Semi-supervised network traffic classification using deep generative models

Tan Li, Shuangwu Chen, Zhen Yao, Xiang Chen, Jian Yang
Department of Automation,
University of Science and Technology of China
Hefei, China

Abstract—**Network traffic classification plays a fundamental role in area of network management and security. Recent days, machine learning techniques have been used to classify network traffic. In particular, semi-supervised learning is very fit for practical scenarios, where pre-labelled training flows are hard to obtain. In this paper, a semi-supervised classification scheme is proposed for network traffic classification by using deep generative models. Specifically, the feature extractor module aims to automatically find representation features of raw traffic data in a lower dimensional feature space. Subsequently, using these representation features, a separated classifier is trained by the semi-supervised classification module. The method is verified with three different levels of datasets: *Anomaly detection-level*, *protocol-level* and *application-level*. The results show that our scheme can not only detect malware traffic, but also classify the traffic according to their protocol, application, and attack types. Using less than 20% labelled flows of the whole dataset, we can achieve the accuracy of over 95% which is a satisfying value compared with supervised learning method.**

Keywords- network traffic classification; semi-supervised learning; deep generative model; feature representation learning; network anomaly detection

I. INTRODUCTION

With the advent of the big data era, new types of Internet applications have emerged and the composition of networks has become more complex. Network managers should learn more information about applications and protocols of the network traffic for better implement network scheduling and attack defense policies. Over the past decades, network traffic classification has attracted a large amount of attentions due to its indispensable value in optimizing network configuration, reducing network security risks and providing better service quality based on user behaviour analysis [1].

There are three categories of traffic classification methods: port-based, deep packets inspection (DPI)-based, and flow statistics-based method [1]. The traditional port-based method identifies the network application by mapping the port number in the data packet to well-known applications. But with the growth of dynamic port services and adoption of network address transformation, the port-based classification becomes unreliable. The DPI-based method searches for the application's signature in the payload of IP packets that can help avoid the problem of dynamic ports [2]. Currently, it is the most practical method. However, DPI-based method fails when it comes to encrypted traffic.

Instead of performing traffic classification by matching predefined rules, statistics-based method utilizes a set of observable features of traffic flows (e.g., mean and variance of packet size and interval time between packets) that demonstrate the different behaviours and latent properties of network traffic. Most statistics-based methods can be further divided into supervised or unsupervised learning algorithms. On the one hand, By extracting patterns from sufficient labelled flows with a set of selective features, the supervised learning mainly uses the machine learning methods to learn a traffic classifier. However, the performance depends on the quality and quantity of the pre-labelled flows which are hard to obtain in practice [3]. On the other hand, the unsupervised methods categorize a set of unlabelled flows by the clustering techniques, like K-means. Then, by means of manual annotation, the cluster results will be used to build a classifier. However, the accuracy are far from satisfactory.

In this paper, we propose a semi-supervised network traffic classification method by utilizing density models which are based on recent advances in deep generative models and variational inference theory [4] [5]. Our scheme only requires a few labelled flows but achieves a satisfying accuracy of over 90%. The major contributions of this paper are summarized as follow:

- We make use of a deep generative model as a feature extractor to provide representation features of raw flows. Related flows will be clustered onto a latent feature space which makes the classification more accurate.
- We propose a semi-supervised scheme to address the traffic classification problem with only a few labelled flows. The method combines successful results in the field of deep generative models and variational inference theory these days.
- We carry out the experiments on three different levels of network traffic datasets. The results show that our scheme can not only detect malware traffic, but also classify the traffic according to their protocol, application, and attack types.

II. RELATED WORK

Port-based and DPI-based traffic classification approaches are widespread in practice application. Finsterbusch et al. [6] made a survey of current DPI-based traffic classification methods. Related works mainly focus on how to exactly

978-1-5386-8097-1/18/$31.00 ©2018 IEEE

determine the mapping rules, reduce time consumption of detecting the payload and improve performance.

