NSA-Net: A NetFlow Sequence Attention Network for Virtual Private Network Traffic Detection

Peipei Fu[1,2], Chang Liu[1,2], Qingya Yang[1,2], Zhenzhen Li[1,2], Gaopeng Gou[1,2], Gang Xiong[1,2], and Zhen Li[1,2(✉)]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
lizhen@iie.ac.cn

[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China

Abstract. With the increasing attention on communication security, Virtual private network(VPN) technology is widely used to meet different security requirements. VPN traffic detection and classification have become an increasingly important and practical task in network security management. Although a lot of efforts have been made for VPN detection, existing methods mostly extract or learn features from the raw traffic manually. Manual-designed features are often complicated, costly, and time-consuming. And, handling the raw traffic throughout the communication process may lead to the compromise of user privacy. In this paper, we apply bidirectional LSTM network with attention mechanism to the VPN traffic detection problem and propose a model named NetFlow Sequence Attention Network (NSA-Net). The NSA-Net model learns representative features from the NetFlow sequences rather than the raw traffic to ensure the user privacy. Moreover, we adopt the attention mechanism, which can automatically focus on the information that has a decisive effect on detection. We verify our NSA-Net model on the NetFlow data generated from the public ISCXVPN2016 traffic dataset. And the experiment results indicate that our model can detect VPN from non-VPN traffic accurately, and achieve about 98.7% TPR. Furthermore, we analyze the performance of our model in the presence of sampling and our model still achieves over 90% TPR and Accuracy at low sampling rates.

Keywords: NetFlow data · Vpn detection · Attention mechanism · Deep learning · Network security

1 Introduction

A virtual private network(VPN) [1] is a virtual network, which establishes secure and encrypted connections to help ensure that sensitive data is safely transmitted. At present, with the increasing emphasis on communication security, VPN technology is widely used in network communications to meet different security requirements. However, with the widespread application of VPN technology,

Z. Huang et al. (Eds.): WISE 2020, LNCS 12342, pp. 430–444, 2020.
https://doi.org/10.1007/978-3-030-62005-9_31

they also bring some challenges to network security and management. On the one hand, VPN is easy to be exploited by attackers or hackers to hide their malicious behaviors, making them difficult to detect [7]. On the other hand, VPN adopts tunneling protocols and encryption techniques, which make it difficult to detect VPN traffic from other encrypted non-VPN traffic [8]. In addtion, encrypted VPN traffic detection is also a huge challenge to traditional port-based and rule-based methods. Therefore, how to effectively identify VPN traffic has already become an increasingly important and practical task in network management and cyberspace security.

At present, VPN traffic detection attracts the widespread attention of academia [3–8]. Currently, the typical methods are based on machine learning [3–5] or deep learning [6–8]. The machine learning methods generally need to select effective features to detect VPN traffic. However, the features are often extracted artificially, which depends on professional experience heavily. And, these methods have a great dependence on these features. Once the features have changed, the model will fail.. In order to reduce the cost of manually constructing features, deep learning is gradually applied, which could learning features automatically. Although current deep learning models have achieved excellent results, they all incline to use the raw encrypted traffic or information from the raw encrypted traffic as input to learn features. It will inevitably lead to huge input and time-consuming problems of the model. At the same time, capturing and using raw traffic can also cause user privacy problems to some extent. To solve these problems, an alternative approach that can be taken into account is to use the NetFow data [9], which is proposed by Cisco and only contains session-level statistical information. Compared with state-of-the-art methods, little research has been found for VPN detection using NetFlow data.

In this paper, we propose a model named NetFlow Sequence Attention Network (NSA-Net) for VPN traffic classification. The NSA-Net learns representative features from the NetFlow sequences rather than the raw traffic to ensure the user privacy. As bidirectional LSTM is able to exploit sequential information from both directions and attention mechanism is able to focus on the important information, the bidirectional LSTM (bi-LSTM) network with attention mechanism [10] is adopted to capture the most important representative information in a NetFlow sequence. Specifically, the NSA-Net model includes four layers: input layer to input the NetFlow sequence, bi-LSTM layer, and attention layer to generate the features, output layer to output the predicted labels. The features for detection are learned automatically from the raw NetFlow sequences by the bi-LSTM layer and are boosted by the attention layer.

Our contributions can be briefly summarized as follows:

– We propose a NSA-Net model for VPN traffic detection. And, to our best knowledge, in the field of VPN detection, it is the first attempt that an attention mechanism is used to obtain features, and the NetFlow sequence is used as input to protect user privacy.

