Validating the effectiveness of machine-learning to detect use of VPN

1st Yifan Zhang
Department of Mathematics
University of Waterloo
Waterloo, Canada
y3748zha@uwaterloo.ca

I. Introduction

A. Background

VPN, now widely used all over the world, is one of the major practices to ensure privacy. It masks user IP address by re-routing all traffic from the user to intermediate proxy server(s) (one intermediate server in the case of one-hop VPN, two or more intermediate servers in the case of multi-hop VPN) such that the IP address of the user will be masked by IP address of proxy servers. User contents will also be encrypted during this process such that hackers, government agencies, businesses, or anyone else that does not have the decryption key, can not get access to user contents (different types of encryptions are used by different VPN providers, and there are three main types: hashing, symmetric cryptography, and asymmetric cryptography). Nowadays, there are 2 basic types of VPNs: remote-access VPN which is also known as client-to-server VPN and site-to-site VPN which is also known as router-to-router VPN. A remote-access VPN allows users to connect to a private network through a private encryption tunnel, while the site-to-site VPN joins two networks on different locations so that hosts within these two networks can connect to each other [1]. People choose VPN for different reasons, which can be ascribed into two main categories: protecting privacy & unblocking access. The first category of reasons is quite obvious. After the elimination of net-neutrality, ISPs are legitimated to collect client data, including browsing history, physical location, etc, and can use these to discriminate against certain users by slowing down their bandwidth. ISPs may also sell these client data to third party businesses, which may use these data to build targeted business strategies for corresponding users. Apart from the above scenarios, a client may also want to avoid the censorship of government authorities when posting on a public forum, or avoid being hackered when using public network at a coffee shop. In summary, today's internet is full of risks, and VPN privides its users with a sense of safety. The second use case of VPN for unblocking access to certain contents, is also becoming more and more popular. For example, when I was back in China, I do not have access towards a series of useful websites such as Google or Youtube unless I use VPN to pretend I was in another country. There are also cases when the requested contents are region-restraint (only available to people in certain regions) due to copyright issues, and such restrictions can also be circumvented through use of VPN [2].

B. Motivations

There is this old saying that "one man's meat is another man's poison", and this is exactly the case with VPN. While clients have various reasons to use VPN, websites (including some globally famous ones) also have plenty of reasons to forbid them. For example, streaming sites like Netflix, Amazon and BBC choose to blacklist VPNs addresses to honor regional contracts with licensing companies; merchandising websites like Paypal aslo blacklists VPN addresses since criminals using them for spam and fraud are more difficult to track down; and although the reason is unclear, some public network also starts to blacklisting VPN addresses. Besides websites, government can be one of the top objectors against the use of VPNs. Back in May 2022, Chinese government announced a large-scale operation called "network purification" that aimed at the use of VPNs to propagate online fraud, gambling, and distribution of other illegal contents [4].

Merely blacklisting VPN addresses is surely not enough, as VPN providers can always switch to new, un-blacklisted addresses, not to mention techniques to dynamically allocate address each time. This approach is also not scalable considering the amount of storage required. Machine learning based techniques, on the other hand, can detect the use of VPN through features of a network traffic regardless of its IP address, and require much less storage (only the pre-trained models need to be cached). There have already been numerous previously proposed machine learning techniques to detect the use of VPN (which will be discussed in more details in *related works* section), however, there is still room for improvement. The first issue with previous studies is that they all used the same dataset: ISCXVPN2016 [5] published by Lashkari et al. [6]. This fact compromises the external generalizability of their results, but what makes it worse is the fact that the ISCXVPN2016 dataset has some inherent limitations. The ISCXVPN2016 dataset contains both VPN and non-VPN network traffics from 6 different application categories: Email(14.94%), Chat(18.92%), Streaming File(14.85%), Transfer(16.70%), VoIP(16.70%) and

