Deep Learning Classifiers for QoS and Traffic Engineering in Community Networks

Jonathan Tooke
University of Cape Town
Cape Town, South Africa

ABSTRACT

The aim of this project is to build a deep learning network traffic classifier for the purpose of Quality of Service (QoS) in community networks. Community networks have emerged as a promising solution in addressing internet connectivity gaps in rural areas across the globe. Although massively successful, these networks are not typically capable of providing the same internet quality that users in urban areas enjoy. One way of improving the experience of internet users in community networks is through the optimization of traffic engineering and QoS at the network access point. Critical to the success of these algorithms is the ability of the router to successfully classify network traffic. Deep learning techniques have emerged as a favourable means of performing this classification, particularly with their ability to classify unsupervised data and their relative success at correctly classifying encrypted traffic. This literature review will survey the deep learning techniques that have been applied to the network traffic classification problem in other papers and report on their results and insights. This will be done while considering the requirements of and the eventual goal of implementing a deep learning technique for the purpose of QoS in community networks. Finally, the conclusion will be drawn that using a CNN seems to be most appropriate for the traffic classification task, although not conclusive as various factors aside from the choice of model effect the eventual accuracy of the result, and LSTM and SAE models also showed strong classification results.

CCS CONCEPTS

• **Networks**; • **Computing methodologies** → **Neural networks**;

KEYWORDS

deep learning, neural networks, network classification, community networks, quality of service

1 INTRODUCTION

Network traffic classification is a problem that has undergone a number of evolutionary steps in line with the improvements in network security standards and the growth of computational power. Improvements in network security such as the reduction in dedicated application ports and an increase in the prevalence of encryption have made traditional classification algorithms less effective at best, and obsolete at worst [12]. At the same time, the growth in computational power and big data resulted in the ever-increasing popularity of traditional machine learning frameworks. However, even traditional machine learning frameworks have declined in popularity in recent years because they depend on domain experts manually selecting important features for training of models which is a time-consuming, error-prone and costly process that needs to be redone each time the classification task changes [9]. Their decline has given way to the usage of deep learning frameworks which are capable of learning patterns in unsupervised data, removing the need for manual feature selection. Numerous studies have investigated the application of deep learning to the classification of encrypted network traffic and have demonstrated significant success.

This literature review is the first step in a project that will tackle the challenge of classifying network traffic in community networks using deep learning algorithms for the purpose of traffic engineering and QoS. The emphasis of this study will fall on the classification of online and encrypted traffic. The consideration of hardware constraints in community networks will be of particular importance in the eventual selection and development of a deep learning framework that must be able to classify online traffic without significant computational or storage overhead. This process will initially involve assessing studies already done in the field of encrypted network traffic classification. Thereafter, an informed decision will be made on a path forward. This will involve selecting and designing a deep learning model which will be trained and tested on data that has been collected from the iNethi community network, a South African community network.

The deep learning techniques that will be considered in this literature review are Multilayer Perceptrons (MLP), Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN), and Autoencorders (AE). Previous network traffic classification studies have made use of various forms of these techniques, with CNNs emerging as the most popular across the papers surveyed in this literature review, although sometimes combined with another deep learning technique. Authors have cited its natural strength at other classification problems, such as image classification [8] as well as its ability to decrease the number of learnable parameters [5, 11] as two compelling reasons for making use of a CNN in the traffic classification problem.

High accuracy in network traffic classification greatly improves the success of QoS which prioritize traffic based on traffic class [9, 12]. It will be important to develop a model specific to community network requirements to ensure best performance. In this regard, the model will be need to be trained on network traffic data representative of community networks and the hardware constraints faced by the community network must be considered in selecting a model.

2 COMMUNITY NETWORKS

Community networks are a solution to providing internet access in rural areas across the globe. They typically involve a small group of people coming together to develop a network infrastructure in their local communities and then creating some sort of access point to

connect the network to the internet [10]. These networks are sometimes supported by non-profit organizations in conjunction with local stakeholders. Community networks face varying hardware constraints, but most are built with cheap and simple hardware [4]. This needs to be kept in mind when selecting a deep learning model for network traffic classification. The data that will be used for the eventual training of the deep learning model in this project will come from an iNethi community network based in South Africa.