Since the rule-based methods are suffering the problems of dynamic ports and encrypted traffic, recent research mainly focuses on how to use flow statistics-based methods with the machine learning techniques. Nguyen and Armitage [2] make a survey to compare the performance of several machine learning approaches deal with network traffic classification. Provided with a completely labelled training dataset, the classifier can be trained using the supervised learning algorithm. For example, Naïve Bayes [9], Support Vector Machines (SVM) [7] and Decision Trees [8] have been evaluated with respect to this problem using selective flow features. With the development of artificial intelligence (AI), in addition to above traditional machine learning methods, deep learning techniques are also applied to build the classifier. Javaid et al. [10] proposes a network intrusion detection scheme taking advantage of sparse auto-encoder (SAE). Wang et al. [11] presents an encrypted traffic classification method by utilizing 1D convolution neural networks. Long short-term memory networks (LSTM) [12] is carried out to learn temporal flow features and improve the classification accuracy.

Some unsupervised learning methods, such as K-Means [13] and Expectation Maximization (EM) [14] are used to cluster traffic flows automatically. Jonas et al. [3] proposes an unsupervised learning traffic classification algorithm using a neural auto-encoder. Semi-supervised learning is a hybrid approach which feeds abundant unlabelled flows and a small quantity of labelled flows to the learning classifier. Erman et al. [13] firstly uses the K-Means algorithm in the clustering procedure, and then the labelled flows are used to classify the clusters results to applications.

In this paper, our approach is also semi-supervised, but it is an end-to-end classification model, rather than a separate two-step process. The labelled and unlabelled flows are used to train the model simultaneously, which is more efficiently and automatically than the previous "feature – cluster – label" scheme. Meanwhile, the selected features play an important role in reflecting the flow behaviors. So instead of using the handcrafting flow statistics, such as the set of 248 statistic features proposed in [9], we make used of the advanced deep generative model. Specifically, the Variational Auto-Encoder (VAE) [4] represents the features of the raw flows by embedding them into a lower dimensional latent feature space.

III. Deep Generative Models For Semi-Supervised Network Traffic Classification

In this section, we present a semi-supervised scheme to deal with network traffic classification problem. There are two main modules in the proposed framework: feature extractor module and semi-supervised classification module. Figure 1 shows the overall framework. The feature extractor module, which should be pre-trained, aims to automatically find representation features of the raw flows in an unsupervised way. Subsequently, using representation features, a separated classifier is trained by the semi-supervised classification module. In this module, we propose a probabilistic model to describe the flows generated by certain traffic types in addition to some latent variables.

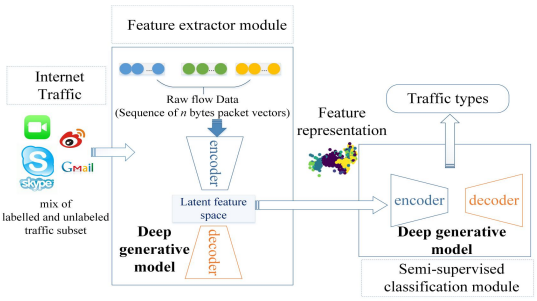

Figure 1. The semi-supervised traffic classification framework.

A. Feature Extractor

A variational auto-encoder (VAE) is used to learn the representation features of the raw flows — the sequence of n bytes network packet. In Figure 2, using the VAE, we can obtain the representation features in a lower dimensional space. These low dimensional features are easier to separate than raw flows since flows with similar characteristics are clustered on the latent feature space. We model the traffic data as pairs: $(x, y) = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$,where $x_i \in \mathbb{R}^D$ denotes the i-th raw flow and $y_i \in \{1, \ldots, L\}$ is its class label. Every raw flow x_i have a corresponding representation feature z_i .

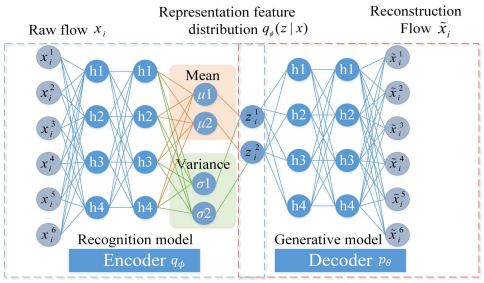

Figure 2. The VAE based feature extractor model.