P2P(17.89%). In an analysis of this dataset in 2019, Peter et al. [7] showed the possibility to classify traffic in ISCXVPN2016 dataset by only mapping packet header and without incoporating any payload data. In another more recent study in May 2022, Jorgense et al. [9] found that some packets within the ISCXVPN2016 dataset has unencrypted payload, which can unfairly strengthen traffic classification methods that leverage packet payloads, such as DPI or the deep packet approaches which do not fit into today's network due to the prevalent usage of encrypted protocols such as HTTPS. In addition, they also found that some VPN-label traffic captures contain multiple connections, while the anticipated VPN captures should only contain a single connection between the VPN client and the VPN server. the presence of multiple connections per VPN capture will unfairly strengthen methods that leverage connection information aggregated at the flow-level. Motivated by the above limitations, Jorgensen et al. published their improved VPN/non-VPN network application traffic dataset: VNAT [10], based on which this study is made. One last thing about previous studies is that they mostly focus on classifying between different application categories and design their approaches accordingly, while this study will focus more on classifying VPN/non-VPN traffic.

C. Main results

Both machine learning and deep learning models generate satisfying performances on classifying VPN/non-VPN traffic in VNAT dataset.

D. Main contributions

This paper makes the following main contributions

- We applied two machine learning techniques and one deep learning technique (all three not used in previous studies) to classify network traffic, using a more recent dataset that improves limitations of a previous, widely used dataset. Therefore, this study increases both internal and external validity of the potential of machine learning techniques to classify network traffics.
- We focused more on classifying VPN/non-VPN traffic, while previous studies focused more on classifying the application category that generated the traffic.

E. Paper structure

This paper is written in the following structure: first, section 1 is a summary of the whole paper, while section 2 provides necessary background knowledge, describes the motivation behind this study, and summarizes the main results, contribution as well as novelty of the study. Then, section 3 describes the dataset and models used for this study as well as the end-to-end process, while section 4 demonstrates the full results of this study. Finally, section 5 covers related previous works and section 6 finishes this paper with a discussion on threats to validity or limitations at the moment and corresponding potential future works.

F. Replication

The three models generated from this study is made public at https://github.com/zhan3680/CS656-project for replication and validation purposes.

II. Approach

A. Data

The VNAT dataset consists of approximately 272 hours of packet capture from five traffic categories. The categories, and their corresponding applications are listed in table I.

TABLE I
TRAFFIC CATEGORY AND CORRESPONDING APPLICATIONS

Traffic category	Application
Streaming	Vimeo, Netflix, YouTube
VoIP	Zoiper
Chat	Skype
Command & Control	SSH, RDP
File Transfer	SFTP, RSYNC, SCP

To create the dataset, virtual sub-networks are created for each category. Each subnetwork consists of a client, a client DNS server, a VPN client, and a VPN server (In addition to the basic settings, The Skype subnetwork contains an additional client to allow for bidirectional chat. The video streaming and web browsing sub-networks are connected to the Internet to enable access to Firefox, Chrome, YouTube, Netflix, and Vimeo. See figure 1). Once the sub-networks are set up, traffic are generated for each category from the corresponding applications separately: the Streaming traffic are generated by watching videos on each of the three platforms, the VoIP traffic are generated by having conversations over Zoiper, the Chat traffic are generated with bots that replay actual chats from publicly available logs from Gitter.im chat rooms, the Command & Control traffic are generated by performing predefined tasks over RDP and by executing predefined SSH commands respectively, and finally, the File Transfer traffic are generated by sending files of sizes ranging from 1KB to 1GB to remote server using one of SFTP, RSYNC, or SCP. All traffic are captured using tcpdump and stored in the PCAP format. The non-VPN traffic are captured between the client and the VPN client while the VPN traffic are captured between the VPN client and the VPN server, as shown in figure 1 [9].

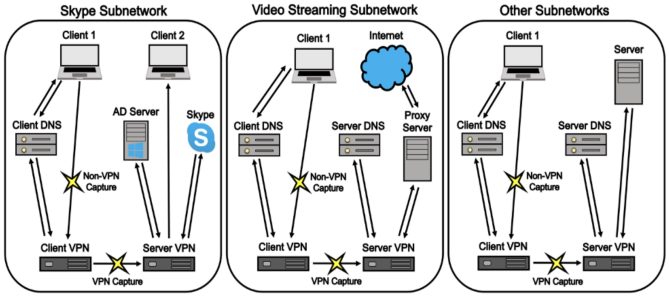

Fig. 1. Sub-network set up and traffic capture for different categories

B. Pre-processing

Due to the limitation of computation power of my machine, I chose only traffic from the "Chat" category for this study. The chat category traffic consists of 11336 packets in total, among which 5934 are VPN-encrypted. After packets are read from the pcap format into csv format, they are first labeled with either "VPN"' or "non-VPN", then shuffled, and finally split into training and testing data at a 7:3 ratio.