3 QUALITY OF SERVICE (QOS)

QoS is a term that refers to a network tool that is employed to manage data traffic with the goal of reducing latency, packet loss and jitter on a network. This works by prioritizing some network packets over others in accordance with the application's specific requirements [2]. A successful QoS configuration will translate to a better user experience on average for users on a given network. In order to prioritize some packets over others, rules must be developed that allow for some packets to be forwarded before others depending on the application or application class. The router can only do this if it can successfully classify the traffic. This is the justification behind dedicating significant research efforts, including this project, towards the task of creating an accurate traffic classifier.

4 DATA GATHERING AND SELECTION

4.1 Data Requirements

Two data set characteristics are particularly important to consider when training a deep learning model. The first is the quantity of the data and the second is the quality. These two characteristics are complementary to each other and are both of critical importance.

The surge in the *quantity* of available data over the last few decades has allowed for the training of increasingly powerful deep learning models. For the task of traffic classification, this data usually comes in the form of network traffic logs. Despite it being a fairly straightforward task to collect masses of data at network access points, this strategy faces limitations. The main limitation is that all of the traffic has to be labelled which is not possible after the traffic has been encrypted. Traditional methods of labelling traffic data such as Deep Packet Inspection (DPI) are made largely ineffective by the encryption of the majority of the packet data [12].

A common solution to the problem of labelling data is to capture data in a controlled environment on the client-side where each class is sampled independently and subsequently labelled. If this is not possible, some DPI tools have been developed in recent years such as nDPI which can classify encrypted data by classifying the server of the initial key exchange when a connection is established [6]. BlindBox is another traffic classifier which can only classify encrypted HTPPS traffic [14]. The main problem with this approach is that the accuracy of the deep learning model is limited to the accuracy of the DPI classification [12], removing any justification of using a deep learning model in the first place.

Another limitation to using organically generated data is that certain classes of data may be under-sampled, making the deep learning model less effective at classifying data from those classes. There are a number of ways that this problem can be addressed. The simplest is to capture the data in a controlled environment and ensure that all classes are equally represented. Another approach that has been used is to randomly remove data from the over-sampled classes sufficient data is relatively balanced [9]. Another highly effective solution to this problem was provided in [15] where an Auxiliary Classifier Generative Adversarial Network was used to generate synthesized data for the minor classes. Lastly, the option of using scripts to generate additional data has been explored by some studies, but these scripts struggle with accuracy and may ultimately hinder the ability of the network to classify the data correctly [11].

Additionally, it is important that the data collected is of high *quality*. There are natural faults in network data such as duplicated and out of order packets that can be handled in a pre-processing phase if deemed important [11]. However, of greater concern is poor data quality that cannot be cleaned in this way. The most important thing to ensure is that the sample data is representative of the data that the classifier would actually be running on in production. The model may perform well on the sample test data, but if the sample test data is not representative of the production data, no real conclusions can be drawn on the effectiveness of the model in production. One way of guarding against this is to test the model on a test set that comes from a different configuration to that of the training set [12].

4.2 Available Features

The first generations of traffic classifiers worked by classifying traffic based on the port number that the application used or by using a packet inspection technique using application layer data [12]. These methods have declined in effectiveness as newer applications have begun disguising their usage of port numbers and network encryption standards have made packet inspection techniques less effective, removing them from the available features under consideration. However, there is one notable exception to the encryption of payload data. The first few packets which contain the handshake information are not encrypted and provide some information that can be effective in helping to classify the traffic [12].

The next set of features to be considered are time series features. These features include the inter-arrival time, direction of consecutive packets, and packet length which are independent of traffic encryption[12]. Most deep learning traffic classification models make use of time series features and they have been shown to be highly successful. Time series features can be used for online traffic classification after the first n packets in a flow have been collected and analyzed.

Statistical features can also be used for classification of encrypted traffic, although they are generally more effective on offline traffic as they depend on extracting statistical features from a traffic flow [12]. This makes statistical features less significant for the purpose of QoS which requires online traffic classification.

Lastly, in [15] a byte-representation of the entire packet is used as input data for a CNN to great success. This is done by extracting the relevant information available in the unencrypted layers of the packets. Typically, payload data above level 4 is left unencrypted [12] and Deep Packet takes advantage of this data to perform a highly effective packet-level classification.