- The optimal NetFlow inputs that can discriminate VPN from non-VPN are demonstrated. Meanwhile, we improve the performances of our model by adjusting the class_weight parameter to solve the problem of data imbalance.
- Our NSA-Net produces excellent results on the NetFlow data generated from the public VPN-nonVPN traffic dataset (ISCXVPN2016) [4], outperforming other deep learning models and the state-of-the-art method. Even at low sampling rates, our model still works very well.

The rest of the paper is organized as follows. Section 2 summarizes the related work. The methodology is elaborated in Sect. 3, and the experiments are presented in Sect. 4. Finally, we conclude this paper in Sect. 5.

2 Related Work

In this section, we present some current related work in the area of VPN detection, NetFlow application, as well as the previous works on attention mechanism.

2.1 VPN Detection

Over the years, a great deal of researches have focused on how to detect VPN traffic on the Internet. Pattern recognition and machine learning are proposed to solve VPN detection problem. Abideen M.Z. et al. proposed a lightweight approach to detect VPN activity by extracting features from plain information [3]. Gerard D. et al. studied the effectiveness of flow-based time-related features with C4.5 and KNN to detect VPN traffic [4]. However, the feature selection of these methods requires rich experiences and lots of human effort.

Nowadays, since deep learning can learn features automatically, researchers pay more attention to the methods based on deep learning [6–8]. However, these methods mostly tend to learn features from: the raw traffic [6], or statistical information extracted from raw traffic [7], or image converted from raw traffic [8]. Learning from raw traffic could improve the results to a certain degree. But it also will lead to huge input and time-consuming problems. Especially, it may invade users' privacy, as the raw traffic data often contains sensitive information about network users.

2.2 NetFlow Application

NetFlow represents a high level summary of network conversations, which is an alternative to the raw traffic. With the development of NetFlow technology, the protocol is extremely attractive to academic research and has been widely used in many research fields [11–14], especially in the field of anomaly detection, traffic classification, network measurement and network monitoring. In recent years, NetFlow has also been gradually applied to the field of deep learning. Liu et al. [15] presented a method to detect network attacks and intrusions with CNN by constructing images from NetFlow data. Yang et al. [16] proposed RNN deep learning method to analyze NetFlow information for attack detection.

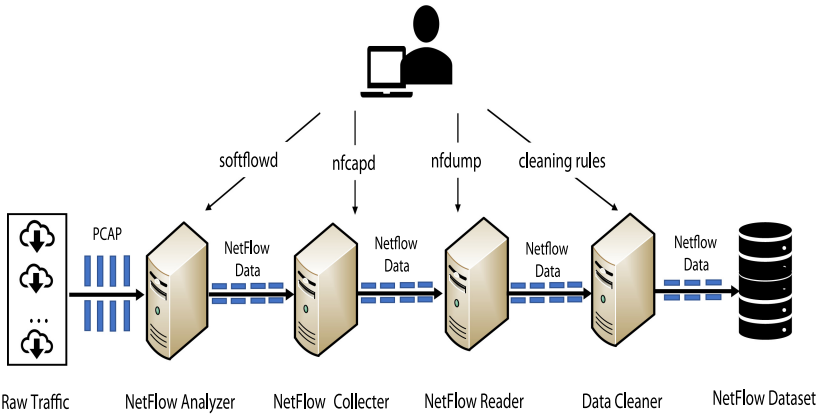

Fig. 1. The data preprocessing process of our work.

Although research on NetFlow has already very abundant, most of them are still based on machine learning, and very little is based on deep learning. Therefore, the application of deep learning on NetFlow is still in its early stage and needs to be explored more deeply.

2.3 Attention Mechanism

In recent years, the study of deep learning has become more and more in-depth, and there have been many breakthroughs in various fields. Neural networks based on attention mechanisms have become a recent hot topic in neural network research. Attention-based neural networks have recently been successful in a wide range of tasks [17–20], especially in image classification and natural language processing.

To the best of our knowledge, we have not found a deep learning algorithm based on the attention mechanism applied to NetFlow data for VPN detection. Therefore, in this paper, we explore the RNN with an Attention mechanism to learn NetFlow sequence features for VPN detection.

3 Methodology

In this work, we develop a NetFlow Sequence Attention Network, called NSA-Net, which based on NetFlow data for VPN detection tasks. In this section, the data preprocessing and the architecture of the proposed NSA-Net will be presented in the following.

3.1 Data Preprocessing

Before training the NSA-Net model, we need to prepare the NetFlow dataset to feed into the model. As it is very hard to label the real-world NetFlow data

accurately, to facilitate research and comparison, we consider using the NetFlow data generated from the manual-captured raw traffic or public raw traffic. Our data preprocessing process is shown in Fig. 1.

