Towards Unknown Traffic Identification Using Deep Auto-Encoder and Constrained Clustering

Yongzheng Zhang[1,2], Shuyuan Zhao[1,2], and Yafei Sang[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences
[2] School of Cyber Security, University of Chinese Academy of Sciences
Beijing, China
{zhangyongzheng,zhaoshuyuan,sangyafei}@iie.ac.cn

Abstract. Nowadays, network traffic identification, as the fundamental technique in the field of cybersecurity, suffers from a critical problem, namely "unknown traffic". The unknown traffic refers to network traffic generated by previously unknown applications (*i.e.*, zero-day applications) in a pre-constructed traffic classification system. The ability to divide the mixed unknown traffic into multiple clusters, each of which contains only one application traffic as far as possible, is the key to solve this problem. In this paper, we propose the *DePCK* to improve the clustering purity. There are two main innovations in our framework: *(i)* It learns to extract bottleneck features via deep auto-encoder from traffic statistical characteristics; *(ii)* It uses the flow correlation to guide the process of pairwise constrained k-means. To verify the effectiveness of our framework, we make contrast experiments on two real-world datasets. The experimental results show that the clustering purity rate of DePCK can exceed 94.81% on the ISP-data and 91.48% on the WIDE-data [1], which outperform the state-of-the-art methods: RTC [20], and k-means with log data [15].

Keywords: Unknown traffic · Deep auto-encoder · Bottleneck features · Pairwise constrained k-means.

1 Introduction

The performance of network traffic identification directly affects network security and controllability, because it is a basic tool for network management tasks such as network monitoring, quality of service, traffic priority [20]. With the explosion of network applications, network traffic identification suffers from a critical problem, namely "unknown traffic". The unknown traffic is defined as network traffic generated by previously unknown applications (*i.e.*, zero-day applications) in a traffic classification system. The network traffic statistics of the Internet2 organization to the North American backbone network shows that nearly 50% of the traffic belongs to unknown traffic [16].

The methods of fine-grained unknown traffic identification can be generally divided into three stages. First, extracting mixed unknown traffic from raw network traffic (including known traffic, and unknown traffic) [10] [20] [21]. Then,

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_23

dividing the mixed unknown traffic into multiple clusters, each of which contains only one application traffic as far as possible [8] [15] [20]. Finally, identifying clusters through manually labeling [20] or association information (*e.g.*, DNS). To solve this problem, machine learning methods based on typical flow-statistical-features (*e.g.*, packet size, packet-interval) have been widely applied in unknown traffic identification, but most of them are aimed at solving the key problems of the first stage [7] [9] [10] [14] [21]. Previous research of the second stage has the following shortcomings: (*i*) Previous studies cannot perform beneficial feature selection just using unlabeled dataset [8] [15] [20]. (*ii*) Flow correlation is not entirely utilized to guide the clustering method [8] [15] [20]. All these issues will reduce the accuracy of clustering and affect the efficiency of unknown traffic identification.

In this paper, an unsupervised framework, which we call *DePCK*, is proposed to improve the dividing power of mixed unknown traffic (focusing on the second stage). To achieve traffic information embeddings without labels, it uses deep auto-encoder to build a self-supervised feature extraction model. To improve clustering performance, it fully uses flow correlation to guide the process of pairwise constrained clustering.

The major contributions can be summarized as follows:

- We propose the *DePCK*, an unsupervised framework for unknown traffic identification problem.
- We first use the bottleneck features (by mean of deep auto-encoder) to model unknown traffic.
- We use flow correlation (*i.e.*, 3-tuple of flow) to guide pairwise constrained clustering.
- The experiments of *DePCK* on two real-world datasets: ISP, and WIDE [1], show that the clustering purity rate of DePCK can exceed 94.81% on the ISP-data and 91.48% on the WIDE-data, which outperform the state-of-the-art methods: RTC [20], and k-means with log data [15].

The rest of this paper is structured as follows. A novel framework for network unknown traffic identification is proposed in Section 2, Section 3 and Section 4. Section 5 describes the datasets and evaluation metrics. Section 6 reports a large number of experiments and experimental results. Section 7 discusses related work in unknown traffic identification. Finally, Section 8 concludes the paper.