5 CLASSIFICATION

In deciding how to classify network traffic, two important decisions need to be made. The first is on the classification type and the second is on the classification process. Classification type refers to the categories that the traffic needs to be classified into. Although not the only options, the two most common classification types are application and application type. Classification process is the decision between classifying based on flow data or on a packet level. Consideration for classification type and process will be further considered below.

5.1 Type: Application versus Application Type

Application classification refers to the ability to classify traffic by its application (e.g. Skype) while application type classification refers to the ability to classify traffic by its application type (e.g. VoIP). The choice of classification type for the purposes of QoS depends on two main things, firstly, the classification type that is most beneficial to network users, and secondly, the accuracy that can be obtained for the chosen classification task. The answer to the first consideration in our situation depends largely on network habits by community network users. If, for example, community network users have any specific applications in a class of applications that they believe should be prioritized over other applications in the same class, this would only be possible with application classification. The other thing to consider is that the model will only be able to classify data on the applications that it has been trained on, which is inherently restrictive, whereas application type classification has been shown to successfully classify traffic by application type on applications that it has never seen before [13].

The accuracy consideration is also important. Deep learning models may be better at the (specific) classification task of recognizing patterns in an application traffic than the (general) classification task of recognizing patterns in application type traffic. Some studies have shown this to be the case when the two classification tasks were compared with each other on the same data set and with the same deep learning framework [15], however this may ultimately depend on the data set and the features that are being used for classification.

5.2 Process: Flow versus Packet

The main decision in classification process is between classifying traffic on a flow or packet level. A flow is determined by a 5-tuple consisting of the source IP, destination IP, source port, destination port and protocol [12]. A flow will contain multiple packets that are sent between an application and the server. The advantage of classifying by flow is that more features are available by virtue of it containing multiple packets, allowing for time series features to be extracted and used for classification. The fact that more features are available should lead to a more accurate classification. Importantly, the entire flow does not have to be transmitted for classification to work and some studies have used the first n packets to classify a flow and thus all subsequent packets in the flow. One downside to this process is that it introduces a storage overhead as the router has to keep a record of the first n packets of all flows before it can classify the flow. Another downside to this approach is that QoS cannot be applied to individual packets until a flow is classified, meaning that the first n packets in all flows will not be optimized for QoS.

Packet classification bears none of the disadvantages of flow classification as packets do not need to be stored before classification and packets can be classified immediately to benefit from QoS optimization. The main disadvantage of packet classification is that time series features cannot be used for classification. However, one study showed that using a CNN on relevant data above level 4 (the encryption level), packet based classification can perform exceptionally well, achieving a 98% accuracy on application classification and a 93% accuracy on application type [15].

6 RELEVANT DEEP LEARNING TECHNIQUES

In this section, the deep learning techniques commonly used for the purpose of traffic classification will be introduced. This will include an explanation of how each deep learning technique works and an overview on how they have been implemented in other traffic classification studies. Further on, in the discussions section, these techniques will be compared with each other for their relative strengths and weaknesses in the context of QoS in community networks.

6.1 Multi-Layer Perceptron (MLP)

Multi-Layer perceptron (MLP) is the original neural network structure. Its architecture consists of an input layer, an output layer, and zero or more hidden layers which are connected between the input and output layer. Each layer consists of one or more neurons with every neuron in a layer connected to every other neuron in the following layer. A neuron takes a weighted sum of inputs (generated by the output of the neurons in the previous layer), adds a bias value, and passes the result through a non-linear activation function. The non-linear activation function produces an output which is passed to the neurons in the next layer where this process repeats itself until the output layer is reached and the network result is produced. In the training phase of the neural network, the output will be compared to the expected output and if the output is incorrect, the weights and biases in the network will be adjusted using a backpropagation algorithm. The main problem with MLP is the enormous number of parameters that need to be learned in the training process. This makes training inefficient, particularly for highly complicated tasks where more hidden layers and neurons are used.