In order to generate the NetFlow data from the raw traffic, the following software solutions are used: softflowd [21] as the NetFlow analyzer, nfcapd [22] as the NetFlow collector, and nfdump [22] as the NetFlow reader. Softflowd is a well-known and popular open-source tool that allows exporting data in Cisco NetFlow format. It could read the raw pcap files, calculate statistics of flow on its expiry and send them to a network card. Nfcapd reads the generated NetFlow data from the network card and stores them into files. Nfdump reads the NetFlow data from the files stored by nfcapd and displays them in a user-defined format. In addition, nfdump can further process NetFlow data to provide different data output formats.

Since the raw traffic generally is captured in a real-world emulation, it may contain some irrelevant packets that we don't care about. And the NetFlow data is transferred from the raw traffic, there are inevitably some outliers and noise in the data. Therefore, before using these data, data cleansing process is required. First, some irrelevant service data should be discarded, such as DNS service(port 53), which is used for hostname resolution and not relevant to either application identification or traffic characterization. Therefore, the NetFlow sequence based on port 53 is omitted from the dataset. Second, sequences with one-way flows and few packets are deleted. With no data interaction between the two sides of the communication, and the number of one-way packets is so small that such data sequence is difficult to characterize the traffic.

Finally, after processing the raw traffic, we collect NetFlow data sequences, including flow packets, flow bytes, flow duration, protocol, TCP flag and so on. To evaluate the NSA-Net model comprehensively and to explore which NetFlow data sequences work better, we generate four kinds of NetFlow data sequences at different sampling rates, which includes the unidirectional original NetFlow data sequence (U_ONData), unidirectional extended NetFlow data sequence (U_ENData), bidirectional original NetFlow data sequence (B_ONData) and bidirectional extended NetFlow data sequence(B_ENData). Bidirectional NetFlow data sequences contains more information than unidirectional data. In addition, to verify the sensitivity and validity of our model, we take three sampling rates: 1, 1/10 and 1/100. Table 1 shows these NetFlow data sequences in detail. The *bps, pps, bpp* indicate the "bytes per second", "packets per second" and "bytes per packet" respectively.

3.2 The NetFlow Sequence Attention Network

In this section, we present our NetFlow Sequence Attention Network (NSA-Net) in detail. The NSA-net considers both feature learning and classification together. As shown in Fig. 2, NSA-Net in this paper contains four layers. In the following of this section, we will describe each layer in detail.

Table 1. The NetFlow dataset description.

Data type	Information description
U_ONData	{duration, protocol, sport, dport, TCP-flag, pktnum, bytnum}
U_ENData	U_ONData + {bps, pps, bpp}
B_ONData	{duration, protocol, sport, dport, TCP-flag, uplink-pktnum, uplink-bytnum, downlink-pktnum, downlink-bytnum, flow-num}
B_ENData	B_ONData + {uplink{bps, pps, bpp}, downlink{bps, pps, bpp} }

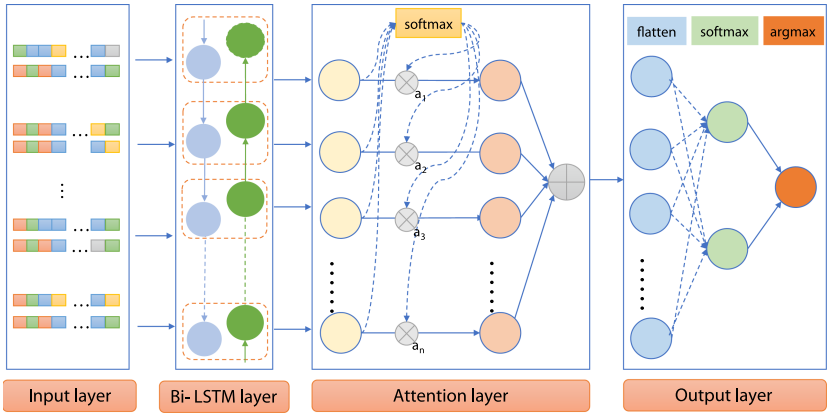

Fig. 2. The framework of our work.

Input Layer. In order to make the data fit the model better, we need to standardize our data to avoid outliers and extremes. In addition, classification labels encoded with one-hot coding. As we evaluate different kinds of NetFlow sequences on our proposed model, each input needs to be reshaped into a form of tensor to meet the input format of bidirectional LSTM. Then, the converted NetFlow sequences will be fed to the bi-LSTM layer.