C. Models

Two machine learning models and one deep learning models are used in this study.

Machine learning models:

- Decision Tree [11]: this is a classical machine learning model that classifies input data into sub-classes by filtering them through a series of internal conditions. In this study, I used the *DecisionTreeClassifier* from Python sklearn package;
- Adaptive Boosting (Adaboost) [12]: this is a typical ensemble machine learning model that ensembles several weak classifiers called "stumps", which only has one node and two leaves. Stumps are added sequentially based on their weighted accuracy, and the later added stumps will try to reduce bias of previously added ones by giving higher weights to the input samples they incorrectly classified. At the ensemble stage, stumps with higher accuracy are given higher weights. The iterative process ends when either all input samples are correctly classified, or the maximum level of iteration has been reached. In this study, I used the *AdaBoostClassifier* from Python sklearn package;
- Long Short-Term Memory Network (LSTM) [13]: this is a recurrent neural network architecture with feedback connections, which can be used to classify entire sequence of data. In this study, I built a LSTM model using the Python Keras module, which consists of two LSTM layers(the first is the LSTM input layer), one Dense output layer, and two Dropout layers in between to mitigate over-fitting.

III. Results

To compare the results, I used the *f1-score* metric from python sklearn, which is harmonic mean of the precision and recall defined as:

$$\text{f1-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \tag{1}$$

where *precision* is the proportion of positive identifications that are actually positive, given by $precision = \frac{TP}{TP+FP}$, and *recall* is the proportion of actual positives that are correctly identified as positive, given by $recall = \frac{TP}{TP+FN}$. (TP stands for "true positive", FP stands for "false positive" and FN stands for "false negative")

The f1-scores for Decison Tree are in table II, the f1-scores for Adaboost are in table III, and the f1-scores for LSTM are in IV.

TABLE II
F1-SCORE FOR DECISION TREE

Category	f1-score
nonvpn	1.00
vpn	1.00
weighted average	1.00

TABLE III
F1-SCORE FOR ADABOOST

Category	f1-score
nonvpn	1.00
vpn	1.00
weighted average	1.00

IV. Related Works

Lashkari et al. [6] generated and released the famous VPN/non-VPN dataset ISCXVPN2016. In their study, they first grouped packets into flows using Source IP, Destination IP, Source Port, Destination Port and Transport layer Protocol as identifier. They then extracted flow-based features such as duration of the flow, average forward & backward internal arrival time between packets in the flow, flow active duration, as well as number of bytes or packets per second in the flow. For machine learning models, they chose KNN and C4.5, and their results showed that time-related features is a good indicator for network traffic category, whether the traffic is VPN-encrypted or un-encrypted. However, since they focus more on the network traffic category, and have 14 sub-classes in total (7 for VPN and 7 for non-VPN), their results does not clearly show whether time-related features can be used to detect use of VPN. Since publish, the ISCXVPN2016 dataset has motivated numerous machine learning related studies. Shane et al. [18] used multi-layer perceptron neural network on the same set of time-related features as Lashkari et al., and provided an automated end-to-end feature extraction, training and testing process through Azure machine learning studio. In 2017, Bagui et al. performed a comparison study of multiple machine-learning algorithms for classification of VPN network traffic flow, including GBT, KNN, LR, NB, RF and SVM. They used the same set of time-related features as Lashkari et al., and their results showed that under optimization, Random Forest and Gradient Boosting Tree outperform other models in terms of accuracy and low over-fitting. More recent studies attempted to first convert network traffic into images, and then use CNN models to classify them. Examples of such directions include [19]–[21]. While such techniques achieve better

TABLE IV
F1-SCORE FOR LSTM

Category	f1-score
nonvpn	0.975
vpn	0.978
weighted average	0.977

performances, they are commonly subject to the limitation of large-sized image representation of network traffic, resulting in very large storage and computational resource overhead. To solve this issue, He et al. [16] came up with a novel lightweight anonymous proxy traffic detection method that first converts the sequences of the size and inter-arrival time of the first N packets of a flow into images, and then categorizes the converted images using the one-dimensional convolutional neural network. This method achieves similar performances, while reducing the overhead by more than 90 percent.