MLP has not been extensively used for the purpose of traffic classification. This is likely due to the training problem mentioned above. However, there are a few papers that have investigated their usage as a benchmark or out of interest. In [1], various deep learning frameworks are trained and compared for the purpose of traffic classification, including several different MLP architectures. The MLP models performs adequately, but not as well as the CNN model. In [3] a Bayesian Neural Network is implemented for the purpose of the traffic classification problem to some degree of success. This uses the same architecture as an MLP model, but each node represents an event and each edge represents a probability, making the logic behind its classification significantly different.

6.2 Convolutional Neural Network (CNN)

Convolutional neural networks (CNNs) also consist of a number of layers, but instead of hidden layers between the input and output layers, they make use of convolutional layers. The convolutional layers consist of a number of kernels which are used repeatedly across all the subsections of the input to produce the output which is fed to the next layer. By reusing the same kernels across the entire input, the number of learnable parameters are greatly reduced, decreasing the computational time required in training these models as compared to MLP.

In [14], the authors make use of a 1D-CNN to classify network traffic at a packet level and they demonstrate significant success. They justify selecting a CNN due to its ability to capture spacial dependencies between subsequent bytes in the packets which results in effective traffic classification. In [11], a CNN model is also used for the traffic classification problem. The authors mention that another advantage of using a CNN for traffic classification is that CNNs are shift invariant, meaning that patterns can be detected by a CNN even if areas of the input are shifted around.

The authors in [13] and [5] take an alternative approach. They record the first few packets in a flow and convert them to an image representation before passing the image representation to the 2D-CNN as input. They both perform very well, with [13] even succeeding in identifying VPN and Tor traffic with a high level of accuracy. The main challenge with this approach is deciding on the process for the image encoding. This encoding process will need to be custom-built according to the available features and the classification task.

In [17] the authors also convert the traffic data to an image representation before feeding it through a CNN. They use a 1D-CNN and a 2D-CNN separately on the same data set to see which CNN model will perform better at the classification task. Their results indicate that the 1D-CNN performed better, but it is important to note that this result is dependent on the process that they used for encoding their image representation as well as some of the other hyperparameters used when constructing the CNN. Furthermore, their better-performing 1D-CNN failed to outperform the 2D-CNN constructed in [13] and can therefore not be considered as conclusive evidence that a 1D-CNN is better suited to the traffic classification task.

6.3 Recurrent Neural Network (RNN)

A Recurrent Neural Network (RNN) is a neural network that stores temporal information, allowing it to use information about previous inputs when analyzing the current input. Traditional RNN models faced various training issues, including gradient vanishing and exploding, that have been mitigated by the development of the LSTM model, a customized RNN that does not face these issues. Therefore the focus on RNN models will be narrowed to LSTM models, which are predominantly used in the available literature.

In [18], a deep learning classifier is built for the purpose of flow-based traffic classification. Three models are considered, a CNN, an LSTM, and a Stacked Auto-Encoder (SAE). All of the models achieve a roughly 99% classification accuracy on encrypted data, with the LSTM performing only slightly worse than the CNN and slightly better than the SAE. This shows that the LSTM is a model worth considering for the traffic classification task.

The authors in [8] and [16] make use of a model that combines CNN and LSTM architectures to capture both spacial and temporal features in the data. They both report considerable accuracy. This shows that models should not only be considered in isolation and that combinations of different deep learning techniques can be just as, if not more, effective.

6.4 Autoencoder (AE)

An autoencoder (AE) is a form of neural network that tries to reconstruct the input at the output layer after the data has passed through the network. This is done with considerably less neurons in the hidden layer as compared to the output layer. AEs are frequently used as a dimensionality reduction or feature extraction technique. They are also sometimes used for the purpose of initializing the weights in a different deep learning architecture.

In [7], the authors make use of another function of autoencoders. They use a neural autoencoder to form a form of classification by clustering. The classification accuracy is relatively compared as compared to the studies done with alternative architectures, achieving an average F1 score of 76%. In [9] the authors make use of a more powerful autoencoder architecture known as a stacked autoencoder (SAE). An SAE is constructed by stacking a number of autoencoders one after the other where the output of the one autoencoder is the input for the next one. In the study, the SAE is compared with a CNN for the purpose of packet-level classification. The SAE performs slightly worse than the CNN for both application and application type classification, but still achieves an F1 score above 90% for both of these tasks.