As shown in Figure 2, we refer to $q_\phi(z|x)$ as a probabilistic *encoder*, since given raw flow $\boldsymbol{x}$ it can produce a posterior distribution over representation feature $\boldsymbol{z}$, which we denote as "*representation feature distribution*". Specifically, the *encoder* we use to infer representation features is a Gaussian network:

$$q_\phi(z|x) = N(z|\mu_\phi(x), diag(\sigma^2(x))) \quad (1)$$

The *encoder* is constructed as deep neural networks form. The mean and variance of this representation distribution $q_\phi(z|x)$ are the output of encoder. Here, the functions $\mu_\phi(x)$, $\sigma^2(x)$ are represented as Multi-Layer Perceptron (MLPs).

We also refer to $p_\theta(x|z)$ as a probabilistic *decoder*, since given the flow representation feature $\boldsymbol{z}$ it produces a

distribution over the possible values of the raw flow $\boldsymbol{x}$. The *decoder* we use to reconstruct flows is:

$$p(z) = N(z \mid 0, I)\,;\; p_\theta(x \mid z) = f(x; z, \theta)\,, \tag{2}$$

Where $f(x; z, \theta)$ is a likelihood function whose probabilities are formed by deep neural networks of parameter θ and a set of representation features $\boldsymbol{z}$.

The variational auto-encoder enables us to obtain a representation feature $\boldsymbol{z}$ for every raw flow x through the inference network $q_\phi(z \mid x)$. It should be noted that this procedure is completely unsupervised.

B. *Semi-Supervised Classification*

Under the semi-supervised classification situation, only a small set flows have known class labels. To address the problem of missing labels, we propose another generative model which is a deformation of VAE. The model describes the flows generated by latent class variables $\boldsymbol{y}$ in addition to other latent variables z' [1]. Figure 3&Figure 4 show the structure details.

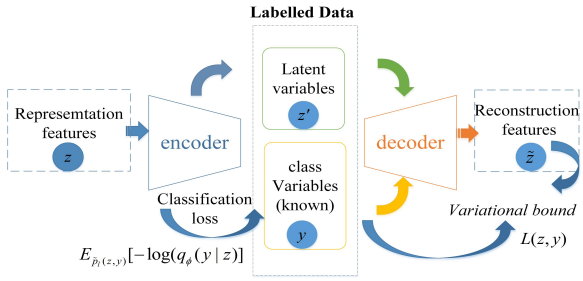

Figure 3. The semi-supervised classification module for labelled data.

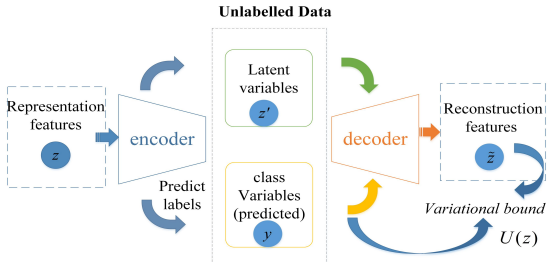

Figure 4. The semi-supervised classification module for unlabelled data.

The classification is integrated into the inference process. Different from feature extractor module, for the *encoder* in classification module, we used a Gaussian network to infer both latent variables z' and $\boldsymbol{y}$ as:

$$\begin{aligned} q_\phi(z' \mid z) &= N(z \mid \mu_\phi(z), diag(\sigma^2(z)))\,; \\ q_\phi(y \mid z) &= Cat(y \mid \pi_\phi(z)) \end{aligned} \tag{3}$$

where $Cat(y \mid \pi)$ is the multinomial distribution. If there is no class label for the flow previously, the class label $\boldsymbol{y}$ will be seen as a latent variable which is needed to inference. The functions $\mu_\phi(z)$, $\sigma^2(z)$ and $\pi_\phi(z)$ are represented as MLPs. The *decoder* in classification module we use to reconstruct features is:

$$\begin{aligned} p(y) = Cat(y \mid \pi)\;\;;\;\; p(z') = N(z' \mid 0, I) \\ p_\theta(z' \mid y, z) = f(z; y, z', \theta)\,, \end{aligned} \tag{4}$$

similarly, $f(z; y, z', \theta)$ is a likelihood function, parameterized by deep neural networks with parameter θ and a set of latent variables $\boldsymbol{y}$ and z'.