Bi-LSTM Layer. This layer takes the converted vectors of a NetFlow sequence as input and generates high level features. To further improve the representation of the model, we use the bidirectional LSTM(bi-LSTM) in our model, which is an artificial recurrent neural network (RNN) architecture.

Bi-LSTM is an extension of traditional LSTM, which can connect two hidden layers of opposite directions to the same output. Therefore, it is able to exploit sequential information from both directions and provide bidirectional context to the network and result in fuller learning on the problem.

Given an input NetFlow sequence $S = [x_1, x_2, \cdots, x_n]$, the bi-LSTM contains a forward LSTM network which reads S from x_1 to x_n and a backward LSTM network which reads S from x_n to x_1. At time step t($t \in [1, n]$), the $\overrightarrow{h}_t$ [Eq.(1)] and $\overleftarrow{h}_t$[Eq.(2)] are the forward and backward hidden states respectively. Then,

the states h_t [Eq.(3)] is obtained from concatenating the forward hidden state and the backward hidden state at the time step t.

$$\overrightarrow{h}_t = \overrightarrow{LSTM}(\overrightarrow{h}_{t-1}, x_t) \tag{1}$$

$$\overleftarrow{h}_t = \overleftarrow{LSTM}(\overleftarrow{h}_{t-1}, x_t) \tag{2}$$

$$h_t = [\overrightarrow{h}_t, \overleftarrow{h}_t] \tag{3}$$

Attention Layer. The essence of the attention mechanism is the idea that it selectively filters out a small amount of important information from a large amount of information and focuses on the important information, ignoring mostly unimportant information. Attention can assign weights to information and finally weight the information. Since the appearence of the attention mechanism, it has now been applied in many areas. Here, we adopt the attention mechanism for filtering out the important information from the output of bi-LSTM layer.

Let H represent the output vector of bi-LSTM layer $[h_1, h_2, h_3, ..., h_n]$, we can get the vector e by a linear transformation of H. Then, softmax function is adopted to compute the weightings α of vector e(Eq.(4)). Finally, the attention vector c is calculated by Eq.(5).

$$\alpha = softmax(e), (\alpha_t = \frac{exp(e_t)}{\sum_{k=1}^{n} exp(e_k)}, t \in [1, n]) \tag{4}$$

$$c = \sum_{t=1}^{n} \alpha_t h_t \tag{5}$$

Output Layer. In the output layer, after flattening the output vector of the attentional layer, we use another softmax classifier to obtain the distribution of different categories (similar to Eq.(4)). And we take the category with the maximum probability as the prediction label y.

4 Experiments

In this part, we give a comprehensive introduction to our experiments. We mainly present our dataset, evaluation criteria, model setting, experiments and sensitivity analysis.

4.1 Dataset

In this paper, the public VPN-nonVPN traffic dataset (ISCXVPN2016) [4], which was published by the University of New Brunswick, is selected to evaluate our proposed model. There are two data formats in this traffic dataset, flow features and raw traffic (i.e. pcap format). We use the raw traffic dataset as our test

dataset, which contains about 150 pcap files. The raw traffic is about 25 GB in the pcap format, which includes regular encrypted traffic (such as Skype and Facebook) and the corresponding traffic with VPN (such as VPN-Skype and VPN-Facebook). Data preprocessing, which is described in detail in Sect. 3.1, is conducted based on ISCXVPN2016 raw traffic. U_ONData, U_ENData, B_ONData and B_ENData represent the unsampled NetFlow data sequence. B_ENData_10S and B_ENData_100S represent the bidirectional extended NetFlow data sequence at 1/10 and 1/100 sampling rates. Finally, we use the datasets (Table 2) to test and verify our NSA-Net model.

Table 2. The statistical information of NetFlow datatset.

Data type	VPN flows	Non-VPN flows
U_ONData and U_ENData	5530	49625
B_ONData and B_ENData	2795	42695
B_ENData_10S	1453	8370
B_ENData_100S	510	863

4.2 Evaluation Critera

In this paper, we use True Positive Rate (TPR), False Positive Rate (FPR), and Accuracy(Acc) as our assessment indicators. Acc means the ratio of all correctly predicted samples and total samples, which is used to evaluate the overall performance of the model. Due to the existence of unbalanced data, the accuracy can not completely evaluate whether a model is good or not. Therefore, TPR and FPR indicators are selected to evaluate the model. TPR means the ratio of predicting positive sequences as positive and all positive sequences. And FPR means the ratio of predicting negative sequences as positive and all negative sequences.