2 The *DePCK* Framework

Fig. 1 provides the details of the *DePCK*. This framework includes two main modules: features extraction module and clustering module. In the features extraction module, according to the demonstrated capabilities [6] of feature learning and the theoretical function approximation properties [11] of deep neural networks (*DNNs*), we use a deep auto-encoder to train a self-supervised deep neural network and learn bottleneck features from unlabeled samples. This part is described in detail in Section 3. In the clustering module, we first extract the constrained relation between traffic flow and then use the *MPCKMeans* algorithm to match the unknown traffic identification scenario. The training data of this module is the bottleneck features of the features extraction module. This part is described in detail in Section 4.

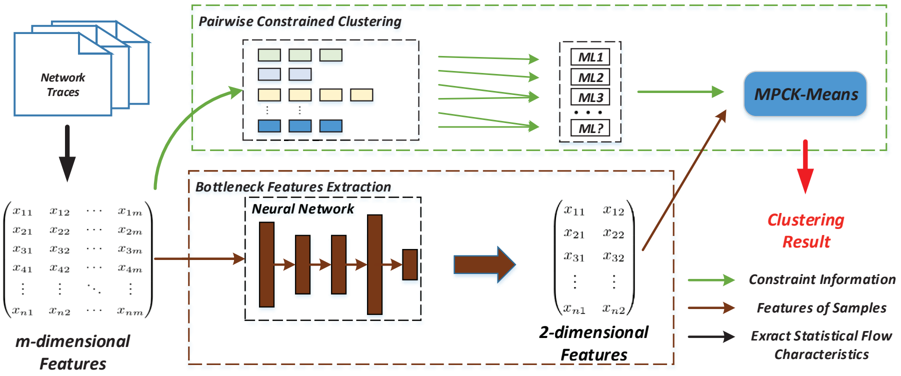


Fig. 1: The *DePCK* Framework

3 Bottleneck Features Extraction

In this section, we describe the feature extraction module of *DePCK* based on deep auto-encoder, which can automatically train an unsupervised deep neural network and obtain the bottleneck features of the samples.

3.1 Deep Embedding

Network traffic classification schemes based on flow statistics generally train a classification model from a set of labeled data, which is composed of multiple statistical characteristics (*e.g.*, packet size, packet-interval) and class labels. Based on labels, most schemes usually first use supervised feature selection methods (*e.g.*, correlation coefficient, and covariance) to remove redundant and unrelated features.

Since unknown traffic has no labels, supervised feature selection methods are invalid to solve the unknown traffic identification problem. To tackle this problem, we use a neural network, which has demonstrated feature learning capabilities [18], to transform the feature with non-linear mapping. The non-linear transformation of features is to map the feature *space* from X to Z:

$$f_\theta : X \to Z \tag{1}$$

where Z is the latent *feature space*, and its dimensionality is smaller than the *space* X, and θ are parameters that can be automatically learned based on a deep neural network.

3.2 Training a Bottleneck Network

Deep neural networks have multiple hidden layers, which can train the input data through non-linear mapping and obtain hidden feature sets of samples. In our scene, without labeled data, we use an unsupervised deep auto-encoder to train the deep neural networks. Deep auto-encoder is an unsupervised neural network, which composed of multilayer auto-encoders.

An auto-encoder is a type of artificial neural network used to learn efficient data codings in an unsupervised manner [12]. The aim of an auto-encoder is to learn a representation (encoding) for a set of data. Architecturally, the form of an auto-encoder is a feedforward, non-recurrent and two-layer neural network. As shown in the green part of Fig. 2, an auto-encoder always consists of two parts, the encoder and the decoder, which can be defined as:

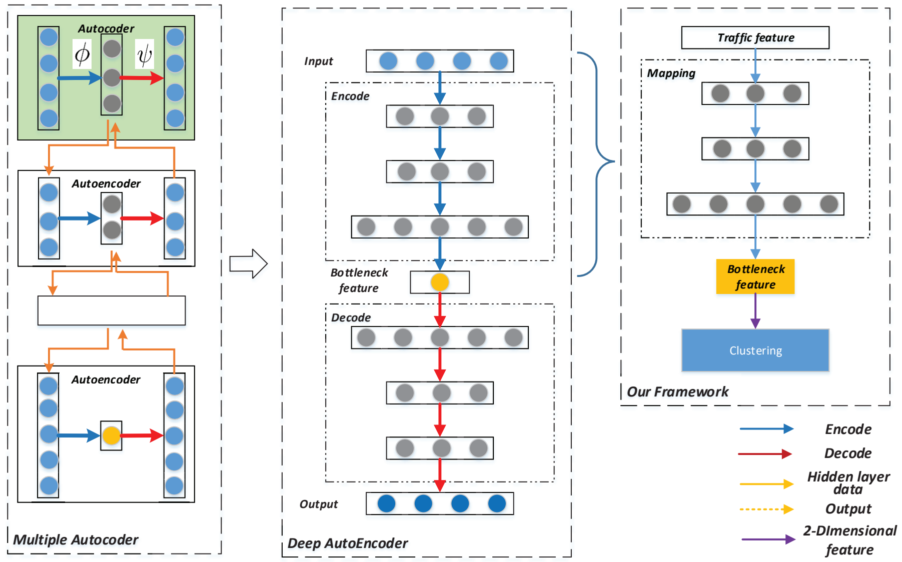


Fig. 2: Deep Auto-Encoder Structure

$$\psi : \alpha \rightarrow \beta \tag{2}$$

$$\phi : \beta \rightarrow \alpha \tag{3}$$

$$\psi, \phi = argmin_{\psi,\phi}||\alpha - (\psi \circ \phi)\alpha||^2 \tag{4}$$

where the encoder stage of an auto-encoder takes the input $x \in R^d = \alpha$ and maps it to $z \in R^P = \beta$, and the decoder stage of the auto-encoder maps z to the reconstruction x' of the same shape as x :

$$z = r_1\{W_1 x + b^1\} \tag{5}$$

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_23

$$x' = r_2\{W_2 z + b^2\} \tag{6}$$

where z is latent representation, r_1 and r_2 are element-wise activation function such as a sigmoid function or a rectified linear unit. W_1 and W_2 are weight matrix, b_1 and b_2 are bias vector. In the model, all activation functions are rectified linear units (ReLUs).

The training process of the auto-encoders is to minimize the loss function J. The loss function is defined as:

$$J(x, x') = \sum_{x \in D} L^p(x, x') \tag{7}$$

where L^p is reconstruction errors, here we use the square of Euclidean norm: $||x - y||^2$. D is the dataset.

As shown in the middle part of Fig. 2, the deep auto-encoder is a deep neural network with multiple layers. After training auto-encoders by greedy layer-wise training, we connect all the encoders in series and then combine all the decoders in the opposite direction to form a deep auto-encoder. When designing the deep network structure, we set the middle layer with the minimum dimension in all layers to build the bottleneck features, because the bottleneck features have recently found success in a variety of speech recognition tasks [18]. Then, we get the final model by discarding and use the model as the initial mapping between the $raw\ feature\ space$ and the $bottleneck\ feature\ space$.

4 Pairwise Constrained Clustering

In this section, we first describe the constrained relation between network traffic and then introduce the modified pairwise constrained clustering algorithm based on flow correlation.

4.1 Correlations between Network Traffic

In Transmission Control Protocol/Internet Protocol (TCP/IP) model, IP flow, a series of data packets transferred between two programs, is the basic unit for end-to-end data transfer. In the program, the system determines an IP flow through the IPv4 five-tuple. A five-tuple refers to a set of five different values that uniquely identifies a UDP/TCP session. It includes a source IP address/port number, destination IP address/port number and transport protocol. An Internet Protocol address (IP address) is a numerical label assigned to each device connected to a computer network that uses the Internet Protocol for communication. A port is an endpoint of communication in an operating system, which identifies a specific process or a type of network service running on that system. Hence, a port can be used with an IP address of a host and the transport layer protocol for communication. For example, to transfer a file to a remote computer, one could specify the machine itself by IP address, use TCP for transport, and the FTP file server service on that computer on port 20.

We assume that the service provided by a particular port lasts for a certain period. In this example, the flows that have the same three-tuple (service IP address, service port number, and transport protocol) can be considered to belong to the same protocol. This assumption is typically valid for the Internet because of the port-reuse restriction rule enforced by operating systems, in which a local port number will become unavailable for some time after closing unless a particular program is bound to it [17]. Therefore, we can use the 3-tuple of flows to obtain constrained dataset.

The flows' constrained relation can be used to guide the clustering process when the unknown traffic is identified based on the clustering algorithm. In the clustering process, if there is a large number of associated flows between two independent clusters, the clustering algorithm can determine that the correlation between the two clusters is strong. Based on this idea, we propose an unknown traffic identification method based on pairwise constrained clustering algorithm.

4.2 Modified PCKMeans

In previous studies, unknown traffic clustering methods based on statistical characteristics cannot make good use of flow correlation [8] [15] [20]. To make full use of the flows' constrained relation, we propose the modified pairwise constraint clustering algorithm based on PCKMeans [5].