The inference network $q_\phi(y \mid z)$ now can be used as a *classifier*, from which we can get predictions for any flows with missing labels.

C. *Objective Functions and Optimization*

For the feature extractor module, the objective function is the *variational bound* [4] of the VAE model which can be written as:

$$L(\theta, \phi; x^{(i)}) = D_{KL}(q_\phi(z \mid x^{(i)}) \,\|\, p_\theta(z)) + E_{q_\phi(z \mid x^{(i)})}[\log p_\theta(x^{(i)} \mid z)] \tag{5}$$

For the semi-supervised classification module, we have two situations to consider [15]:

(1) Considering the labelled flows, as shown in Figure 3, the variational bound is a extension of Eq.(5):

$$\begin{aligned} &L(z^{(i)}, y) \\ &= D_{KL}(q_\phi(z' \mid z^{(i)}) \,\|\, p_\theta(z')) + E_{q_\phi(z' \mid z^{(i)})}[\log p_\theta(z^{(i)} \mid y, z')] \\ &= E_{q_\phi(z' \mid z^{(i)}, y)}[\log p_\theta(z^{(i)} \mid y, z') + \log p_\theta(z') + \log p_\theta(y) - \log q_\phi(z' \mid z^{(i)})] \end{aligned} \tag{6}$$

(2) For the unlabelled flows, as shown in Figure 4, the class label $\boldsymbol{y}$ is seen as a latent variable over the distribution, hence the variation bound for the flow with unknown label is:

$$\begin{aligned} &U(z^{(i)}) \\ &= D_{KL}(q_\phi(y, z' \mid z^{(i)}) \,\|\, p_\theta(z', y)) + E_{q_\phi(y, z' \mid z^{(i)})}[\log p_\theta(z^{(i)} \mid y, z')] \\ &= \sum_y q_\phi(y \mid z^{(i)})(-L(z^{(i)}, y)) + H(q_\phi(y \mid z^{(i)})) \end{aligned} \tag{7}$$

We set distribution of the labelled and unlabelled flow sets as $\tilde{p}_l(z, y)$ and $\tilde{p}_u(z)$, respectively. Then the variational bound for the overall dataset is:

$$J = \sum_{(z,y) \sim \tilde{p}_l(z,y)} L(z, y) + \sum_{z \sim \tilde{p}_u(z)} U(z) \tag{8}$$

We also add the classification loss of the labelled flows, and the overall objective function is:

$$L = J + \alpha E_{\tilde{p}_l(z,y)}[-\log(q_\phi(y \mid z))] \tag{9}$$

The Eq.(5) and Eq.(9) provide objective functions for optimization of feature extractor module and the semi-supervised classification module, respectively. Using *stochastic gradient variational Bayes* (SGVB) [5] method,

[1] The input of this classification module is the representation feature $\boldsymbol{z}$ obtained by the feature extractor module. To distinguish the latent variables between this two module, we denote the latent variables as z' for the classification module.

both the parameters θ and ϕ of the *encoder* and *decoder* can be optimized.

Considering Eq.(5), as the prior $p(z) = N(0, I)$ and the posterior distribution $q_\phi(z \mid x)$ are Gaussian distribution, the KL term can be computed analytically. Hence only the reconstruction error $E_{q_\phi(z|x^{(i)})}[\log p_\theta(x^{(i)} \mid z)]$ requires estimation by sampling from the representation distribution $q_\phi(z \mid x)$. By using the *reparameterize trick* [4], $\tilde{z} = g_\phi(\varepsilon, x)$ with $\varepsilon \sim p(\varepsilon)$, we can optimize the parameters by derivation while still maintaining the ability to randomly sample from the representation distribution. We can now obtain the estimator of the Eq.(5):

$$\tilde{L}(\theta, \phi; x^{(i)}) = D_{KL}(q_\phi(z \mid x^{(i)}) \parallel p_\theta(z)) + \frac{1}{L}\sum_{l=1}^{L}(\log p(x^{(i)} \mid z^{(i,l)})) \tag{10}$$

