Using Parametric t-Distributed Stochastic Neighbor Embedding Combined with Hierarchical Neural Network for Network Intrusion Detectione

Huijun Yao[1], Chaopeng Li[2], and Peng Sun[3]
(Corresponding author: Chaopeng Li)

Jiangsu Cable Technology Research Institute Co.Ltd[1]
Nanjing, Jiangsu Province, 210001, China
National Network New Media Engineering Research Center, Institute of Acoustics, Chinese Academy of Sciences[2]
Beijing, 100190, China
(Email: licp@dsp.ac.cn)

(Received July 29, 2019; Revised and Accepted Dec. 6, 2019; First Online Feb. 9, 2020)

Abstract

Parametric t-distributed stochastic neighbor embedding (t-SNE) algorithm is a kind of unsupervised dimensionality reduction method which is widely used and effectively. However, current research rarely involves the application of parametric t-SNE in network attack detection. Simultaneously, it is rare to apply a reasonable model for parametric t-SNE. Therefore, we propose a novel unsupervised dimensionality reduction algorithm to detect attack behaviors, which uses t-SNE combined with a hierarchical neural network. This algorithm maps a high-dimensional network data space into a low-dimensional latent space. Furthermore, we evaluate the performance of the parametric t-SNE method in experiments using two public network intrusion datasets and a self-collected network dataset. In experiments, several unsupervised dimensionality reduction algorithms are discussed and compared with the algorithm we proposed. This comparison shows that parametric t-SNE based on hierarchical neural network gets excellent dimensionality effect, which achieved a maximum of 99% accuracy for 1-nearest neighbor.

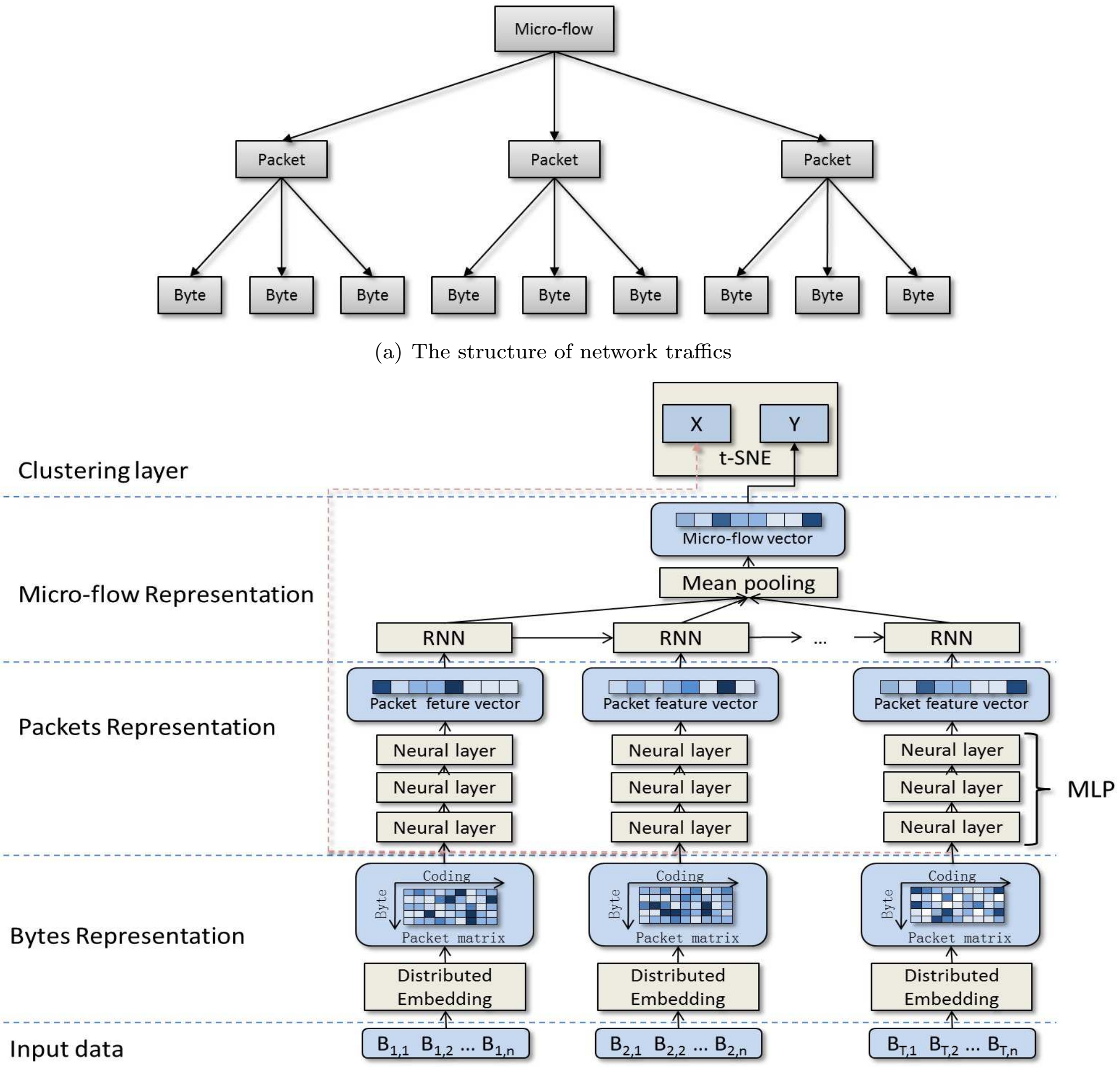
Keywords: Hierarchical Neural Network; Network Intrusion Detection; Parametric T-SNE

1 Introduction

Network data possesses high-dimensional characteristics, which hinders a machine learning model from achieving good performance. Therefore, dimensional reduction is commonly used for a large amount of high-dimensional network data. Traditional reduction algorithms, such as principal component analysis (PCA) and neighborhood components analysis [22], are the commonly used linear reduction techniques. However, these linear reduction algorithms are not ideal when dealing with nonlinear data in a high-dimensional space. In addition, auto-encoders proposed by Hinton [6] can map high-dimensional data by maximizing the variances in latent space. Manifold learning is another such reduction algorithm. Various algorithms, such as Isomap [8], Locally Linear Embedding (LLE) [9], and Maximum Variance Unfolding (MVU) [15], focus more on the local structure of the high-dimensional data. Unfortunately, these algorithms are non-parametric and cannot map the out-of-sample data. A typical t-distributed stochastic neighbor embedding (t-SNE) algorithm [20] is another such non-parametric manifold learning algorithm. Furthermore, Maaten *et al.* [3] presented a parametric t-SNE model based on stacked restricted Boltzmann machine models [12] and solved this problem of out-of-sample data.