7 DISCUSSION

In this section of the paper, the requirements for QoS in community networks will be considered along with the process that should be followed to build an appropriate deep learning traffic classifier. This will include a discussion on data gathering and selection, the classification process and type, and the selection of a deep learning model.

7.1 Data Gathering and Selection

In the data requirements section, some of the complications experienced in gathering network traffic were introduced and various data collection techniques were compared. The conclusion was drawn that gathering the data in controlled environment would be best for ensuring the accuracy of the labels and sufficient sampling from all of the classes. Some of the problems associated with labelling the data after-the-fact were discussed, the biggest of which is that the accuracy of a deep learning model would be limited to the accuracy of the application used for labelling. Some options for correcting the imbalance problem in class sampling were introduced, as well as some data cleaning techniques. An overview of the available features was provided and it was shown that time series and unencrypted payload data were most important for online, encrypted traffic classification.

7.2 Classification

The concepts of classification type and process were introduced. For *classification type* the choice is between application type classification and application classification. In the absence of significant data on user preferences for specific applications, it seems most appropriate to classify by application type. One of the most compelling factors in selecting type classification is that it has been shown to successfully classify traffic by application type on applications that it has never seen before, reducing the need to retrain the model as new applications are released [13]. For *classification process*, the choice is between flow-based classification and packet-based classification. The advantage of packet level classification is that it can classify packets immediately and that it has a lower storage requirement. The main downside is that there is less data available in an individual packet than a flow, potentially limiting the accuracy of the model. However, in [6], it was shown that a very high classification accuracy can be obtained on a packet level, making classification by packet a much more compelling option.

7.3 Deep Learning Model Comparison

The first major observation made was that MLP was rarely used for the purpose of traffic classification. One study discussed had implemented an MLP model, but found that it was not very successful [1]. MLP does not pose inherent problems with regard to computational power or storage in community networks, but the poor accuracy achieved in the study assessed and the inherent training restrictions of MLP models makes a compelling case for the removal of MLP from consideration for this task.

CNN was the next model considered. CNNs have been used widely across the literature for the network traffic classification task and have emerged as the most popular model for this task. Various formats of the CNN model were considered, including flow level classification where the data is converted into an image representation [5, 13], and packet-level representation where the packet data is represented as a one dimensional array of bytes [9]. In studies where a CNN model was compared with other models, CNN models performed better on average [1, 9, 18]. It is challenging to draw direct conclusions about what CNN structure was the most effective as the studies do not use the same data set or the same representation of the data and can therefore not be directly compared, but what is clear is that CNNs are well suited to the traffic classification task and should be considered. Furthermore, they do not introduce significant storage or computational overhead, provided the model is not too large, making them an appropriate choice for QoS in community networks.

RNNs (and the LSTM model in particular) were then considered. There have not been as many studies done on the LSTM model as the CNN, but the few studies that have been performed have shown the LSTM to be a highly effective model. An interesting architecture for the LSTM is presented in [8, 18] where it is combined with a CNN model and this appears to be a better option than using an LSTM model on its own. The only concerns with this approach is the additional computational complexity involved as well as the additional storage requirements of an LSTM model which may pose as an issue for community networks.

Lastly, the AE was considered. On its own, it was not effective at classifying the traffic data [7], but an augmented form of the autoencoder called the SAE [9] showed good results. However, the accuracy was not as high as for the CNN in the same study, making it not as compelling an option. Two other interesting applications of autoencoders are for network weight initialization and for dimensionality reduction and these could be useful in combination with an LSTM or CNN model and should be investigated further.

8 CONCLUSIONS

In this paper, the requirements for QoS in community networks were discussed and the need for an effective network traffic classifier was emphasised. Not only should this traffic classifier report high classification accuracy on encrypted, online traffic, but it must also consider the hardware constraints of community networks with respect to storage and computational power. Thereafter, an overview of the features that can be used for classification with encrypted traffic was provided, as well as some important considerations for selecting the data that will be used for training the model. Finally, a number of deep learning techniques that have been used for the purpose of encrypted traffic classification in other studies were introduced and their structure reported on, while highlighting some of the results produced by the studies.

In the discussions section the various studies were compared along with their applicability to the task of classifying online, encrypted network traffic for the purpose of QoS in community networks. The need for accurate labels in the training data was emphasised, and highlighted that time series and payload data were the features most applicable to the task of classifying online, encrypted traffic. The conclusion was made that application type classification and packet-level classification were the most compelling choices for classification type and process respectively.