Pairwise Constraint Conditions. PCKMeans, an improved k-means algorithm, uses the prior knowledge of the data to guide the clustering process and gets better clustering results. Consequently, in addition to the distance between samples in the data, this algorithm uses pairwise *must-link* (ML) and *cannot-link* (CL) constraints to guide clustering. ML is a set of must-link pairs and CL is a set of cannot-link pairs. if $(x_i, x_j) \in ML$, x_i and x_j should be assigned to the same cluster. Conversely, if $(x_i, x_j) \in CL$, x_i and x_j should be assigned to the different cluster.

In our *DePCK*, we can use the 3-tuple of network flows to construct the ML set but can not build the CL set.

Modified PCKMeans. The PCKMeans algorithm implements the use of ML set and CL set by adding a constraint violation penalty term to the objective function of the k-means algorithm. In the case of a given dataset D, a set of must-link constraints ML, a set of cannot-link constraints CL, the PCKMeans algorithm can minimize objective function by giving the weights corresponding to the ML and CL respectively. The objective function J of PCKMeans can be computed as:

$$J_{ML} = \sum_{(x_i, x_j) \in ML} w_{ij} I(C_i \neq C_j) \tag{8}$$

$$J_{CL} = \sum_{(x_i, x_j) \in CL} \overline{w}_{ij} I(C_i = C_j) \tag{9}$$

$$J = \frac{1}{2} \sum_{x_i \in D} ||x_i - c_i||^2 + J_{ML} + J_{CL} \tag{10}$$

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_23

where x_i and x_j are the single sample of the dataset; C_i and C_j are the assigned cluster of x_i and x_j respectively, w_{ij} and $\overline{w}_{ij}$ are two sets that give weights corresponding to the ML and CL respectively, $I(\cdot)$ is the indicator function, with $I(true) = 1$ and $I(false) = 0$. In Eq. (10), c_i is centroid of C_i.

Because there is no CL set of network traffic, we do not need to add J_{CL} to the objective function J. The objective function J of our model is defined as:

$$J = \frac{1}{2} \sum_{x_i \in D} ||x_i - c_i||^2 + J_{ML} \tag{11}$$

Algorithm 1 Modified PCKMeans

```
Input: $D = \{x_i\}_{i=1}^{n}$:set of samples; $ML = \{(x_i, x_j)\}$: set of must link samples;
       $k$: number of clusters;$w$: weight of constraints;
Output: $J_{min}$:divide the dataset into $k$ clusters and have the smallest $J$ value;
        $C = C_1, C_2, ..., C_n$:the set of clusters.
initialize the Centroids $\{c_i\}_{i=1}^{k}$ of $k$ clusters at random
repeat
    for $x_i \in D$ do
        for $C_j \in C$ do
            Calculate the objective function: $J^j = \sum_{x_i \in D} ||x_i - c_j||^2$
        end for
        assign sample $x_i$ to the cluster $j$ where $J^j$=$argmin(J^*)$
    end for
    for $c_i \in \{c_i\}_{i=1}^{k}$ do
        recalculate the Centroids $\{c_i\}_{i=1}^{k}$ of $k$ clusters: $c_i = \frac{\sum_{x_i \in D} x_i}{|C_i|}$
    end for
until none of the Centroids $\{c_i\}_{i=1}^{k}$ of $k$ clusters changes
repeat
    for $x_i \in D$ do
        for $C_j \in C$ do
            Calculate the objective function: $J_{ML} = \sum_{(x_i,x_j) \in ML} w_{ij} I(C_i \neq C_j)$
            Calculate the objective function: $J^j = \sum_{x_i \in D} ||x_i - c_i||^2 + J_{ML}$
        end for
        assign sample $x_i$ to the cluster $j$ where $J^j$=$argmin(J^*)$
    end for
    for $c_i \in \{c_i\}_{i=1}^{k}$ do
        recalculate the Centroids $\{c_i\}_{i=1}^{k}$ of $k$ clusters: $c_i = \frac{\sum_{x_i \in D} x_i}{|C_i|}$
    end for
until none of the Centroids $\{c_i\}_{i=1}^{k}$ of $k$ clusters changes
```

In the PCKMeans algorithm proposed by Basu *et al.* [5], the initialization phase strategy is designed as follows: Firstly, using the *must-link* set to construct

DOI: 10.1007/978-3-030-22734-0_23

λ neighborhood sets $\{N_p\}_{p=1}^{\lambda}$, then using the information of neighborhood sets to initialize the center centroids of k clusters as much as possible. The most significant advantage of this strategy is that the *must-link* set directly determines the distribution of the clusters. However, in the unknown traffic identification scenario, this advantage will have the opposite effect because it prevents the algorithm from discovering new traffic protocol or application from mixed traffic.