However, for network data, it cannot simply build a stacked and fully connected neural network model because of the hierarchical structure of network data. A network streaming data consists of two layers, *i.e.*, packet and micro-flow layers. The micro-flow layer is a set of IP packets that contain the same source IP address, destination IP address, source port, and destination port; and are from the same time window. In this study, micro-flow (defined by five tuple) data is considered to be a sequence of network packets, and these packets can be considered as limited-length data blocks. Thus, a packet layer means byte-level data and the streaming layer is a sequence of packets. To detect network attacks by modeling both packet and streaming layers, a hierarchical model is designed in this study.

This study aims at investigating and proposing a new



(a) The structure of network traffics

(b) The RNN-MLP model

Figure 1: The structure of network traffics and RNN-MLP model

parametric t-SNE model, which is adapted to the hierarchical structure of network data and performs unsupervised dimensionality reduction because of the lack of labels for malicious network behaviors. Unlike typical t-SNE, the new algorithm should solve the problem of out-of-sample data. We performed dimensional reduction and visualization based on the Defense Advanced Research Projects Agency (DARPA) 1998 dataset [4] and the Information Security Centre of Excellence (ISCX)-2012 dataset [11]. The effects of various hyper-parameters, such as perplexity, learning rate, packet length, and flow length, on the results of dimensionality reduction are discussed. The experiments show that the parameterized t-SNE method has about four to five percent absolute improvement as compared with other classical dimensionality reduction methods, such as using auto-encoders and PCA.

The remainder of this study is organized as follows. In the second section, the parametric t-SNE algorithm combined with a hierarchical deep neural network is described. Furthermore, in the third section, the experimental setup and results, and performance of various influencing factors are discussed. Finally, conclusions and future work are described in the fourth section.

2 Parametric T-SNE Based On Hierarchical Deep Neural Network

In this section, we introduce parametric t-SNE based on the recurrent neural network (RNN)-multilayer perceptron (MLP) model [5, 17]. First, a typical t-SNE algorithm is described, which is a global dimensional reduction method. However, the out-of-the-sample extension is invalid. Further, we introduce the parametric t-SNE algorithm. As an improvement, the algorithm can train the RNN-MLP model while performing global dimensionality reduction, thereby making the model effective for external samples. In addition, we also discuss the preprocessing method of network traffics.

2.1 Structure Of Micro-flow And Hierarchical Neural Network

In this study, the micro-flow is the intrusion detecting object. The micro-flow sequence is divided into two layers, *i.e.*, flow and packet layers. The network data is temporal sequence and hierarchical. In Figure 1(a), the structure of micro-flow is shown. One micro-flow is composed of an ordered set of network packets and one packet is composed of bytes. Therefore, it is necessary to design a hierarchical model corresponding for the special data structure.

Inspired by the special structure of network traffics, we design a hierarchical deep neural network model, named RNN-MLP. The structure of this model is shown in Figure 1(b). The model consists of 4 parts. The first layer from the bottom is the byte representation. The second layer is the packet representation, and the third layer is RNN model, which is the flow representation. The RNN model is a deep neural network, which is suitable for modeling temporal sequences, such as speech recognition [1], language models [10] and micro-flow [14]. The top layer is a t-SNE clustering model.

1) Byte representation. In this paper, we adopt distributed embedding for byte representation. The network packets are composed of bytes,which are presented as $packet = \{b_1, b_2, \ldots, b_n\}$, where n is the number of bytes in a packet. As the input of an embedding function f_{emb}, each byte is mapped to a k-dimensional byte-embedding vector, and each element of the vector follows a uniform distribution from 0 to 1. The mapping packet is contracted presented as:

$$v_p = \{f_{emb}(b_1), f_{emb}(b_2), \ldots, f_{emb}(b_n)\}$$

where, v_p is the packet vector which is concatenated by byte vectors and is taken as the input for the following RBM model.

2) Packet representation. The packet representation refers to the whole MLP because raw data are recommended as inputs in the deep neural network commonly. The packets of byte-level data are directly considered as model inputs. The output of MLP is presented as follows:

$$o_{mlp} = \theta(W_{m\_o} \cdot \theta(W_{m\_h} \cdot x + b_{m\_h}) + b_{m\_o}),$$

where x refers to input data that equals to v_p; W_{m_o} and W_{m_h} are the weights of the output and hidden layers, respectively; b_{m_h} and b_{m_o} are the biases of the hidden and output layers, respectively; the function $\theta(\cdot)$ is the activation function; o_{mlp} is the packet feature vector.

3) Flow representation. The flow representation refers to the entire recurrent neural network [19] There are two aspects of the inputs to a recurrent model. One part is the output of MLP, o_{mlp}, and the other is the output of the recurrent layer from the last time step. The recurrent network is presented as follows:

$$o_{rnn,t} = \theta(W_{r\_i} \cdot o_{mlp} + W_{r\_h} \cdot o_{r,t-1} + b),$$

where W_{r_i} and W_{r_h} are the weights of the input layer and the recurrent layer, and the symbol b is the bias; $o_{r,t}$ is the output of the recurrent layer at the tth step.

4) Clustering layer. After obtaining the output $o_{rnn,t}$ from the RNN model, a parametric t-SNE method is adopted to cluster whose detail is discussed in Section 2.2.

