A Research of Network Applications Classification Based on Deep Learning

Hong Shao, Liujun Tang, Ligang Dong(✉), Long Chen, Xian Jiang, and Weiming Wang

School of Information and Electronic Engineering, Zhejiang Gongshang University, Hangzhou 310018, China
1564027103@qq.com, tlj2016@126.com, {donglg, jiangxian, wmwang}@zjgsu.edu.cn, smllchuju@163.com

Abstract. Nowadays, the huge traffic generated by a growing number of network applications occupies enormous network bandwidth and increases the burden of network management. The ability to identify and categorize network applications accurately is crucial for learning network traffic conditions, finding people's online behavior and accelerating the development of the Internet. The prior traffic classification methods often have unstable recognition rate and high computational complexity, which affects the network traffic management and application categories monitoring. Therefore, this paper proposes a method of using the deep learning technology to classify network applications. First, we propose a network application classification model based on Deep Belief Network (DBN). Then we construct a DBN-based model suitable for network applications classification with the Tensorflow framework. Finally, the classification performances of this DBN-based model and the BP-based model are compared on the real data sets. The experimental results show that the applications classification model based on DBN has higher classification accuracy for P2P applications.

Keywords: Deep learning · Deep belief network
Network applications classification

1 Introduction

According to the Visual Networking Index (VNI) report, from 2016 to 2021, global IP traffic is likely to triple, with an average annual increase of 1.2 ZB to 3.3 ZB. The majority of IP traffic is from P2P network applications. The increase of P2P network applications will inevitably generate huge traffic and occupy tremendous network bandwidth, which will exacerbate the problem of network congestion and eventually aggravate the burden of network management. Being able to identify and categorize network applications accurately is extremely important for learning the state of network traffic and accelerating the development of the internet.

The existing methods for P2P applications classification have certain limitations. Even the learning-based applications classification method, with better classification

L. Meng and Y. Zhang (Eds.): MLICOM 2018, LNICST 251, pp. 13–21, 2018.
https://doi.org/10.1007/978-3-030-00557-3_2

performance, has unstable recognition rate and high computational complexity, which affect the management of network traffic and the monitoring of application categories.

Aiming at the deficiency of existing network applications classification technologies, and in order to learn the characteristics of network traffic data fully and improve the accuracy of network application classification, this paper proposes a network application classification model based on Deep Belief Networks (DBN). The original DBN model was proposed by Hinton [3] which can not only realize the automatic learning of characteristics but also learn the essential features that characterize the data and overcome the difficulty in the training through the method of layer-by-layer initialization. The utilization of DBN technique for characteristic classification and recognition has obvious advantages.

The remainder of this paper is structured as follows: Sect. 2 reviews the studies on network applications classification. Section 3 uses the Tensorflow framework to build the DBN model, including the construction of data sets and the determination of model parameters. Section 4 uses the DBN model to learn the traffic characteristic based on the constructed train data sets; then utilizes test datasets to analyze the classification accuracy of this model; and compares the result with that of the BP-based model at the same condition. The last chapter is the summary of this article.

2 Related Research

In view of the rapid growth of network traffic in the future, the majority of traffic is from the P2P applications. Recently, the most popular network classifications method is the method based on machine learning [6]. In 2005, Zuev et al. [9]. utilized the naive Bayesian method to extract the network traffic characteristics for training, but the classification accuracy of this method was only about 60%. Subsequently, Huang et al. [5] used KNN (k-Nearest Neighbor) algorithm to conduct experimental research on network traffic classification, and the classification accuracy can reach 90%. However, once the data packets are coming, all the streams in the train set will be calculated, so that the performance of classification is poor. In 2009, Xu et al. [1] used the C4.5 decision tree to classify network traffic, and the classification accuracy could reach 94%. However, the C4.5 decision tree classification method needs more traffic characteristics and data groups, and has high computational complexity, which hinders the further research. In 2015, Hong et al. [8] utilized the SVM algorithm to classify the P2P traffic, and the accuracy of that is only about 80%.

According to the demonstration above, this paper proposes a method of using the deep learning technology to construct a DBN model which is suitable for network application classification.

3 Network Applications Classification Model Based on DBN

In order to classify the existing network applications accurately, this paper proposes the network application classification model based on DBN. This section selects a framework to build a DBN-based model for applications classification. First, we use the

TensorFlow framework to initialize a DBN model. Second, we use the pre-process datasets to make it suitable for applications classification. Furthermore, we determine the number of hidden nodes and hidden layers of this DBN-based model through experiments. Then, we use the training datasets to train the DBN-based model until this model with better parameters, and the process of which include the unsupervised training [11] and the supervised training [12]. Finally, utilize the test dataset to evaluate the classified effect of the DBN-based model.