Where $z^{(i,l)} = g_\phi(\varepsilon^{(i,l)}, x^{(i)})$ and $\varepsilon^{(l)} \sim p(\varepsilon)$. The gradients of Eq.(5) for the feature extractor with respect to parameter θ and φ can be written as:

$$\nabla_{\{\theta,\phi\}}\tilde{L}(\theta, \phi; x^{(i)}) = \nabla_{\{\theta,\phi\}} D_{KL}(q_\phi(z \mid x^{(i)}) \parallel p_\theta(z)) + \frac{1}{L}\sum_{l=1}^{L}\nabla_{\{\theta,\phi\}}(\log p_\theta(x^{(i)} \mid \mu_\phi(x^{(i)}) + \sigma_\phi(x^{(i)}) \odot \varepsilon)) \tag{11}$$

The same computation strategy can be used to compute the gradients of Eq.(9). Since the KL term can be computed analytically and the gradients of the reconstruction error terms can be estimated using the process Eq.(11).

IV. EXPERIMENT

In this section, we perform various experiments on public ISCX VPN traffic dataset [16] and USTC-TFC2016 dataset [17] to evaluate the performance of the proposed semi-supervised classification method. These experiments have the following purposes:

- Evaluate the performance of the feature extractor module;
- Evaluate the effect of the quantity of labelled flows on the semi-supervised classification module's performance;
- Evaluate the performance of the semi-supervised classification method on different levels of datasets: *Anomaly detection-level*: detect the malware flows from network traffic. *Protocol-level:* classify the flows based on protocol type; *Application-level*: classify the flows based on application/attack type.

A. Experiment Implementation

1) Datasets

Most current public traffic datasets are based on flow statistics. But since we have to show the model's ability to extract latent flow features, we choose the following three dataset containing raw traffic data rather than statistics:

(1) ISCX VPN Dataset: Includes 6 types of regular encrypted protocol traffic.

(2) USTC-TFC2016 Dataset: Wang et al. [19] published the dataset which is made up of two parts: the MALWARE TRAFFIC part contains 10 types of malware traffic; The NORMAL TRAFFIC part contains 10 types of normal traffic which are collected by network traffic simulation equipment.

(3) USTC Anomaly Detection Dataset: To further explore the model's ability of anomaly detection, each flow in USTC-TFC2016 is labelled either normal or malware according to its type to construct a 2-class dataset.

The flow compositions of the four datasets are shown in TABLE Ⅰ.

TABLE I. THE FLOW COMPOSITION OF DATASETS.

USTC Malware Traffic	USTC Normal Traffic	ISCX VPN	USTC Anomaly Detection
Cridex	BitTorrent	Chat	Normal
Geodo	Facetime		
Htbot	FTP	Email	
Miuref	Gmail		
Neris	MySQL	File	
Nsis-ay	Outlook		Malware
Shifu	Skype	P2P	
Tinba	SMB		
Virut	Weibo	Streaming	
Zeus	WorldOfWarcraft	VoIP	

2) Experiment Setup

We use the *Pytorch* [20] as the deep learning software framework for our experiments. Our experiments run on Ubuntu 16.04 64bit OS. The server is Intel i7-6700 CPU with 8GB memory. A Nvidia Tesla K80 GPU is used to train the neural network. Datasets are divided into training dataset and test dataset in a ratio of ten to one. The training dataset is further divided into labelled subset and unlabelled dataset.

B. Experiment Evaluation

1) Evaluation Metrics

Four evaluation metrics [2] are used to show our method's performance: accuracy (A), precision (PR), recall (RC) and f1 value (F1). We use the Accuracy to evaluate the overall performance of a classifier. And the other three metrics (PR, RC, F1) are used to evaluate the classification results of every class of the dataset.

$$A = \frac{TP + TN}{TP + FP + FN + TN}, \quad P = \frac{TP}{TP + FP}$$

$$R = \frac{TP}{TP + FN}, \quad F1 = \frac{2 * PR}{P + R}$$

By using above metrics, we can judge the performance of the classifier after training.