2.2 Parametric T-SNE Model and Backward Propagation

In this part, we firstly introduce the t-SHE algorithm and obtain a gradient of cost function. Then the parametric t-SNE algorithm based on hierarchical neural network is discussed. The t-SNE algorithm consists of two steps. The first step is probability distribution in a high-dimensional space is performed. Accordingly, the more similar a pair of objects in the space are, the easier it is to be selected. Conversely, the probability of selecting two dissimilar objects is reduced. Further, the probability in a low-dimensional space is constructed, and the high-dimensional probability distribution is similar to the low-dimensional probability distribution. Different algorithms use different criteria to measure similarity distances (such as k-means using Euclidean distance). The t-SNE algorithm uses conditional probability to present the similarity distances of two objects. Specifically, when given a set $X = \{x1, x2, \ldots, xN\}$ containing N samples (objects), between any two samples xi and xj, the distance is defined as follows:

$$p_{ij} = \frac{p_{i|j} + p_{j|i}}{2N}$$

The definition of the probability condition between two samples is as follows:

$$p_{j|i} = \frac{\exp(-\frac{||x_i - x_j||^2}{2\sigma_i^2})}{\sum_{k \neq i} \exp(-\frac{||x_i - x_k||^2}{2\sigma_i^2})}$$

where σ_i^2 is the standard deviation of the Gaussian distribution of the data.

After dimensional reduction via t-SNE, the samples' set is presented as$Y = \{y1, y2, \ldots, yN\}$, which is the mapping from a high-dimensional space into a low-dimensional space. The distance q_{ij} between two samples in the low-dimensional space is presented as follows:

$$q_{ij} = \frac{(1 + \|y_i - y_j\|^2)^{-1}}{\sum_{k \neq l}(1 + \|y_k - y_l\|^2)^{-1}}$$

The final optimization of the t-SNE algorithm is minimizing Kullback-Leibler (KL) divergence, which is presented as follows:

$$C = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}$$

Generally, the values of pii and qii are 0. The minimization of the KL divergence is non-convex optimization; thus, the technique of mini-batch gradient descent is adopted, and the gradient is presented as follows:

$$\frac{\partial C}{\partial y_i} = 4 \sum_j (p_{ij} - q_{ij})(y_i - y_j)(1 + \|y_i - y_j\|^2)^{-1} \qquad (1)$$

So far the partial derivative about y_i are obtained and the method of t-SNE algorithm has been introduced.

In the model of parametric t-SNE, the symbol y_i refers to the output of the hierarchical neural network. The weights of neural networks are updated by back propagation. In this case, the weights of the model are presented as $W = \{w1, w2, \ldots, wK\}$, where K refers to the amount of weights from the neural network, and the gradient is presented as follows:

$$\frac{\partial C}{\partial w_j} = \frac{\partial C}{\partial Y}\frac{\partial Y}{\partial w_j}, j = 1, 2, \ldots, K,$$

where $\frac{\partial C}{\partial Y}$ can be calculated by Equation (1) and $\frac{\partial Y}{\partial w_j}$ can be calculated by back propagation.

2.3 Preprocessing

The micro-flow sequence is preprocessed. A micro-flow sequence, f, contains many ordered network packets, which can be presented as $f = \{p1, p2, \ldots, pm\}$. The length of a micro-flow sequence refers to the number of the packets it contains. Furthermore, the length of packets is also different between any two packets. Thus, the dimensions of the samples are inconsistent and not suitable as the input for the t-SNE algorithm. The preprocessing is described as follows:

1) Cutting and padding. To construct an equal-length micro-flow sequence, cutting and padding are involved. Preset each micro-flow sequence to contain m packets and each packet contains t bytes. Under known m and t conditions, the truncated network packets or micro-flow sequence can be cut. For a network packet or micro-flow that is too short, it needs to be padding with zero. The details of cutting and padding are presented in Algorithm 1.

2) Ignoring the address information. In network intrusion detection, the IP and Mac addresses are usually shielded to avoid interference from these messages in the detecting model.

3 Parametric T-SNE Based On Hierarchical Deep Neural Network

In this section, the experimental setups, evaluation metrics, effects of hyper parameters, and performance comparison of different algorithms are discussed.

3.1 Experimental Setup

In the experimental setup, two public datasets, DARPA 1998 [4] and ISCX-2012 [11], are involved. Additionally, a self-collected real network dataset without label is also employed in the experiment.

DARPA 1998 is a public dataset, which was sponsored by DARPA for the first realistic and systematic evaluation of research intrusion detection system, published by the MIT Lincoln Laboratory in the United State in 1998. This dataset contains a seven-week training set and a two-week test set. In this dataset, the traffic data contains four types of attacks, *i.e.*, DoS, Probe, U2R, and R2L. The percentage of attacks in the training set of DARPA 1998 is about 65.54%, while the proportion of attacks in DARPA is 63.29%, 1.99%, 0.26%, and 0.01%. The proportion of the test set is similar to that of the training set.

ISCX-2012 is a public network dataset published by the Information Security Centre of Excellence (ISCX) of the University of New Brunswick in Canada in 2012. This dataset contains the full network traffic data of seven days. All traffic data are normal on the first day, while four types of malicious traffics occurred in the following six days. The different typesof malicious traffic were BF-SSH, infiltrating, DDoS, and HttpDoS. The percentage of normality in ISCX-2012 is about 97.27%, while the proportion of attacks in ISCX-2012 is 0.46%, 0.66%, 0.23%, and 1.38% respectively.

The self-collected dataset contains data of seven days full traffic, which is collected by our self-developed network data acquisition equipment from a Chinese telecommunication operator. This dataset is without labels and plays a validated role in the experiments.