4.3 NSA-Net Setting

To implement our proposed model, we choose Keras Library [24], with Tensorflow [23] as its backend, which runs on Redhat 7.2 64bit OS. We take the NetFlow sequences as the input of the NSA-Net. 1/5 of data were randomly selected as test data, and the rest is training data. The mini-batch size is 60 and the cost function is cross entropy. Moreover, we set the dimension of hidden states of each LSTM as 128, and take dropout with 0.4 ratio to avoid over-fitting, and the Adam optimizer [25] with learning rate 0.005 is used. Totally, the learning procedure adopts early-stopping.

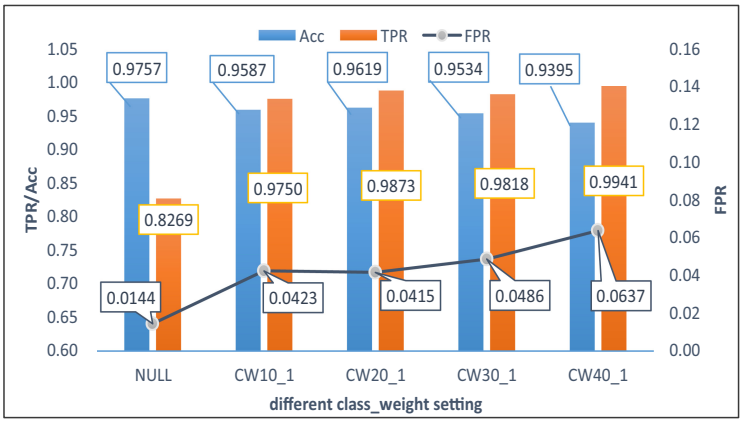

Fig. 3. Results of NSA-Net with different class_weight.

Class_weight Setting. As shown in Table 2, our non-VPN sequences number is much larger than that of the VPN sequence, which indicates samples of our NetFlow dataset are out-of-balance. On the problem of binary classification of unbalanced data, the accuracy indicator cannot fully reflect the performance of a model. In the paper, our goal is to get a higher TPR and a lowever FPR, while guaranteeing a certain accuracy. Keras provides a **class_weight** parameter for model fitting. In the process of model training, class_weight parameter can make the loss function pay more attention to the data with insufficient samples, by providing a weight or bias for each output class.

In order to solve the inbalance problem, different values of class_weight (i.e., 10:1, 20:1, 30:1, 40:1) are set, which means VPN samples are provided a bigger weight. As shown in Fig. 3, the higher the weight of the positive sample(VPN sample), the better the TPR obtained by the model. The parameter class_weight can effectively promote the TPR of our model. When we do not set the class_weight parameter, the TPR is only 82%. However, the TPR achieves more than 98% when class_weight is in {20:1, 30:1, 40:1}, and increases about 16%. Although it leads to increased FPR and reduced accuracy, the impact is small. Finally, considering the overall performance, we choose {20:1} as class_weight parameter by considering the performance of all metrics.

Optimal Inputs Selection. We have four kinds of NetFlow data, which contain U_ONData, U_ENData, B_ONData, and B_ENData. The comparison results are shown in Fig. 4. From the comparative results, we can obtain the following conclusions:

1. Bidirectional NetFlow sequence outperform the unidirectional sequence. Although TPR increases by only 1%, Acc increases by almost 15%, and FPR decreases by about 16%. The main reason is that the bidirectional NetFlow

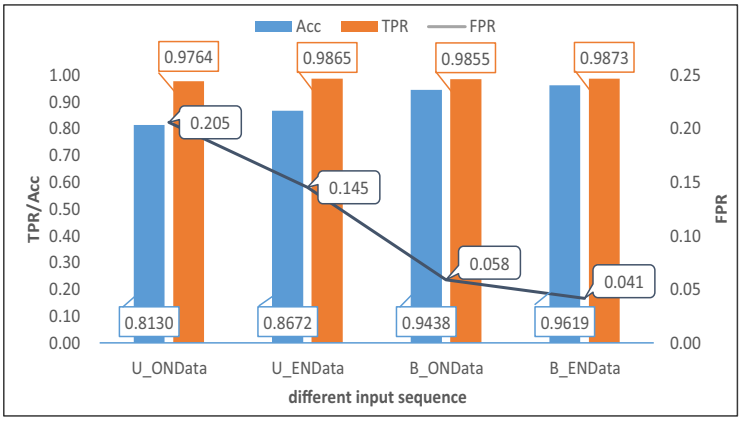

Fig. 4. Results of NSA-Net with different input sequences.

sequence aggregates two-way flow information, therefore it has better feature representation than the unidirectional sequence.