3.1 Data Pre-processing

The public dataset provided by LiWei et al. [4] is the only available dataset for UDP traffic that contains traffic data from P2P applications and non-P2P applications. Therefore, we select the UDP public data set provided by LiWei et al. as the experimental dataset. The dataset includes the feature sets and tag sets. The feature sets contain 9 kinds of features extracted from the data stream such as the port number, the stream size. The tag sets involve 6 applications like P2P. The feature sets and tag sets are presented in Tables 1 and 2.

Table 1. Features of the dataset

Source port number	Destination port number	Total number of packets (bidirectional)
The minimum packet size (Client -> server)	The minimum packet size (Server side -> client side)	Client sends the first packet size (after receiving the server to return data)
The maximum packet size (Client -> server)	The maximum packet size (Server side -> client side)	The maximum number of consecutive packets the client sends to the server

Table 2. Applications of the label set

P2P	Services	Attack
Multimedia	VOIP	Game

The initial DBN model is built by the Tensorflow framework, and the appropriate number of hidden layers and hidden layer nodes are determined by datasets. Before the experiment, we need to standardize those two datasets mentioned above.

1. Feature normalization

The values of nine features in above datasets are integers among 0 and 65535. The mean and variance of each feature are different, therefore, all the input features are normalized to the range of [0, 1] with Eq. (1), in which x_i represents the original network traffic data, and x_{max} and x_{min} represent the maximum and minimum traffic respectively.

$$X_i' = \frac{X_i - X_{min}}{X_{max} - X_{min}} \tag{1}$$

2. Label coding

When the DBN model is applied to the classification study, the Softmax regression model [9] will be adopted at the final output layer of the model. The Softmax model classifies different objects by assigning probabilities. A neuron node in the model output layer corresponds to a label type. Label sets are expressed by one-hot encoding [10] as shown in Table 3 below.

Table 3. Dataset encoding

Label	Attack	Services	P2P	Multimedia	VOIP	Game
Number	1	2	3	4	5	6
Coding	000001	000010	000100	001000	010000	100000

The public data sets provided by LiWei et al. have 774141 streams, including 54,659 streams from P2P applications accounting for 7.1%. The dataset is divided into 10 sub-datasets (i.e., Dataset1–Dataset10). Among, the Dataset1 and Dataset2 which contain labels are as unsupervised training datasets; the Dataset3 which contain labels are as used in the supervised fine-tuning phase; the Dataset4–Dataset10 which contain labels are as test datasets. The number of streams contained in every dataset and that of P2P traffic are shown in Table 4 below.

Table 4. Details of sub-datasets

Sub-datasets	1	2	3	4	5	6	7	8	9	10
P2P	3567	4857	3632	4431	3949	3656	4472	9339	8472	8606
Total	60000	60000	60000	60000	60000	60000	60000	120000	120000	114013

3.2 Determination of the Model Parameters

Before the unsupervised training of DBN-based model, we need to determine the number of hidden nodes and hidden layers. The number of hidden nodes will affect the model on the abstract expression of features and the number of hidden layers will be directly related to the depth of the DBN-based model. And the increase of the number of hidden layers is conducive to a more comprehensive study on the characteristics. Since the problem of taking the number of different nodes and different hidden layers into account together is complicated, this section will firstly determine the number of hidden nodes when the model contains two hidden layers. When the number of hidden nodes is decided, we will determine the appropriate number of hidden layers.

Determination of the Number of Hidden Nodes

Based on the first step, we can obtain the range of the number of hidden nodes with the formula 2. After several times of experimental comparison about the classification effect among models which vary in the number of hidden nodes, we can eventually acquire the number of hidden nodes corresponding to the model with the best classification performance; and put it as the numerical value for later experiment.

We select the DBN-based model only containing 2 hidden layers as initial model. We get the range of n is [1, 15] based on the calculation. In order to perform fully comparable experiment, we selected another 6 points, which is 16, 20, 21, 22, 26, 30.

$$n = \sqrt{m + p} + a \tag{2}$$

In the equation above, "m" represents the number of input feature; "p" stands for the number of output label; "n" is the number of hidden layer nodes, and a symbolizes an integer within [1, 10].

When there is difference in the number of different hidden layer nodes, the overall classification accuracy of the model will be variant as shown in Fig. 1 below. The sum of the unsupervised and supervised training time of the model is used as the total training time of the model. The total training time of the models containing different number of hidden nodes is compared and shown in Fig. 2 below.