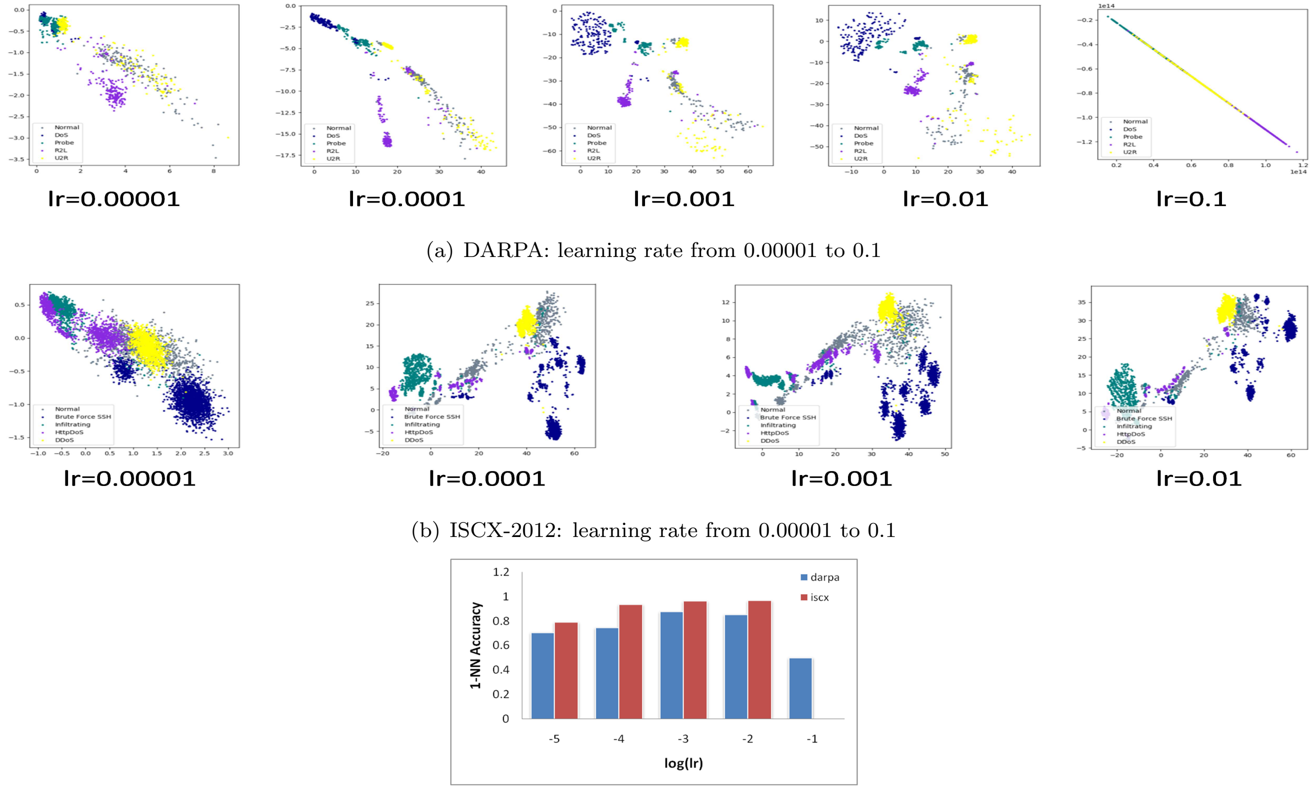
The experimental platform is Dell R720, which consists of a CPU of 16 cores with 2.7 GHz, 96 GB memory, and Nvidia Grid K2 GPU. The OS used is Ubuntu 14.04.

3.2 Evaluation Metrics

In this study, the 1-nearest neighbor (1-NN) algorithm is adopted, and the metrics for this are accuracy and recall. Accuracy is a description of systematic errors, a measure of statistical bias that represents the reliability of a rule, usually represented by the proportion of correct classifications. However, if some attacks are more important, the recall, which is the fraction of relevant instances that have been retrieved over the total amount of relevant instances,

Algorithm 1 Preprocessing the network flow

```
Input: micro-flow, presetting m and t
Output: preprocessed micro-flow
cnt $\leftarrow$ 0
if length_flow ¡ t then
    padding with (t-length_flow) packets, where the packets are filled by zero values
end if
for each packet in flow: do
    if cnt $\geq$ t then
        break
    end if
    if length_packet ¡ m: then
        padding with zero value at the end of the packet
    else
        if length_packet ¿ m: then
            cutting and only reserve the first m bytes
        end if
    end if
    cnt $\leftarrow$ cnt + 1
end for
```



(a) DARPA: learning rate from 0.00001 to 0.1

(b) ISCX-2012: learning rate from 0.00001 to 0.1

(c) The accuracy with learning rate

Figure 2: Two-dimensional (2D) dimensionality reduction effect under different learning rates

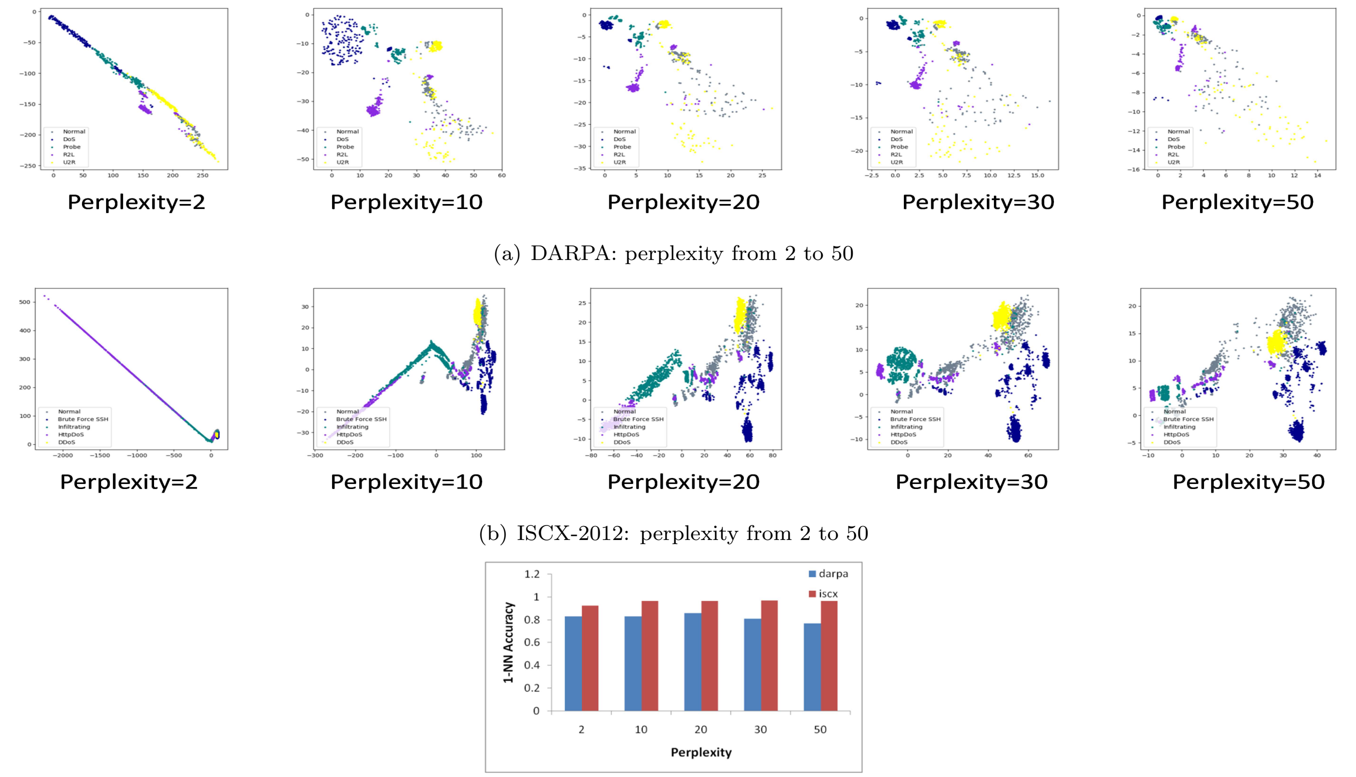
should be given more attention.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