2) Experiment Results

a) Feature representation

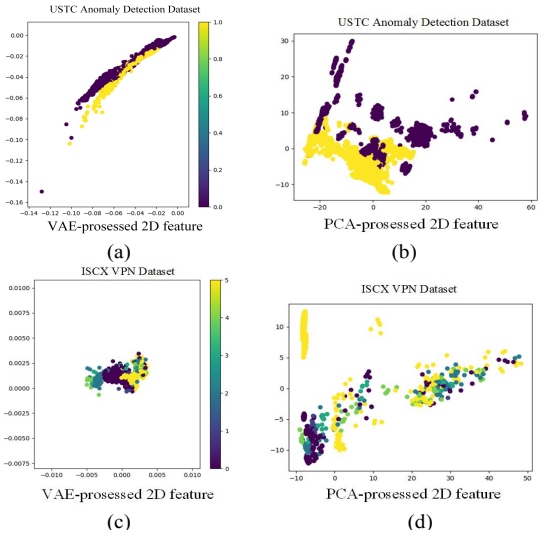

Figure 5. The 2D latent feature of traffic flows.

(a) and (c) in Figure 5 are corresponding to USTC Anomaly Detection Dataset and ISCX VPN Dataset, respectively. They show the 2D representation feature obtained by the VAE-based feature extractor module; (b) and (d) in Figure 5 show the dimension reduction results obtained by performing the PCA (Principal Component Analysis) method on the two datasets. Comparing the performance on the 2D plane, the VAE-processed data has better clustering results. That means the same type traffic has the similar inner characteristics and after this low dimensional embedding, the raw flows are easier separated in the representation feature space.

b) The semi-supervised classification performance

Firstly, to evaluate the effect of labelled flows on classification performance, we choose different amounts of the labelled flows and compare the variation tendency of evaluation metrics. We set the *n_labels (labelled flows per traffic class)* as 20, 50, 100, 200, each accounted 0.31%, 0.78%,1.56% and 3.12% for the USTC Malware Traffic Dataset; 0.31%, 0.77%,1.55% and 3.10% for the USTC Normal Traffic Dataset; 1.77%, 4.43%, 10.0% and 20% for the ISCX VPN Dataset; 0.15%, 0.39%, 0,78% and 1.56% for the USTC Anomaly Detection Dataset. Compared with the supervised learning methods, which commonly set the labelled trainset as 70~80% of the overall dataset, our method only uses minimally labelled flows. Figure 6 shows the evaluation metrics with different *n_labels* of four datasets, respectively. We also list the specific values of classification metrics for USTC Malware Traffic Dataset in TABLE Ⅱ.

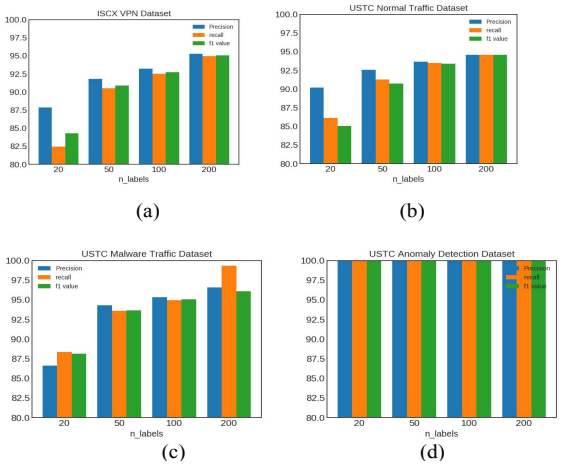

Figure 6. The metric values of four datasets.

As the results shown in Figure 6, although there are only 20 labelled flows per class, the precision values of four datasets are all over 85%, especially for the USTC Anomaly Detection Dataset, reaching 100%. When the labelled flows increases to 50 per class, the precision, recall and f1 value for all four datasets are over 90%. The metric values reach about 95% when the label number increases to 200 per class.

Among the four datasets, the proposed method has the best performance on USTC Anomaly Detection Dataset (evaluation metrics reaching 100%). It means that the normal flows and malware flows have a big difference in flow characteristics, so they can be separated easily. When it comes to protocol-level and application-level classification, the method's performance has a little decline. The possible reason is that applications belonging to different traffic class may have a certain degree of similarity, resulting in some misclassifications. For example, Figure 7 presents the confusion matrix of USTC Malware Traffic Dataset with 50 labelled flows per class. The depth of the diagonal colour represents the classification accuracy of the corresponding class. We can see that most classes have been classified accurately, only the *Neris* and *Virut*'s performance are not up to satisfaction.