2. Extended sequence, which adds *bps, pps, bpp* information, have enhanced the performance of the model. For the bidirectional data, the Acc of extended sequence improves almost 2%, and the FPR decreases by 1%. As a result, the richer the information, the more expressive the model.

Therefore, we choose the bidirectional extended sequence (B_ENData) as our default input data.

4.4 Comparison Experiments

Different Deep Learning Models Comparison. Here, a number of typical deep learning models, including 1-layer bidirectional LSTM(1biLSTM), 2-layer bidirectional LSTM with Attention mechanism(2biLSTM+Att), 1D-CNN in [6] are applied to our NetFlow sequence to detect VPN.

Figure 5 respectively describes the experimental results of TPR, FPR, and Acc of these classifiers, which indicates that our proposed NSA-Net model outperforms three alternatives methods in VPN detection. The Accuracy and TPR of NSA-Net are higher than 1biLSTM by 1%, which indicates that the attention mechanism is able to enhance the performance of the model. Compared to our model, 2biLSTM+Att has no significant improvement, but it has one more bi-LSTM layer, which needs more time to train the model. In addition, 1D-CNN models in [6], which presents the best performance in VPN detection, is applied to our dataset. The results reveal that the ability of the 1D-CNN model to learn features from NetFlow data is not as good as our NSA-Net model. Therefore, the best overall performance is our proposed NSA-Net model.

Existing Methods Comparison. The state-of-the-art method [4] is compared with our proposed NSA-Net model comprehensively. As we focus on detecting

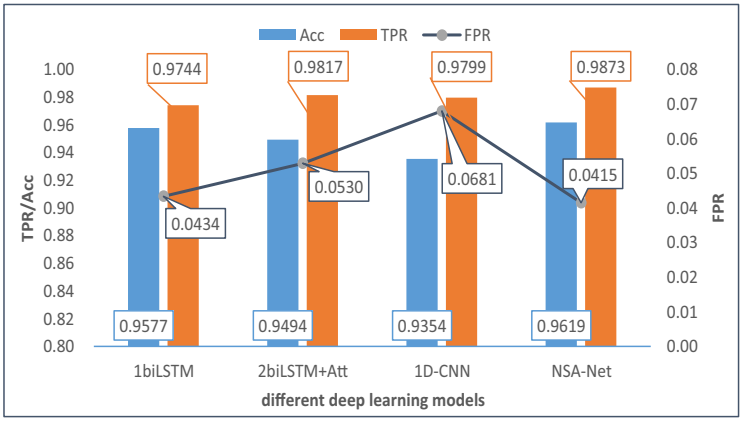

Fig. 5. Comparison results of different deep learning models.

VPN from non-VPN traffic, we compare the two-category identification results of the method. We make a comparison from the following aspects: (1) C45-TRF: the existing work in [4], (2) C45-BEND: the existing model used in [4] applied to our NetFlow data(B_ENData), (3) NSA-TRF: our NAS-Net model applied to the time-related features in [4], and (4) NSA-BEND: our work. Table 3 shows the comparison results between our proposed method and the state-of-the-art method in [4]. As their paper only showed the precision and recall(TPR) results, so we compare performance using the recall(Rec), precision(Pre), and F1-score(F1), which presents an overall evaluation of precision and recall.

As shown in Table 3, NetFlow sequence shows better than time-related features at these two methods, which indicates that flow sequence features have great potential in traffic detection. Obviously, the NSA-BEND performs the best, which indicates that our NSA-Net model performs very well in VPN detection based on our NetFlow data. Although our method needs a long time than existing C4.5 method, our overall performance is better than the other methods.

Table 3. Comparison results with existing method.

Method	Model	Data	Rec	Pre	F1
C45-TRF	C4.5 [4]	Time-related features [4]	0.9200	0.89	0.9048
C45-BEND	C4.5 [4]	B_ENData	0.8539	0.9818	0.9133
NSA-TRF	NSA-Net	Time-related features [4]	0.7388	0.7569	0.7477
NSA-BEND	NSA-Net	B_ENData	**0.9873**	**0.9735**	**0.9804**

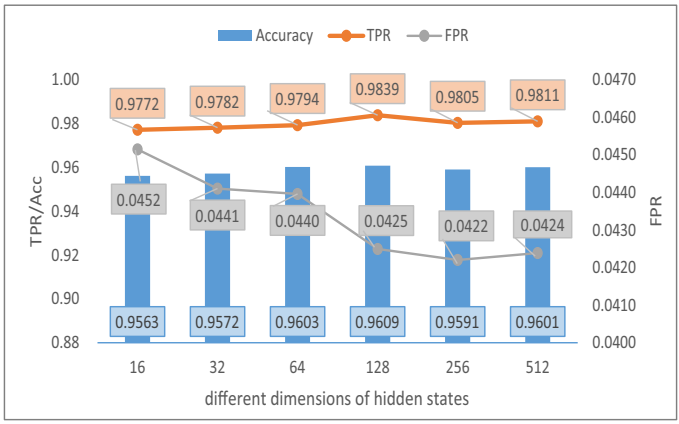

Fig. 6. The results of different dimensions of hidden states.