$$Recall = \frac{TP}{TP + FN}$$

where TN is the number of instances correctly predicted as a non-attack instance. FN is the number of instances wrongly predicted as a non-attack instance. FP is the number of instances wrongly predicted as an attack. TP is the number of instances correctly predicted as an attack.

In this study, the 2D dimensionality reduction renderings are also involved. The visual renderings are not numerical indicators; however, they enable us to visually determine the effect of dimensionality reduction.



(a) DARPA: perplexity from 2 to 50

(b) ISCX-2012: perplexity from 2 to 50

(c) The accuracy with perplexity

Figure 3: 2D dimensionality reduction effect under different perplexity

3.3 Influence Of Hyper Parameters

There are two main types of hyper parameters that affect the performance of the parametric t- SNE algorithm. One consists of the inherent parameters of the algorithm, such as learning rate and perplexity. The other type consists of parameters of the preprocessing network data (the length of micro-flow and the size of packets).

1) Learning rate. The learning rate of the parametric t-SNE algorithm affects the speed at which the model converges in back propagation. There is an argument [20]that if the learning rate is too low, the distribution of samples tends to be spherical; Conversely, the model cannot converge.

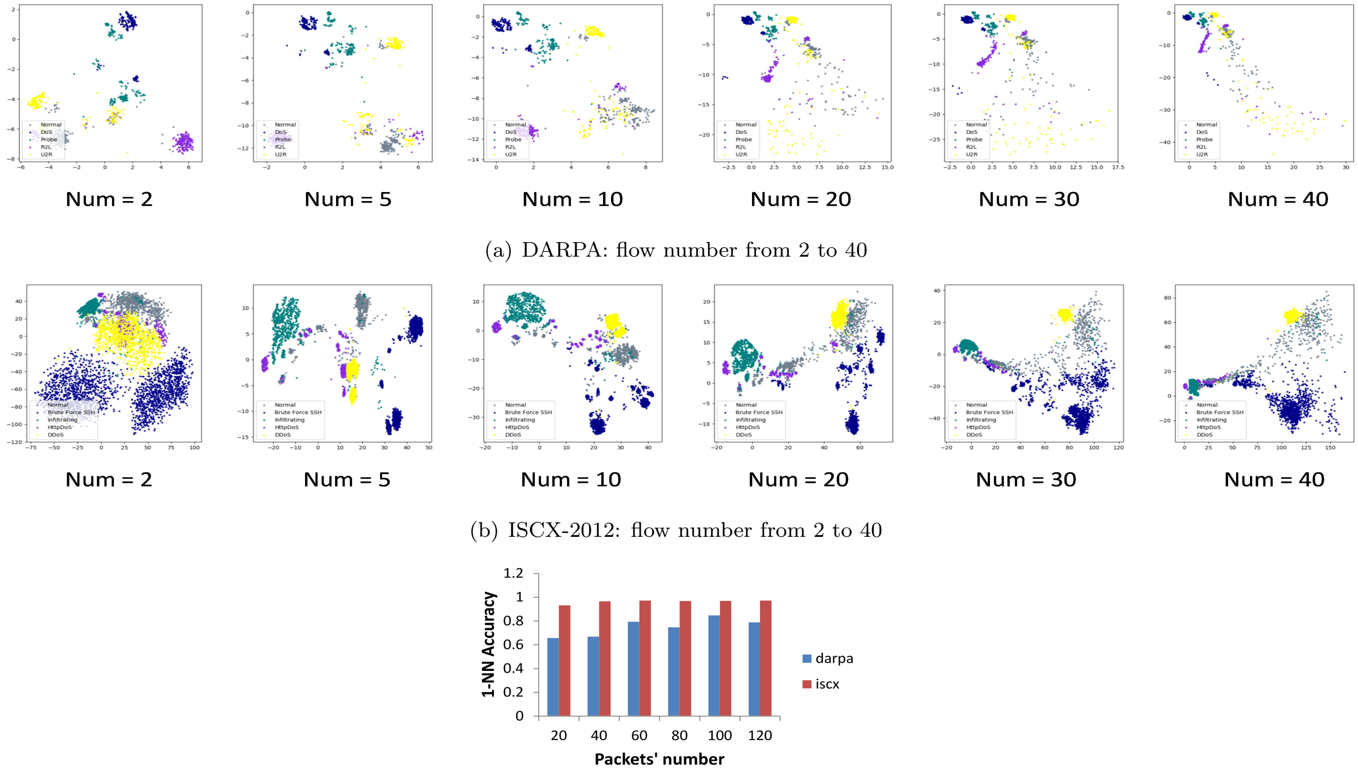
   In the experiments of learning rate, the range is from 0.0001 to 0.1. In Figure 2, the 2D dimensionality reduction renderings are shown. In the DARPA data set, the 1-NN error rate is the lowest between 0.001 and 0.01 learning rates, and too large or too small learning rates will increase the error rate. A similar phenomenon exists in the ISCX-2012 data set. However, the difference is that when the learning rate reaches 0.1, this model cannot converge, and the learning rate cannot be calculated.