In order to solve the above problem, we propose an improved pairwise constraint clustering algorithm (MPCKMeans) Algorithm 1.

In this algorithm, we first use the k-means algorithm to complete the clustering of data D and obtain the centroid of the clustering clusters as the initial centroid of the next stage. Then, based on the result of k-means, we use the *must-link* constraints to guide the clustering process. The improved algorithm not only makes full use of the pairwise constraints, but also has excellent ability to discover unknown protocols.

5 Preliminaries

In this section, we first introduce how we build the ground truth dataset using traffic traces. Then, we show the assessment criteria.

5.1 Dataset

In this paper, two Internet traffic traces, WIDE [1] and ISP, are used for our experimental study. Table 1 shows the main detail of traffic traces.

Table 1: Traffic Traces

Trace	Data Time	Duration	Type	Volume
ISP-data	2015-08-17	1 day	edge	130.7 GB
WIDE-data	2012-03-30	5 hours	backbone	482.8 GB

The ISP trace was collected from our routers in the edge of a campus network on August 17, 2015 from 1 am to 12 pm. This trace consists of 3 million flows with full packet payload. The WIDE trace was captured by MAWI Working Group in March 2012 that was during 5 hours. In this trace, all the IP addresses are anonymized and each packet just includes forty bytes of application layer payload.

We used two steps to obtain the ground truth dataset. Firstly, we used an open source tool nDPI [2] to label the ISP trace. Besides we used the port-based method to enhance the reliability of the dataset. Because the WIDE trace does not include the full payload, we directly use the port-based approach to label this dataset. Then we used tool Netmate to extract statistical flow characteristics. This tool's job is to classify packets into flows and to calculate the statistics

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_23

of flows.. When building the experimental dataset, we will calculate as many features as possible. Finally, we select 28 flow features, which are described in Table 2.

From the ISP trace, eight protocols, BT, DNS, HTTP, IMAP, NTP, SSDP, SSL, LLMNR, were extracted and constituted the ISP-data. Our sampling rules are that: randomly sampling 40K flows from each protocol if it contains more than 40k flows, otherwise sampling all the flows. The experimental dataset consists of 233k flow randomly sampled from the initial traffic dataset, which is described in Fig. 3(a). From the WIDE trace, seven protocols, POP3, FTP, SSH, SMTP, SSL, DNS, HTTP, were extracted and constituted the WIDE-data. This experimental dataset consists of 56k flows with the rule of randomly sampling up to 10K flows from each protocol, which are described in Fig. 3(b). During experiments, we simulated the problem of unknown applications. Both ISP-data and WIDE-data represent mixed unknown traffic datasets.

Table 2: Network Flow Statistical Features

Category of features	**Description of feature**	**No. of feature**
Packets	Number of packets transferred in unidirection	2
Bytes	Volume of bytes transferred in unidirection	2
Packets Size	Min, Max, Mean and Standard deviation of packets size in unidirection	8
Inter Packet Time	Min, Max, Mean and Standard deviation of inter packet time in unidirection	8
Connection Duration	Min, Max, Mean and Standard deviation of subflow activity time	4
Idle Time	Min, Max, Mean and Standard deviation of subflow idle time	4
	Total	**28**

5.2 Assessment Criteria

To evaluate the effectiveness of our method, we focus on clustering purity. The clustering purity is defined as the average percentage of the dominant class label in each cluster [3]. To calculate the purity, each cluster is assigned to the category which is most frequent in the cluster. The definition of clustering purity is shown in Eq. (12).