4.5 Sensitivity Analysis

The Dimension of Hidden States: Hidden states of bi-LSTM layer in the NSA_Net model is used to extract the latent usful information of the NetFlow data sequences. The larger units number, the stronger the ability to learn potential information, and the longer training time. The dimension of hidden states is vital to the performance of our model.

Therefore, we select different dimensions of bi-LSTM hidden states (i.e., 16, 32, 64, 128, 256, 512) to train our model. From the results shown in the Fig. 6, the performance of the model is improving as the dimensions of hidden states increases. Even the dimension of hidden state is small, our model still performs well. When the dimensions of hidden states exceeds 128, the training time becomes longer and the enhancement of detection effects becomes less and less obvious. Therefore, considering the balance between performance and training time, we choose 128 as the dimension of our paper.

The Sampling Rates: In this section, we analyze the influence of different sampling rates on the detection results. Due to the explosive growth of network traffic, packet sampling is widely used as a means of network measurement and network management to reduce resource consumption. The lower the sampling rate, the less useful information could be extracted and used.

Therefore, we focus on analyzing the result based on the bidirectional extended NetFlow sequence at different sampling rates($p = 1,\ 1/10,\ 1/100$), which is presented in Fig. 7. Obviously, as the sampling rate decreases, the TPR and Acc decrease while the FPR rises. The main reason is that the sampled data is only a part of raw traffic data, and some information of traffic characteristics will be lost. The datasets(Table 2) reveals that as the sampling rate decreases,

the amount of data decreases dramatically. The decrease in available information will inevitably lead to a decrease in detection rate.

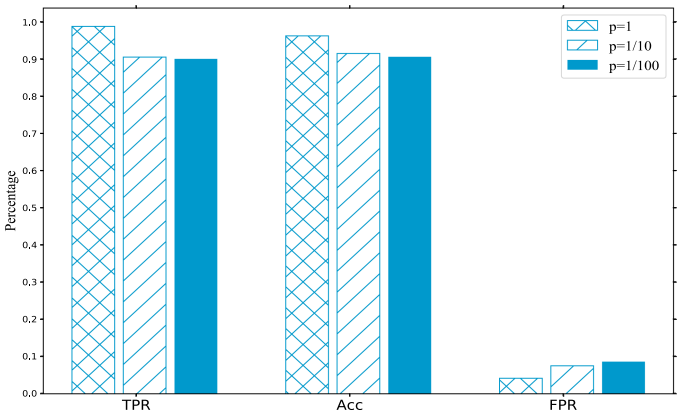

Fig. 7. The overall performance at different sampling rates.

Despite the decrease in available information, our model still achieves over 90% accuracy and TPR at low sampling rates. It reveals that our NSA-Net model still has great potential in sparse data environment.

From the above comprehensive comparison experiments, we can get that our proposed NSA-Net outperforms other deep learning models and the state-of-the-art method. While considering the user privacy and lightweight input, our model still achieves excellent performance. Therefore, the NSA-Net model is a very promising VPN detection method based on the NetFlow sequence, even at low sampling rate situation. In our future work, we will apply our NSA-Net model on a real-world Netlfow dataset. I believe that our work will still work very well.

5 Conclusion

In this paper, considering the feature automatic learning and user privacy protection, we propose a model named NAS-Net to detect VPN traffic. The NSA-Net takes the bidirectional LSTM model with attention mechanism to learn features from the NetFlow sequences rather than the raw traffic. As far as we know, this is the first time to combine attention mechanism and NetFlow data for VPN traffic detection. The model combines the advantages of a recurrent neural network and attention mechanism to learn representative information from NetFlow data automatically and saves human effort to design features. In addition, we demonstrate that the bidirectional NetFlow data could improve the performance of detection. Thorough and comparative experiments on the NetFlow data generated from the public ISCXVPN2016 traffic dataset are conducted to

validate the effectiveness of the NSA-Net model. The experiment results show that our NSA-Net can achieve an excellent performance(0.987 TPR, 0.041 FPR, and 0.962 Acc). Even at low sampling rate, our NSA-Net model still has great potential.