2) Perplexity. The degree of perplexity is the number of nearest neighbors selected during the iterative update process. Generally, a larger sample set requires a higher degree of perplexity. It is highlighted in the literature [20]that the non-parametric t-SNE algorithm is less sensitive to the confusion parameter. However, it has been found through experiments that the parametric t-SNE algorithm is more sensitive to perplexity than the non-parametric t-SNE algorithm.

   In the experiments of perplexity, the range is from 2 to 50. It can be seen that the degree of perplexity is data sensitive. For the DARPA 1998 dataset, the degree of perplexity has a greater impact on the 1-NN accuracy; however, for the ISCX-2012 dataset, the degree of perplexity is less affected. The preliminary assumption is that in different datasets the manifold characteristics are not identical in high-dimensional data space. The 1-NN accuracy rate does not change too much; however, the effect of data's 2D reduction is significant. It can be clearly seen that when the perplexity is 2 or 10, the high-dimensional data is not effectively mapped into the latent space.

3) Length of micro-flow. The length of micro-flow refers to the number of packets that one flow contains. We use a five tuple (source IP, destination IP, source port, destination port, and time window) sequence of packets that indicate the micro-flow. The number of network packets included in each flow is not uniform; thus, cutting and padding is required.

   As shown in Figure 4, in the experiments of the length of micro-flow, the range is from 2 to 50. There is a phenomenon that the 1-NN error rate with short-



(a) DARPA: flow number from 2 to 40

(b) ISCX-2012: flow number from 2 to 40

(c) The accuracy with flow number

Figure 4: 2D dimensionality reduction effect under different length of micro-flow

length flow is lower than that of the longer one. A simple explanation is that irrespective of a normal or attack traffic, the first few network packets in a micro-flow are all connected packets, which cannot be detected as attacks.

4) length of packets. The size of a packet is the number of bytes a sampled IP packet contains. The short IP packets are padding. Conversely, long packets are cutting. Considering the importance of the header data of the IP packet, at least 60 bytes are reserved (The IP packet contains at least a 20 byte header. The TCP layer also contains at least a 20 byte header. Other application layer protocol data reserved 20 byte header).

The experiment of the length of packets is illustrated in Figure 5. The range of length is from 20 to 120 bytes. The best performance is achieved at the 100 byte length in both DARPA 1998 and ISCX-2012 datasets. In most cases, packets need to be 100 bytes long to contain enough information to be detected; however, packets with more than 100 bytes may cause excessive padding, and it involves too much noise.

3.4 Results Of Comparison Experiments

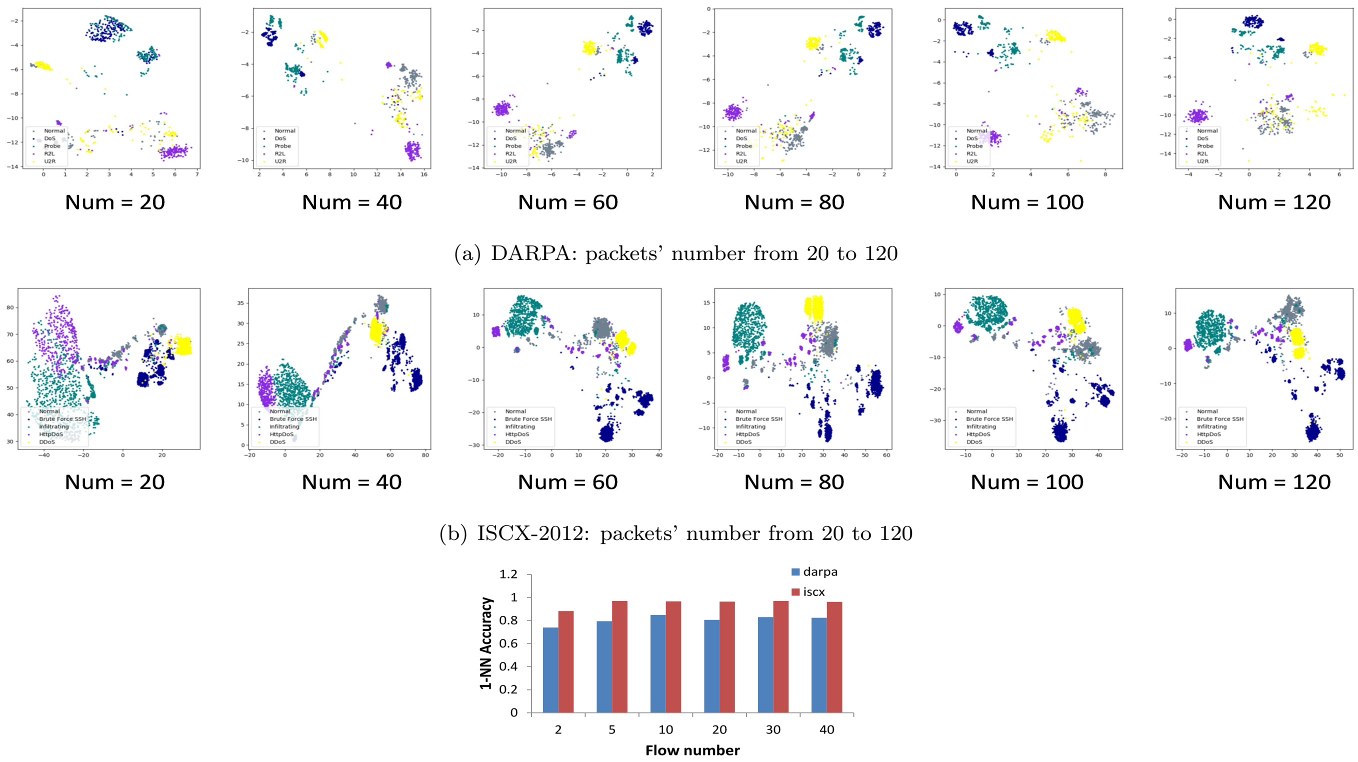
We set up a control experiment choosing PCA and auto-encoder as the control group.