As discussed in the Introduction session, though the ISCXVPN2016 dataset was widely used in previous machine learning based studies, it has several inherent limitations. Therefore, in 2022 Jorgensen et al. [9] generated their improved version of VPN/non-VPN dataset, and built their own neural network to train and test on this dataset. The features they extracted are very similar to those used by Lashkari et al. [6]. They achieved an f1-score of 0.98, and demonstrate that their framework can extend to an enterprise network.

There are also some recent studies that tried to detect the use of VPN or anonymizing proxy servers, but without the use of machine learning. For example, Chen et al. [23] presented an approach that uses bipartite graphs to model host communications from network traffic and build one-mode projections of bipartite graphs for discovering social-behavior similarity of web proxy users. Based on the similarity matrices of end-users from the derived one-mode projection graphs, they applied a simple yet effective spectral clustering algorithm to discover the inherent web proxy users behavior clusters, and used these clusters to classify whether the top URLs visited by the web proxy users are web proxies.

V. Discussion

The resulting f1-score, especially for Decision Tree and Adaboost models, are very high, which inevitably leads to the discussion on over-fitting. I followed the online tutorial form *Towards Data Science* [14] to apply both pre-pruning and post-pruning techniques to the Decision Tree model to tune hyper parameters including max_depth, min_samples_leaf, min_samples_split and ccp_alpha, but the resulting f1-scores after this effort is still 1. I did not apply any further techniques to the Adaboost model, as it is an ensemble method, and ensembling itself is a technique to mitigate over-fitting. For the LSTM model, two Dropout layers with dropout rate 0.5 are added. One of my guess is that in the resulting model, the packet headers unluckily far outweigh the packet payloads, resulting in over-fitting. Due to time limitation, I haven't validated this hypothesis, but this is certainly one of the future works with top priority.

Another potential perspective for future improvements is the tuning of hyper-parameters in the models. Due to time limitation, I only tuned a small fraction of the hyper-parameters, such as max_depth of Decision Tree model. For most of the hyper-parameters, I simply used either the default setting, or other people's parameters that are publicly available online, such as "gini" for Decision Tree criterion, "adam" for LSTM activation function, "binary_crossentropy" for LSTM loss function, and 7:3 for training-testing split ratio. In the future, with a more thorough tuning of all these parameters, I should be able to further improve the results.

This study only use traffic from "Chat" category, which may leads to doubt on the generalizability of the results, in the future I should extend the approach to other categories in the VNAT dataset as well, in order to further increase the external validity of the study.

This study focus on features that are related with each individual packet, while most of the previous works extract features from a flow of packets. An obvious advantage of using flow-based features rather than using packet-based features is to avoid over-fitting. It is a good direction of future study to apply the techniques of flow-based feature extraction and classification in the previous works to the VNAT dataset, and compare their results.

Finally, both the VNAT dataset used in this study and the ISCXVPN2016 dataset used in most of the previous studies contain only traffic from one-hop VPN, so it would be interesting to collect traffic using multi-hop VPN and see the performance of the same approaches on that dataset.

