanti A , et al. Offline/realtime traffic classification using semi-supervised learning[J]. Performance Evaluation, 2007, 64(9-12):1194-1213.
[12] Wang Y , Chen C , et al. Unknown pattern extraction for statistical network protocol identification[C]// 2015 IEEE 40th Conference on Local Computer Networks (LCN 2015). IEEE, 2015.
[13] Zhang J , Chen C , et al. An Effective Network Traffic Classification Method with Unknown Flow Detection[J]. Network & Service Management IEEE Transactions on, 2013, 10(2):133-147.
[14] Glennan T, Leckie C, et al. Improved Classification of Known and Unknown Network Traffic Flows Using Semi-supervised Machine Learning[C]. australasian conference on information security and privacy, 2016: 493-501.
[15] Zhang J, Chen X, et al. Robust Network Traffic Classification[J]. IEEE/ACM Transactions on Networking, 2014, 23(4):1-1.

[16] Lin R, Li O, et al. Unknown network protocol classification method based on semi-supervised learning[C]// 2015 IEEE International Conference on Computer and Communications (ICCC). IEEE, 2015.

[17] Li Lin-lin, ZHANG Xiao-yi, et al. Semi-Supervised Traffic Identification Based on K-Means and k-Nearest Neighbours[J]. Journal of Information Engineering University, 2015(02):110-115(in Chinese).

[18] Chen Z, Liu Z, et al. A novel semi-supervised learning method for Internet application identification[J]. Soft Computing, 2017, 21(8):1963-1975.

[19] Iliyasu A S, Deng H. Semi-Supervised Encrypted Traffic Classification With Deep Convolutional Generative Adversarial Networks[J]. IEEE Access, 2020: 118-126.

[20] Erman J, Arlitt M F, et al. Traffic Classification Using Clustering Algorithms[C]// Proceedings of the 2nd Annual ACM Workshop on Mining Network Data, MineNet 2006, Pisa, Italy, September 15, 2006. ACM, 2006.

[21] Jun Zhang, Yang Xiang, et al. Unsupervised traffic classification using flow statistical properties and IP packet payload[J]. Journal of Computer and System Sciences, 2013, 79(5).

[22] Jian-zhen LUO. Unsupervised Traffic Classification Using Affinity-Propagation-Based Clustering Method[C]. Advanced Science and Industry Research Center.Proceedings of 2014 International Conference on Computer,Network Security and Communication Engineering(CNSCE 2014).Advanced Science and Industry Research Center:Science and Engineering Research Center,2014:422-426.

[23] Alizadeh H, Khoshrou A, et al. Traffic Classification and Verification using Unsupervised Learning of Gaussian Mixture Models[C]// IEEE International Workshop on M&n Measurement & Networking. IEEE, 2015.

[24] Hochst J, Baumgartner L, et al. Unsupervised Traffic Flow Classification Using a Neural Autoencoder[C]. local computer networks, 2017: 523-526.

[25] Zhang Y, Jin R, et al. Understanding bag-of-words model: a statistical framework[J]. International Journal of Machine Learning & Cybernetics, 2010, 1(1-4):43-52.

[26] Mikolov T, Le Q V, et al. Exploiting Similarities among Languages for Machine Translation[J]. arXiv: Computation and Language, 2013.

[27] Yan x, Guo J, et al. A biterm topic model for short texts[C]//Proceedings of the 22nd international conference on World Wide Web. International World Wide Web Conferences Steering Committee, 2013: 1445-1456.

[28] Zhao Shuang, Chen Shuhui, et al. Identifying Known and Unknown Mobile Application Traffic Using a Multilevel Classifier[J]. Security & Communication Networks, 2019, 2019:1-11.