The PCA algorithm is a typical linear dimensional reduction algorithm. The main idea of PCA is mapping data along the maximum direction of the variance makes the data easier to distinguish.

The auto-encoder was proposed by Hinton in 2006 [6], which is a deep neural network unsupervised algorithm. The core idea is that using multi-layer neural network makes input vectors closed to the output vectors. The structure of the auto-encoder is shown in Figure 6. The structure of the auto-encoder contains encoder and decoder, which are multi-layer neural network. The dimensional reduction is the output of the encoder.

The 2D dimensionality reduction renderings of DARAP1998, ISCX-2012 and the self-collected dataset are shown in Figure 6. In the subfigures of PCA, the sample points are scattered and have a certain clustering effect; however, the distinction between U2R, normal, and DoS is not obvious. In the subfigures of the auto-encoder, the sample points are scattered, and only R2L can be distinguished from other types. In the subfigures of parametric t-SNE, the effects of reduction are obvious.

In Figure 6(c), the performance of each reduction algorithms are shown based on the self-collected dataset. Differently from the other datasets, the data is untagged here, so the picture is with only one color to mark the network flow. It shows that by PCA and auto-encoder algorithms, samples have not been distinguished or reduce the dimension sensibly. However, sample points are effectively divided into 5 clusters via t-SNE algorithm.

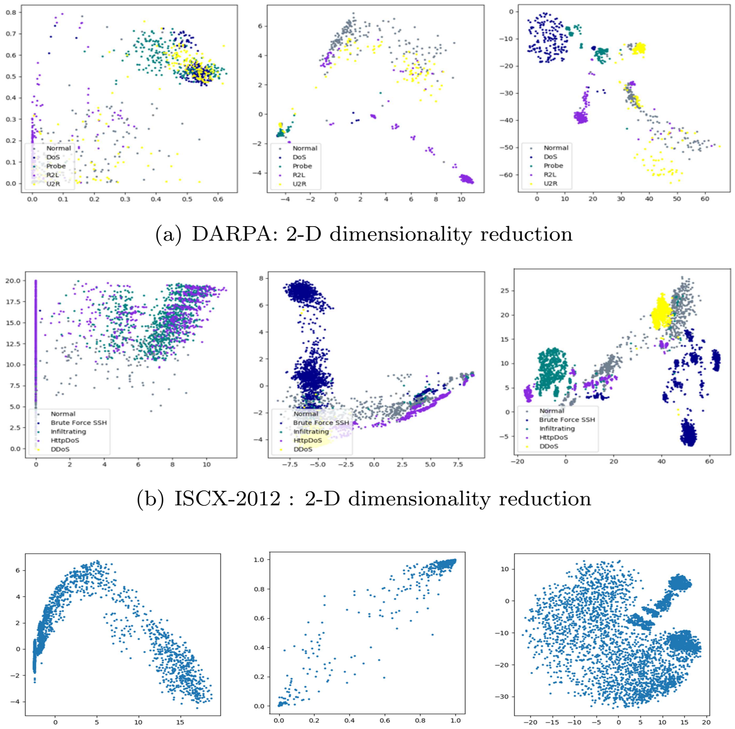


(a) DARPA: packets' number from 20 to 120

(b) ISCX-2012: packets' number from 20 to 120

(c) The accuracy with packets'number

Figure 5: 2D dimensionality reduction effect under different length of packets



(a) DARPA: 2-D dimensionality reduction

(b) ISCX-2012 : 2-D dimensionality reduction

(c) self-collected dataset: 2-D dimensionality reduction

Figure 6: 2D dimensionality reduction effect of PCA (left), auto-encoder (middle) and parametric t-SNE algorithm (right)

In Table 1, the accuracies and recalls of 2-D, 5-D, and 10-D dimensional reductions are presented. According

Table 1: 1-NN accuracy and recall rate for PCA, auto-encoder, and parametric t-SNE

Dimensions	2-D				5-D				10-D			
Datasets	DARPA		ISCX		DARPA		ISCX		DARPA		ISCX	
Metrics	Acc	Rc	Acc	Rc	Acc	Rc	Acc	Rc	Acc	Rc	Acc	Rc
PCA	0.784	0.83	0.831	0.854	0.823	0.819	0.988	0.986	0.832	0.831	0.985	0.982
Auto-encoder	0.411	0.654	0.361	0.517	0.684	0.774	0.678	0.698	0.826	0.856	0.875	0.863
t-SNE (RNN)	0.85	0.871	0.97	0.967	0.773	0.819	0.986	0.983	0.842	0.874	0.99	0.988
t-SNE (MLP)	0.819	0.862	0.975	0.972	0.808	0.858	0.962	0.957	0.819	0.869	0.949	0.945
t-SNE (RNN-MLP)	0.848	0.872	0.981	0.978	0.791	0.858	0.989	0.987	0.871	0.897	0.99	0.988

Table 2: The performance of different algorithms based on DARPA1998 dataset and ISCX-2012 dataset

Dataset	DARPA1998		ISCX-2012	
Algorithm	Accuracy	Avg-Rc	Accuracy	Avg-Rc
SVM [21]	79.4	47.6	N/A	N/A
Random forest [7]	91.4	78.23	N/A	N/A
Bayes network [13]	90.6	53.47	N/A	N/A
PLSSVM [2]	99.8	68.25	N/A	N/A
ALL-AGL [16]	N/A	N/A	95.4	93.2
AMGA2-NB [18]	N/A	N/A	94.5	92.7
t-SNE(RNN-MLP)	87.1	89.7	99.0	98.8

to the table, the parametric t-SNE method based on the RNN-MLP model is better than other algorithms. By comparing different dimensions, we can observe that as the dimension increases, both the 1-NN accuracy and 1-NN recall rate of the algorithm increase.

3.5 Algorrithm And Implementation Comparison

In this paper we also compare some algorithms based on the KDD99 dataset and ISCX-2012 dataset. It is noteworthy that most of the current studies are based on supervised method, and the t-SNE method we proposed are unsupervised. The methods of KNN, SVM, Tree and random forests, and Bayes are involved in Table 2.