REFERENCES

[1] https://www.top10vpn.com/what-is-a-vpn/vpn-types/
[2] https://www.avast.com/c-what-is-a-vpn
[3] https://www.howtogeek.com/403771/why-do-some-websites-block-vpns/
[4] https://baijiahao.baidu.com/s?id=1733223972351393246
[5] https://www.unb.ca/cic/datasets/vpn.html
[6] Draper-Gil, G., Habibi Lashkari, A., Mamun, M.S., & Ghorbani, A.A. (2016). Characterization of Encrypted and VPN Traffic using Time-related Features. ICISSP.
[7] Mo, Chen & Xiaojuan, Wang & Mingshu, He & Lei, Jin & Javeed, Khalid & Wang, Xiaojun. (2020). A Network Traffic Classification Model Based on Metric Learning. Computers, Materials & Continua. 64. 941-959. 10.32604/cmc.2020.09802.
[8] https://github.com/Mr-Pepe/iscx-analysis
[9] Steven Jorgensen, John Holodnak, Jensen Dempsey, Karla de Souza, Ananditha Raghunath, Vernon Rivet, Noah DeMoes, Andrés Alejos, Allan Wollaber (2022) Extensible Machine Learning for Encrypted Network Traffic Application Labeling via Uncertainty Quantification. arXiv:2205.05628
[10] https://www.ll.mit.edu/r-d/datasets/vpnnonvpn-network-application-traffic-dataset-vnat
[11] Quinlan, J.R. Induction of decision trees. Mach Learn 1, 81–106 (1986). https://doi.org/10.1007/BF00116251
[12] Freund, Y., Schapire, R.E. (1995). A desicion-theoretic generalization of on-line learning and an application to boosting. In: Vitányi, P. (eds) Computational Learning Theory. EuroCOLT 1995. Lecture Notes in Computer Science, vol 904. Springer, Berlin, Heidelberg. https://doi.org/10.1007/3-540-59119-2_166
[13] Hochreiter, Sepp & Schmidhuber, Jürgen. (1997). Long Short-term Memory. Neural computation. 9. 1735-80. 10.1162/neco.1997.9.8.1735.
[14] https://towardsdatascience.com/3-techniques-to-avoid-overfitting-of-decision-trees-1e7d3d985a09
[15] Miller, S., Curran, K., & Lunney, T. (2021). Detection of Anonymising Proxies Using Machine Learning. International Journal of Digital Crime and Forensics (IJDCF), 13(6), 1-17. http://doi.org/10.4018/IJDCF.286756
[16] He, Y.; Li, W. A Novel Lightweight Anonymous Proxy Traffic Detection Method Based on Spatio-Temporal Features. Sensors 2022, 22, 4216. https://doi.org/ 10.3390/s22114216
[17] Julian Andres Caicedo-Muñoz, Agapito Ledezma Espino, Juan Carlos Corrales, Alvaro Rendón, QoS-Classifier for VPN and Non-VPN traffic based on time-related features, Computer Networks, Volume 144, 2018, Pages 271-279, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2018.08.008.
[18] S. Miller, K. Curran and T. Lunney, "Multilayer Perceptron Neural Network for Detection of Encrypted VPN Network Traffic," 2018 International Conference On Cyber Situational Awareness, Data Analytics And Assessment (Cyber SA), 2018, pp. 1-8, doi: 10.1109/CyberSA.2018.8551395.
[19] T. Shapira and Y. Shavitt, "FlowPic: A Generic Representation for Encrypted Traffic Classification and Applications Identification," in IEEE Transactions on Network and Service Management, vol. 18, no. 2, pp. 1218-1232, June 2021, doi: 10.1109/TNSM.2021.3071441.
[20] Guo, L., Wu, Q., Liu, S. et al. Deep learning-based real-time VPN encrypted traffic identification methods. J Real-Time Image Proc 17, 103–114 (2020). https://doi.org/10.1007/s11554-019-00930-6
[21] Wang, W., Zhu, M., Wang, J., Zeng, X., & Yang, Z. (2017). End-to-end encrypted traffic classification with one-dimensional convolution neural networks. 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), 43-48.
[22] Bagui, Sikha & Fang, Xingang & Kalaimannan, Ezhil & Bagui, Subhash & Sheehan, Joseph. (2017). Comparison of machine-learning algorithms for classification of VPN network traffic flow using time-related features. Journal of Cyber Security Technology. 1. 1-19. 10.1080/23742917.2017.1321891.
[23] Chen, Z. , Zhang, P. , Liu, Q. , Guo, L. (2018). 'Web Proxy Detection via Bipartite Graphs and One-Mode Projections'. World Academy of Science, Engineering and Technology, Open Science Index 138, International Journal of Information and Communication Engineering, 12(6), 459 - 466.