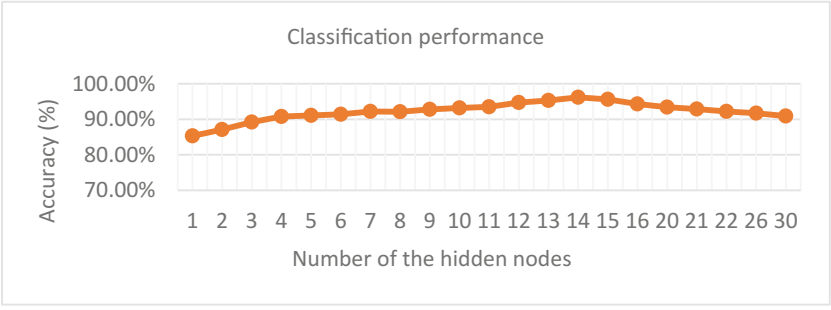

Fig. 1. The classification performance of the models with different number of hidden nodes

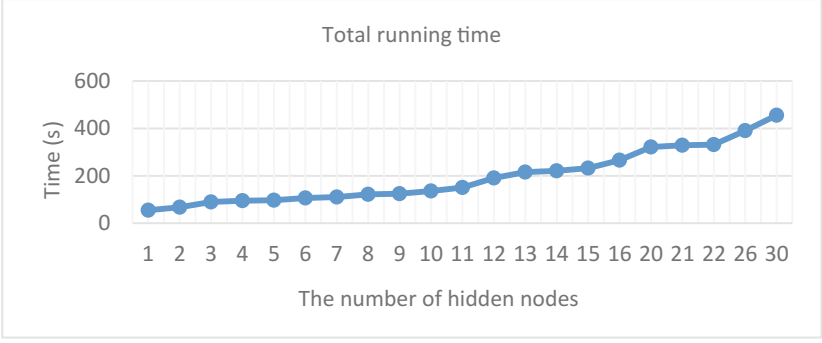

Fig. 2. Total running time of the models corresponding to the different number of hidden nodes

As can be seen from Fig. 1, with the increase of the number of hidden nodes, the classification accuracy of the model grows roughly. When n = 14, the classification accuracy of the model reach in 96%. From Fig. 2, we can learn that the total running time of the model generally shows an upward trend along with the growth of the number of hidden nodes. When n = 14, the total running time of the model is about 221 s, which is a bit higher than that of the nearby hidden nodes, but the classification accuracy at this time is the highest. Therefore, the number of hidden layer nodes in the DBN model is set to 14 preparing for subsequent experiments.

Determination of the Number of Hidden Layers

The number of hidden layers will directly affect the depth of the DBN model. The researchers [7] proved that the accuracy of the classification can be further improved by increasing the number of hidden layers of the DBN model to improve the abstraction ability of the data features. But not the more the number of hidden layers is, the better the classification effect [2] will be.

According to the analysis before, the number of hidden nodes of the model is set to 14. We will train the model respectively with different number of hidden layers, from 1 to 6. By having train on DBN-based models containing different number of hidden layers, we can obtain a group of corresponding DBN-based models. Then, we use the test datasets-Dataset4–Dataset10 to test the classification ability of those DBN models. The experimental results of DBN models with different number of hidden layers are shown in Table 5.

Table 5. The performance of the DBN-based models with different number of hidden layers

The number of hidden layers	Classification accuracy	Running time (s)
1	90.2%	221.9
2	95.1%	335.1
3	95.4%	546.8
4	96.9%	601.2
5	94.7%	693.8
6	93.6%	842.5

As what's shown in Table 5, the total running time of the models increases with the growth of the number of hidden layers, but the classification accuracy does not always increase. When the number of hidden layers is 4, the classification accuracy of the model reaches in 97%, and the total running time at this time is 601.2 s. Therefore, we choose to construct a DBN-based model with 4 hidden layers.

Through the analysis of many experimental results, this paper will construct a DBN-based model that contains 4 hidden layers and every hidden layer contains 14 hidden nodes.

3.3 Training Phase of the Model

The process of training the constructed DBN-based model is as follows. The first step is the unsupervised training with unlabeled-datasets based on the initial model. Then Dataset3 (including tag) are used to adjust the model. That is, the weights trained in the unsupervised phase are transferred to the BP neural network for fine-tune of the DBN-based model. As a result, a fully trained DBN-based model is obtained preparing for the later evaluation.

4 Performance Evaluation

4.1 Experimental Environment

The experimental platform for this paper is an Intel Core i5 processor with a 3.3 GHz, 14.0 GB memory on a HP computer running window 10 (64 bit) operating system. This paper uses the TensorFlow framework to build the DBN-based model, in which all DBN-based algorithms are implemented in Python language.