As can be seen, for the DARPA1998 dataset, the t-SNE model performed well in terms of the accuracy rate and obtained a higher average recall than any other algorithms in Table 2. Our model were constructed via a recurrent neural network and t-SNE which takes the byte-level data (raw data) as inputs. It could be inferred that the recurrent model is suitable for streaming-type data.

According to the performance of difference algorithms based on ISCX-2012 dataset, we compared the t-SNE method with three supervised methods. In spite of the t-SNE algorithm is unsupervised, t-SNE method got a best accuracy rate and a second good recall rate.

4 Conclusions

- In this study, we are committed to developing an unsupervised dimensionality reduction method for facing network attacks and have proposed a parametric t-SNE method based on a hierarchical neural network. Furthermore, a data preprocessing method adapted to the parametric t-SNE algorithm for the indefinite-length network data is discussed.
- In the experiments, the proposed method achieves better results of dimensional reduction than other algorithms.
- In future works, we aim to investigate how to introduce geographic information, such as IP addresses, as input data. Furthermore, we aim to investigate other unsupervised reduction or clustering methods for network attacks.

References

[1] M. Z. Alom, V. Bontupalli, and T. M. Taha, ”Intrusion detection using deep belief networks,” in *National Aerospace and Electronics Conference (NAECON'15)*, pp. 339–344, 2015.

[2] F. Amiri, M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, ”Mutual information-based feature selection for intrusion detection systems,” *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1184–1199, 2011.

[3] L. V. Der Maaten and G. Hinton, "Visualizing data using t-SNE," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[4] A. Fischer and C. Igel, "An introduction to restricted boltzmann machines," in *Iberoamerican Congress on Pattern Recognition*, pp. 14–36, 2012.

[5] Y. Gal and Z. Ghahramani, "A theoretically grounded application of dropout in recurrent neural networks," in *Advances in Neural Information Processing Systems*, pp. 1019–1027, 2016.

[6] J. Goldberger, G. E. Hinton, S. T. Roweis, and R. R. Salakhutdinov, "Neighbourhood components analysis," in *Advances in Neural Information Processing Systems*, pp. 513–520, 2005.

[7] M. A. M. Hasan, M. Nasser, B. Pal, and S. Ahmad, "Support vector machine and random forest modeling for intrusion detection system (IDS)," *Journal of Intelligent Learning Systems and Applications*, vol. 6, no. 1, pp. 45, 2014.

[8] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[9] T. Iwata, K. Saito, N. Ueda, S. Stromsten, T. L. Griffiths, and J. B. Tenenbaum, "Parametric embedding for class visualization," in *Advances in Neural Information Processing Systems*, pp. 617–624, 2005.

[10] R. Jozefowicz, O. Vinyals, M. Schuster, N. Shazeer, and Y. Wu, "Exploring the limits of language modeling," *Computation and Language*, 2016. arXiv Preprint arXiv:1602.02410

[11] R. Lippmann, R. K. Cunningham, D. J. Fried, I. Graf, K. R. Kendall, S. E. Webster, and M. A. Zissman, "Results of the darpa 1998 offline intrusion detection evaluation," in *Recent Advances in Intrusion Detection*, vol. 99, pp. 829–835, 1999.

[12] L. V. D. Maaten, "Learning a parametric embedding by preserving local structure," in *Artificial Intelligence and Statistics*, pp. 384–391, 2009.

[13] H. A. Nguyen and D. Choi, "Application of data mining to network intrusion detection: Classifier selection model," in *Asia-Pacific Network Operations and Management Symposium*, pp. 399–408, 2008.

[14] T. L. Pao, W. Y. Liao, Y. T. Chen, and T. N. Wu, "Mandarin audio-visual speech recognition with effects to the noise and emotion," *International Journal of Innovative Computing, Information and Control (IJICIC'10)*, vol. 6, no. 2, pp. 711–724, 2010.

[15] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[16] H. Sallay, A. Ammar, M. B. Saad, and S. Bourouis, "A real time adaptive intrusion detection alert classifier for high speed networks," in *IEEE 12th International Symposium on Network Computing and Applications*, pp. 73–80, 2013.

[17] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Computers & Security*, vol. 31, no. 3, pp. 357–374, 2012.

[18] Z. Tan, A. Jamdagni, X. He, P. Nanda, R. P. Liu, and J. Hu, "Detection of denial-of-service attacks based on computer vision techniques," *IEEE Transactions on Computers*, vol. 64, no. 9, pp. 2519–2533, 2014.

[19] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu, "HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2017.

[20] K. Q. Weinberger, F. Sha, and L. K. Saul, "Learning a kernel matrix for nonlinear dimensionality reduction," in *Proceedings of the Twenty-first International Conference on Machine Learning*, pp. 106, 2004.

[21] X. Xu, "Adaptive intrusion detection based on machine learning: Feature extraction, classifier construction and sequential pattern prediction," *International Journal of Web Services Practices*, vol. 2, no. 1-2, pp. 49–58, 2006.

[22] E. Zyad, C. Khalid, and B. Mohammed, "Improving network intrusion detection using geometric mean LDA," *International Journal of Network Security*, vol. 20, no. 5, pp. 820–826, 2018.

Biography

Huijun Yao received his master's degree from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2007. He began to work in Jiangsu broadcasting cable information network, in 2007. He is now an experienced researcher of technology Institute of Jiangsu broadcasting cable information network, Nanjing, China. His research interests are in the fields of cable television, digital Set-top Box, 5th-Generation.

Peng Sun received his master's degree from Northwestern Polytechnical University, Xi'an, China, in 2011 and his doctoral degree from the Institute of Acoustics, Chinese Academy of Sciences, Beijing, China, in 2014. He began to work in Institute of Acoustics, in 2014. He is currently a researcher of National Network New Media Engineering Research Center, Chinese Academy of Sciences, Beijing, China. His research interests are in network media, digital signal processing, smart terminal..

Chaopeng Li received his PhD degree in National Network New Media Engineering Research Center, Chinese Academy of Sciences, Beijing, China, in 2019 and received his Bachelor's degree from Beijing University of Post and Telecommunications, Beijing, China, in 2013. He currently works at Jimei University. His current research interests are in deep learning, network intrusion detection and information processing.