$$P(C,S) = \frac{1}{|D|}\sum_{i=1}^{k}\max_{j}|c_i \cap s_j| \tag{12}$$

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_23

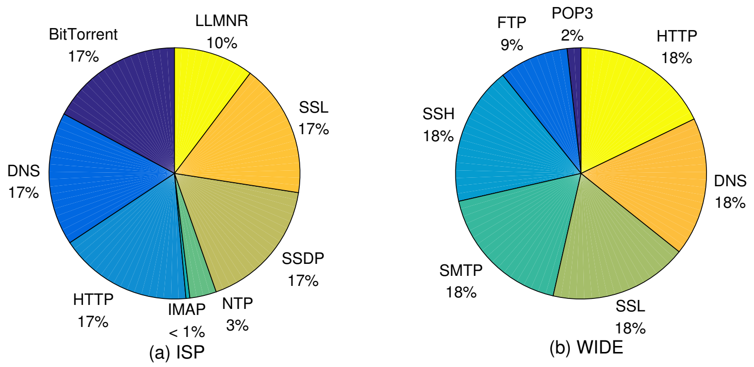


Fig. 3: Class distribution of the dataset

where k is the number of clusters, $C = \{c_1, c_2, ..., c_i\}$ is the set of clusters and $S = \{s_1, s_2, ..., s_j\}$ is the set of classes.

6 Performance Results

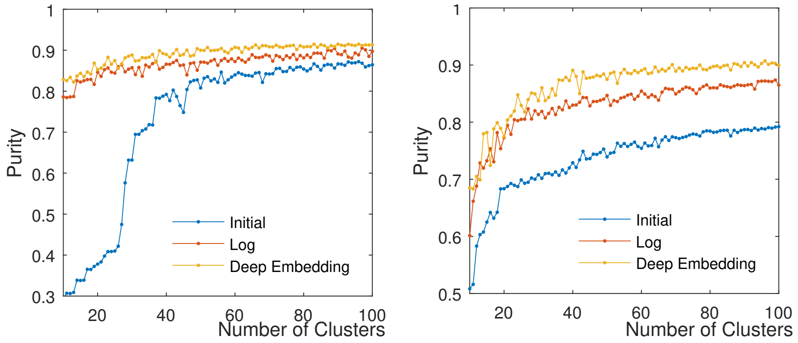
In this section, we first compare our feature extraction method with traditional approaches to explain why we use deep embedding. Secondly, we prove the effectiveness of the modified PCKMeans. Finally, we present and discuss the comprehensive experiments. To evaluate the effectiveness of the method, we use labeled data to simulate unknown traffic. In the experiment, the number of clusters is the only input parameter, which ranges from 10 to 100. Every experiment is repeated 100 times to ensure the reliability of the results.

6.1 Why We Use Deep Embedding

To show the validity of deep embedding, we used the k-means algorithm to identify unknown traffic traces based on three kinds of feature sets: initial statistical features [20], log transformation features [15], and deep embedding features. For each k, we repeat the clustering with different random seeds. Fig. 4(a) and Fig. 4(b) illustrate the purity of clustering on the ISP-data and WIDE-data, respectively.

Our clustering target is to obtain high clustering purity with small cluster number. The results indicate that when the number of clusters is 10, that deep embedding outperforms initial features and log transformation features clustering purity on two datasets. This satisfies the original intention of our algorithm design. Besides, with the increase of cluster number, the clustering purity of deep embedding is almost always higher than that of the other two methods on two datasets.

DOI: 10.1007/978-3-030-22734-0_23

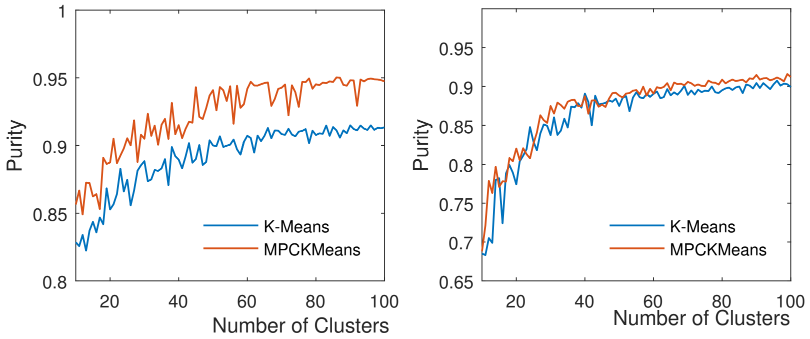


(a) ISP (b) WIDE

Fig. 4: Clustering Purity Comparison of Different Data Preprocessing Methods

6.2 Benefits of Modified PCKMeans

To show the ability of MPCKMeans, we perform the following experiments. In the case bottleneck features as the original input, we compare the effectiveness of the MPCKMeans algorithm and the k-means algorithm for unknown traffic identification.