Acknowledgments. This work is supported by The National Key Research and Development Program of China (No. 2020YFE0200500 and No.2016QY05X1000) and The Key research and Development Program for Guangdong Province under grant No. 2019B010137003 and The National Key Research and Development Program of China (No. 2018YFB1800200). Zhen Li is the corresponding author.

References

1. Harmening, J.T.: Virtual private networks. In: Vacca, J.R. (ed.) Computer and Information Security Handbook, pp. 843–856. Morgan Kaufmann, Burlington (2017)
2. Lotfollahi, M., Siavoshani, M.J., et al.: Deep packet: a novel approach for encrypted traffic classification using deep learning. Soft Comput. **24**(3), 1999–2012 (2020). https://doi.org/10.1007/s00500-019-04030-2
3. Zain ul Abideen, M., Saleem, S., Ejaz, M.: VPN traffic detection in SSL-protected channel. Secur. Commun. Netw. **2019**(5), 1–17 (2019)
4. Draper-Gil, G., Lashkari, A.H., Mamun, M.S.I., et al.: Characterization of encrypted and vpn traffic using time-related. In: ICISSP, pp. 407–414 (2016)
5. Bagui, S., Fang, X., et al.: Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features. J. Cyber Secur. Technol. **1**(2), 108–126 (2017)
6. Wang,W., Zhu, M., Wang, J., et al.: End-to-end encrypted traffic classification with one-dimensional convolution neural networks. In: 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), pp. 43–48. IEEE (2017)
7. Miller, S., Curran, K., Lunney, T.: Multilayer perceptron neural network for detection of encrypted VPN network traffic. In: 2018 International Conference On Cyber Situational Awareness, Data Analytics And Assessment, pp. 1–8. IEEE (2018)
8. Guo, L., Wu, Q., Liu, S., et al.: Deep learning-based real-time VPN encrypted traffic identification methods. J. Real-Time Image Proc. **17**(1), 103–114 (2020). https://doi.org/10.1007/s11554-019-00930-6
9. Claise, B.: Cisco systems neflow services export version 9 (2004)
10. Zhou, P., Shi, W., Tian, J., et al.: Attention-based bidirectional long short-term memory networks for relation classification. In: Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pp. 207–212 (2016)
11. Hofstede, R., Hendriks, L., Sperotto, A., et al.: SSH compromise detection using NetFlow/IPFIX. ACM SIGCOMM Comput. Commun. Rev. **44**(5), 20–26 (2014)
12. Schatzmann, D., Mühlbauer, W., Spyropoulos, T., et al.: Digging into HTTPS: flow-based classification of webmail traffic. In: Proceedings of the 10th ACM SIGCOMM Conference on Internet Measurement, pp. 322-327 (2010)
13. Manzoor, J., Drago, I., Sadre, R.: How HTTP/2 is changing Web traffic and how to detect it. In: 2017 Network Traffic Measurement and Analysis Conference (TMA), pp. 1–9. IEEE (2017)
14. Lv, B., Yu, X., Xu, G., et al.: Network traffic monitoring system based on big data technology. In: Proceedings of the International Conference on Big Data and Computing 2018, pp. 27–32 (2018)

15. Liu, X., Tang, Z., Yang, B.: Predicting network attacks with CNN by constructing images from NetFlow Data. In: BigDataSecurity, pp. 61–66. IEEE (2019)
16. Yang, C.T., Liu, J.C., Kristiani, E., et al.: NetFlow monitoring and cyberattack detection using deep learning with Ceph. IEEE Access **8**, 7842–7850 (2020)
17. Mnih, V., Heess, N., Graves A.: Recurrent models of visual attention. In: Advances in Neural Information Processing Systems, pp. 2204–2212 (2014)
18. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. Comput. Sci. arXiv preprint arXiv:1409.0473 (2014)
19. Chorowski, J., Bahdanau, D., Serdyuk, D., et al.: Attention-based models for speech recognition. Comput. Sci. **10**(4), 429–439 (2015)
20. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. Comput. Sci. arXiv preprint arXiv:1508.04025 (2015)
21. Softflowd. http://www.mindrot.org/projects/softflowd/
22. Nfdump. http://nfdump.sourceforge.net/
23. Abadi, M., Agarwal, A., et al.: Tensor-flow: large-scale machine learning on heterogeneous distributed systems, arXiv preprint arXiv:1603.04467 (2016)
24. Chollet, F., et al.: Keras (2017). https://github.com/fchollet/keras
25. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR, vol. abs/1412.6980 (2014)