Specific version of the software tools used herein is Tensorflow 1.2.1, Python 3.5.1.

4.2 Performance Comparison

In this section, we will compare the classification performance of the DBN-based model with that of the BP-based model. To make the comparison more accurately, we utilized the same method demonstrated in Sect. 3.2 to build a BP-based model in the same condition, including dataset and building process. Through several experiments, we finally construct a BP-based model with two hidden layers and every hidden layer contains 14 hidden nodes.

For the DBN-based model and BP-based model trained by public datasets, Dataset4–Dataset10 are utilized as the test datasets to test the performance of two models respectively. The comparison results of classification precision of the two models are presented in Fig. 3.

It can be seen from the figure above that due to the different distribution of network applications in each data set, the classification results of the same model are different. For the comparison between the classification precision of P2P application in Fig. 3, every precision value of the DBN-based model is higher than that of BP-based model, and the precision of DBN-based model is up to 98%.

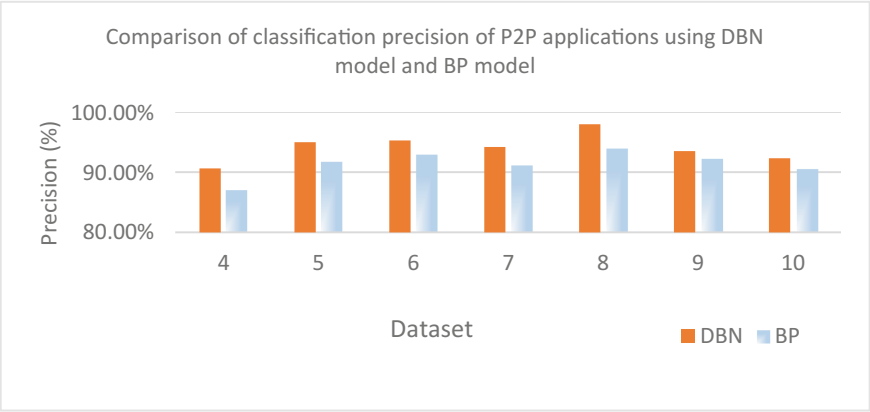

Fig. 3. Comparison of classification precision of P2P applications

5 Summary

In order to improve the classification accuracy of network applications and resolve the deficiency of existing network applications classification technology, this paper proposes a network applications classification model based on DBN. This model improves the accuracy of traditional network applications classification models. To make it more persuasive, we compare the classification results between DBN-based model and BP-based model in the same condition. Finally, we found the network application classification model based DBN has higher classification accuracy.

Acknowledgement. This work was supported by a grant from the Key Research and Development Program of Zhejiang (No. 2017C03058), Zhejiang Provincial Key Laboratory of New Network Standards and Technologies (NNST) (No. 2013E10012).

References

1. Xu, P., Lin, S.: A method to classify network traffic with the C4.5 decision tree. Chin. J. Comput. **20**(10), 2692–2704 (2009)
2. Yu, K., Jia, L., Chen, Y., et al.: The yesterday, today and tomorrow of deep learning. J. Comput. Res. Dev. **50**(9), 1799–1804 (2013)
3. Hinton, G., Salakhutdinov, R.: Reducing the dimensionality of data with neural networks. Science **313**(5786), 504–507 (2006)
4. http://www.tensorfly.cn/tfdoc/tutorials/mnist_beginners.html
5. Internet assigned numbers authority [EB/OL] (2008). http://www.iana.org
6. Lu, G., Zhang, H.L., Ye, L.: P2P traffic identification. J. Softw. **22**(6), 1281–1298 (2011)
7. Ruijuan, Z., Jing, C., Mingchuan, Z., et al.: User abnormal behavior analysis based on neural network clustering. J. China Univ. Posts Telecommun. **23**(3), 29–44 (2016)
8. Wang, D., Zhang, L., Yuan, Z., et al.: Characterizing application behaviors for classifying P2P traffic. In: International Conference on Computing, Networking and Communications, pp. 21–25. IEEE (2014)

9. Zuev, D., Moore, A.W.: Traffic classification using a statistical approach. In: Dovrolis, C. (ed.) PAM 2005. LNCS, vol. 3431, pp. 321–324. Springer, Heidelberg (2005). https://doi.org/10.1007/978-3-540-31966-5_25
10. Wang, X., Li, Y.: The Introduction and Improvement of EDA, p. 225. Xidian University Press, Xi'an (2005)
11. Le, Q.V.: Building high-level features using large scale unsupervised learning. IEEE (2013)
12. Oravec, M., Podhradsky, P.: Medical image compression by backpropagation neural network and discrete orthogonal transforms. WIT Trans. Biomed. Health **4** (1970)