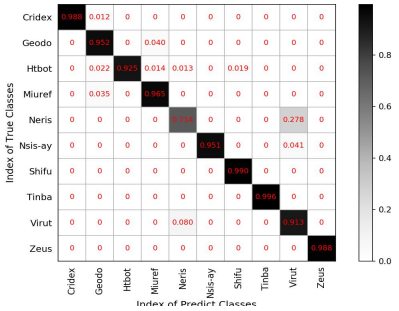

Figure 7. The confusion matrix of USTC Malware dataset(n_labels = 50)

TABLE II. THE PRECISION, RECALL AND F1 VALUE OF THE USTC MALWARE TRAFFIC DATASET.

n_labels	20			50			100			200		
Traffic type	**PR**	**RC**	**F1**	**PR**	**RC**	**F1**	**PR**	**RC**	**F1**	**PR**	**RC**	**F1**
Cridex	95.87	98.78	97.30	99.88	98.78	99.33	98.92	99.88	99.39	99.88	99.27	99.57
Geodo	100	92.84	96.28	93.41	95.22	94.31	98.21	98.51	98.36	98.52	99.25	98.88
Htbot	65.34	87.91	74.97	98.33	92.46	95.31	98.85	94.82	96.79	99.02	95.60	97.28
Miuref	91.77	100	95.71	93.22	96.49	94.82	97.34	100	98.65	96.43	100	98.18
Neris	78.47	62.52	69.63	89.79	71.41	79.55	88.46	77.53	82.63	95.62	79.53	86.83
Nsis-ay	83.39	88.47	85.85	96.97	95.06	96.01	98.28	94.23	96.22	98.45	94.23	96.30
Shifu	97.25	99.27	98.25	98.76	98.96	98.86	99.79	98.96	99.27	99.90	99.58	99.74
Tinba	96.48	99.88	98.15	100	99.65	99.82	99.88	99.88	99.88	100	99.88	99.94
Virut	72.79	60.74	66.22	69.03	91.26	78.06	72.45	88.34	79.61	74.88	95.55	83.96
Zeus	100	91.71	95.67	100	98.84	99.42	99.83	99.34	99.58	100	100	99.67
Average	**86.63**	**88.32**	**88.12**	**94.28**	**93.61**	**93.67**	**95.34**	**94.96**	**95.03**	**96.55**	**99.34**	**96.07**

Figure 8 shows the accuracy of four datasets with the different number of labelled flows.

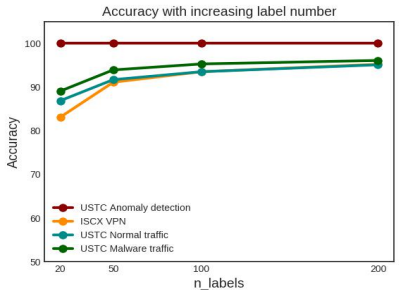

Figure 8. Accuracy with different labelled flows

We can see that the accuracy is increasing with the number of labelled flows. Specifically, the proposed method has the highest accuracy on USTC Anomaly Detection Dataset, reaching 100% with only 0.15% labelled flows of the whole dataset. On the other three datasets, accuracy reaches over 80% with 20 labelled flows per class and it increases to over 90% with 50 labelled flows. When it comes to 200 labelled flows per class, the accuracy reaches 95% which is a satisfying value compared to the supervised classification methods, but we only use less than 20% labelled flows of the whole dataset. Similar to our method, [3] using a neural auto-encoder to cluster the flows and the cluster is assigned the most similar label from the previously labelled data. The method only achieves average 80% precision, 75% recall and 76% f1 score.

V. CONCLUSION

This paper addresses the problem of network traffic classification when the labels are impossible to obtain for the entire dataset. Our proposed method gets rid of handcrafting features, but use the variational auto-encoder to extract the latent feature from raw traffic in an unsupervised way. Then using the representation features, a deep generative model based traffic classifier is trained with mix of abundant unlabelled flows and few labelled flows. Our evaluation is carried out on four different levels of datasets.