(a) ISP (b) WIDE

Fig. 5: Clustering Purity Comparision of Different Algorithms

The experimental results are shown in the Fig. 5. Viewing the trend as a whole, the experimental results of *DePCK* are almost always better than the results of k-means. Individually, Fig. 5(a) shows the results on the ISP-data. No matter how the parameter changes, the clustering purity of *DePCK* is always higher than that of k-means by about 5%. In Fig. 5(b), although the advantage of *DePCK* is not as obvious as in Fig. 5(a), the overall result is that our method

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22734-0_23

is superior to the k-means. The results show that the MPCKMeans algorithm can get more pure clusters and is stable in different datasets.

6.3 Performance of *DePCK*

We perform a series of experiments on real-world traces to evaluate the proposed framework and the state-of-the-art methods (k-means [20]and log k-means [15]). All methods use the same datasets and parameters. The parameter ranges from ten to one hundred: k = 10,20,...,50,60,...,100. Table 3 shows the experimental results. When the parameters are the same, our method *DePCK* can always get the best experimental results on both datasets. The *DePCK* achieves over 94.81% clustering purity on the ISP-data and 91.48% clustering purity on the WIDE-data, which is obviously superior to other methods.

Table 3: Experiment Results (The Best Results Are in Bold)

K	ISP-data			WIDE-data		
	k-means	log k-means	*DePCK*	k-means	log k-means	*DePCK*
10	29.50%	78.60%	**85.64%**	50.82%	63.12%	**82.02%**
20	37.80%	84.63%	**88.74%**	68.35%	80.26%	**82.02%**
30	63.19%	85.75%	**90.49%**	70.79%	84.36%	**87.49%**
40	79.21%	86.59%	**91.50%**	72.89%	85.98%	**88.74%**
50	80.79%	87.09%	**94.09%**	73.91%	87.68%	**88.83%**
60	83.97%	87.23%	**94.18%**	75.42%	88.43%	**89.09%**
70	84.18%	88.59%	**94.27%**	77.29%	87.94%	**90.59%**
80	85.40%	88.12%	**94.52%**	78.43%	88.96%	**90.88%**
90	86.66%	89.09%	**94.81%**	78.69%	89.39%	**91.22%**
100	86.44%	89.73%	**94.73%**	79.22%	89.44%	**91.48%**

7 Related Work

How to classify network traffic into known protocols and applications was extensively focused in past studies, but few discussed the identification of unknown traffic. We briefly review the work related to the unknown traffic identification.

In previous studies, (N+1)-Class traffic classification model was proposed to solve the problem of unknown traffic. N represents the number of known classes, and one represents all unknown classes. Erman *et al.* [9] proposed a semi-supervised classification method that can accommodate both known and unknown applications. Then Casas *et al.* [7] use the ensemble clustering technique to improve the semi-supervised method. Later, Erman's work is introduced to classify encrypted traffic by using the composite feature set and combining the first 40-B payload with statistical features of the flow level [14]. Besides, Fu *et al.* construct multiple one-class classifiers to divide traffic into N classes and an unknown class [10]. Although the way that all unknown traffic is identified as one class helps to increase the accuracy of a traffic classification system, the system's ability to efficiently achieve fine-grained identification of unknown traffic is feeble.

In theory, unsupervised methods can solve network traffic identification problem. Unsupervised methods have been widely applied in the network traffic classification, which deserves our reference. In [19], Zander *et al.* presented a classification method based on AutoClass program that uses the Expectation-Maximization (EM) algorithm and mixture models. Then Erman *et al.* [8] applied two unsupervised clustering algorithms, namely k-means and DBSCAN, to classify network traffic and compare them to the previously used AutoClass algorithm. The experimental results showed that both k-means and DBSCAN work very well and much more quickly than AutoClass. Similarly, in Liu *et al.* [15], the author adopted feature selection to find an optimal feature set and log transformation to improve the k-means accuracy. The report of this method showed that overall accuracy was up to 80%, and, after a log transformation, the accuracy was improved to 90% or more. This approach is superior to the previous methods. Zhang's work [20] is based on Erman's semi-supervised classification method. For fine-grained unknown traffic, an update module, based on k-means algorithm, is proposed to finely classify unknown traffic in the system. Besides, some other well-known unsupervised algorithms, such as Fuzzy C-means [13] and hierarchical clustering [4], were also used for traffic classification.

8 Conclusion and Future Work

In this paper, we proposed a robust scheme, the *DePCK*, for unknown traffic identification based on deep auto-encoder and modified pairwise constrained k-means. To the best of our knowledge, this is the first application of bottleneck features and pairwise constrained k-means in this area. Extensive experimental results reveal that *DePCK* achieves better performance compared to the state-of-the-art methods on real-world datasets. In addition, the *DePCK* can deal with textual protocols and binary protocols. To further solve the unknown traffic identification problem, our future research will focus on how to automatically determine the number of traffic clusters.