There are some problems require further study in future work. Firstly, we should improve the feature representation ability as well as classification performance by using other deep learning techniques, such as the convolutional neural network (CNN) Moreover, as statistic flow features are proven to be useful for network classification, we will integrate the statistic features with the raw traffic data to do further exploration.

ACKNOWLEDGMENT

This work is supported in part by the Equipment Pre-Research project under Grant (No.614B0801010a).

REFERENCES

[1] Biersack, E., Callegari, C., Matijasevic, M. 2013. Data traffic monitoring and analysis.

[2] Nguyen, T. T., Armitage, G. 2008. A survey of techniques for internet traffic classification using machine learning. IEEE Communications Surveys & Tutorials, 10(4), 56-76.

[3] Hochst, J., Baumgartner, L., Hollick, M., Freisleben, B. 2017. Unsupervised Traffic Flow Classification Using a Neural Autoencoder. In 2017 IEEE 42nd Conference on Local Computer Networks (LCN) pp. 523-526

[4] Kingma, D. P. and Welling, M. 2014. Auto-encoding variational Bayes. In Proceedings of the International Conference on Learning Representations (ICLR).

[5] Rezende, D. J., Mohamed, S., and Wierstra, D. 2014. Stochastic backpropagation and approximate inference in deep generative models. In Proceedings of the International Conference on Machine Learning (ICML), volume 32 of JMLR W&CP.

[6] Finsterbusch, M., Richter, C., Rocha, E., Muller, J. A., Hanssgen, K. 2014. A survey of payload-based traffic classification approaches. IEEE Communications Surveys & Tutorials, 16(2), 1135-1156.

[7] Este, A., Gringoli, F., Salgarelli, L. 2009. Support vector machines for TCP traffic classification. Computer Networks, 53(14), 2476-2490.

[8] Bujlow, T., Riaz, T., Pedersen, J. M. 2012. A method for classification of network traffic based on C5. 0 Machine Learning Algorithm. In, 2012 International Conference on Computing, Networking and Communications (ICNC) pp. 237-241

[9] Moore, A. W., Zuev, D. 2005. Internet traffic classification using bayesian analysis techniques. In ACM SIGMETRICS Performance Evaluation Review. Vol. 33, No. 1, pp. 50-60. ACM.

[10] Javaid, A., Niyaz, Q., Sun, W., & Alam, M. 2016. A deep learning approach for network intrusion detection system. In Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies.

[11] Wang, W., Zhu, M., Wang, J., Zeng, X., & Yang, Z. 2017. End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In Intelligence and Security Informatics (ISI), 2017 IEEE International Conference on pp. 43-48.

[12] Wang, W., Sheng, Y., Wang, J., Zeng, X., Ye, X., Huang, Y., Zhu, M. 2018. HAST-IDS: learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. IEEE Access, 6, 1792-1806.

[13] Erman, J., Mahanti, A., Arlitt, M., Cohen, I., Williamson, C. 2007. Semi-supervised network traffic classification. In ACM SIGMETRICS Performance Evaluation Review. Vol. 35, No. 1, pp. 369-370.

[14] Erman, J., Arlitt, M., & Mahanti, A. 2006. Traffic classification using clustering algorithms. In Proceedings of the 2006 SIGCOMM workshop on Mining network data. pp. 281-286.

[15] Kingma, D. P., Mohamed, S., Rezende, D. J., Welling, M. 2014. Semi-supervised learning with deep generative models. In Advances in Neural Information Processing Systems. pp. 3581-3589.

[16] ISCX VPN-nonVPN traffic dataset. http://www.unb.ca/cic/datasets/index.html

[17] USTC-TFC2016 dataset https://github.com/echowei/DeepTraffic

[18] Wang, W., Zhu, M., Zeng, X., Ye, X., Sheng, Y. 2017. Malware traffic classification using convolutional neural network for representation learning. In 2017 International Conference on Information Networking. pp. 712-717.

[19] Pytorch https://pytorch.org/

[20] Zander, S., Nguyen, T., Armitage, G. 2005. Automated traffic classification and application identification using machine learning. The IEEE Conference on Local Computer Network. pp. 250-257.