9 Acknowledgment

This work was supported by the National Natural Science Foundation of china (No.61572496 and No.U1736218). Yafei Sang is the corresponding author.

References

1. http://mawi.wide.ad.jp/mawi/
2. https://www.ntop.org/products/deep-packet-inspection/ndpi/

3. Aggarwal, C.C.: A human-computer interactive method for projected clustering. IEEE transactions on knowledge and data engineering **16**(4), 448–460 (2004)
4. Bacquet, C., Zincir-Heywood, A.N., Heywood, M.I.: Genetic optimization and hierarchical clustering applied to encrypted traffic identification. In: Computational Intelligence in Cyber Security (CICS), 2011 IEEE Symposium on. pp. 194–201. IEEE (2011)
5. Basu, S., Banerjee, A., Mooney, R.J.: Active semi-supervision for pairwise constrained clustering. In: Proceedings of the 2004 SIAM international conference on data mining. pp. 333–344. SIAM (2004)
6. Bengio, Y., Courville, A., Vincent, P.: Representation learning: A review and new perspectives. IEEE transactions on pattern analysis and machine intelligence **35**(8), 1798–1828 (2013)
7. Casas, P., Mazel, J., Owezarski, P.: Minetrac: Mining flows for unsupervised analysis & semi-supervised classification. In: Proceedings of the 23rd International Teletraffic Congress. pp. 87–94. International Teletraffic Congress (2011)
8. Erman, J., Arlitt, M., Mahanti, A.: Traffic classification using clustering algorithms. In: Proceedings of the 2006 SIGCOMM workshop on Mining network data. pp. 281–286. ACM (2006)
9. Erman, J., Mahanti, A., Arlitt, M., Cohen, I., Williamson, C.: Offline/realtime traffic classification using semi-supervised learning. Performance Evaluation **64**(9-12), 1194–1213 (2007)
10. Fu, N., Xu, Y., Zhang, J., Wang, R., Xu, J.: Flowcop: Detecting" stranger" in network traffic classification. In: 2018 27th International Conference on Computer Communication and Networks (ICCCN). pp. 1–9. IEEE (2018)
11. Hornik, K.: Approximation capabilities of multilayer feedforward networks. Neural networks **4**(2), 251–257 (1991)
12. Liou, C.Y., Cheng, W.C., Liou, J.W., Liou, D.R.: Autoencoder for words. Neurocomputing **139**, 84–96 (2014)
13. Liu, D., Lung, C.H.: P2p traffic identification and optimization using fuzzy c-means clustering. In: Fuzzy Systems (FUZZ), 2011 IEEE International Conference on. pp. 2245–2252. IEEE (2011)
14. Liu, H., Wang, Z., Wang, Y.: Semi-supervised encrypted traffic classification using composite features set. Journal of Networks **7**(8), 1195 (2012)
15. Liu, Y., Li, W., Li, Y.C.: Network traffic classification using k-means clustering. In: Computer and Computational Sciences, 2007. IMSCCS 2007. Second International Multi-Symposiums on. pp. 360–365. IEEE (2007)
16. Statistics, I.N.: Internet2 netflow statistics (2011)
17. Wang, Y., Xiang, Y., Zhang, J., Zhou, W., Wei, G., Yang, L.T.: Internet traffic classification using constrained clustering. IEEE transactions on parallel and distributed systems **25**(11), 2932–2943 (2014)
18. Yaman, S., Pelecanos, J., Sarikaya, R.: Bottleneck features for speaker recognition. In: Odyssey 2012-The Speaker and Language Recognition Workshop (2012)
19. Zander, S., Nguyen, T., Armitage, G.: Automated traffic classification and application identification using machine learning. In: Local Computer Networks, 2005. 30th Anniversary. The IEEE Conference on. pp. 250–257. IEEE (2005)
20. Zhang, J., Chen, X., Xiang, Y., Zhou, W., Wu, J.: Robust network traffic classification. IEEE/ACM Transactions on Networking (TON) **23**(4), 1257–1270 (2015)
21. Zhao, S., Zhang, Y., Chang, P.: Network traffic classification using tri-training based on statistical flow characteristics. In: Trustcom/BigDataSE/ICESS, 2017 IEEE. pp. 323–330. IEEE (2017)