Finally, the various deep learning models were critically compared in the context of this project and a number of conclusions were drawn. Firstly, that MLP was not a model that should be considered due to difficulties in training with a vast number of learnable parameters and the limited effectiveness as a result. CNN emerged as perhaps the most studied and most effective model, making it a compelling choice for implementing the community network traffic classifier. LSTM and SAE models showed promise, but have not been shown to outperform CNN models in any studies where they were compared. The additional complexity of the LSTM model is also cause for concern in community networks.

Moving forward, the most compelling option is to use a CNN model as it has been shown to perform well across multiple studies and outperformed the other models in studies where they were compared. The effectiveness of an autoencoder for the purpose of dimensionality reduction would also be an interesting option to explore before feeding the compressed features into the CNN model. This does not seem to be something that papers have tried before and could be considered as a new contribution to the literature.

REFERENCES

[1] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé. 2019. Mobile Encrypted Traffic Classification Using Deep Learning: Experimental Evaluation, Lessons Learned, and Challenges. *IEEE Transactions on Network and Service Management* 16, 2 (2019), 445–458.

[2] A. O. Adedayo and B. Twala. 2017. QoS functionality in software defined network. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)*. 693–699.

[3] T. Auld, A. W. Moore, and S. F. Gull. 2007. Bayesian Neural Networks for Internet Traffic Classification. *IEEE Transactions on Neural Networks* 18, 1 (2007), 223–239.

[4] Bart Braem, Chris Blondia, Christoph Barz, Henning Rogge, Felix Freitag, Leandro Navarro, Joseph Bonicioli, Stavros Papathanasiou, Pau Escrich, Roger Baig Viñas, Aaron L. Kaplan, Axel Neumann, Ivan Vilata i Balaguer, Blaine Tatum, and Malcolm Matson. 2013. A Case for Research with and on Community Networks. 43, 3 (2013).

[5] Z. Chen, K. He, J. Li, and Y. Geng. 2017. Seq2Img: A sequence-to-image based approach towards IP traffic classification using convolutional neural networks. In *2017 IEEE International Conference on Big Data (Big Data)*. 1271–1276.

[6] L. Deri, M. Martinelli, T. Bujlow, and A. Cardigliano. 2014. nDPI: Open-source high-speed deep packet inspection. In *2014 International Wireless Communications and Mobile Computing Conference (IWCMC)*. 617–622.

[7] J. Höchst, L. Baumgärtner, M. Hollick, and B. Freisleben. 2017. Unsupervised Traffic Flow Classification Using a Neural Autoencoder. In *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*. 523–526.

[8] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret. 2017. Network Traffic Classifier With Convolutional and Recurrent Neural Networks for Internet of Things. *IEEE Access* 5 (2017), 18042–18050.

[9] Mohammad Lotfollahi, Mahdi Jafari Siavoshani, Ramin Shirali Hossein Zade, and Mohammdsadegh Saberian. 2019. Deep packet: a novel approach for encrypted traffic classification using deep learning. *Soft Computing* 24, 3 (2019), 1999–2012. https://doi.org/10.1007/s00500-019-04030-2

[10] P. Micholia, M. Karaliopoulos, I. Koutsopoulos, L. Navarro, R. Baig Vias, D. Boucas, M. Michalis, and P. Antoniadis. 2018. Community Networks and Sustainability: A Survey of Perceptions, Practices, and Proposed Solutions. *IEEE Communications Surveys Tutorials* 20, 4 (2018), 3581–3606.

[11] Shahbaz Rezaei and Xin Liu. 2018. How to Achieve High Classification Accuracy with Just a Few Labels: A Semi-supervised Approach Using Sampled Packets. *CoRR* abs/1812.09761 (2018). arXiv:1812.09761 http://arxiv.org/abs/1812.09761

[12] S. Rezaei and X. Liu. 2019. Deep Learning for Encrypted Traffic Classification: An Overview. *IEEE Communications Magazine* 57, 5 (2019), 76–81.

[13] T. Shapira and Y. Shavitt. 2019. FlowPic: Encrypted Internet Traffic Classification is as Easy as Image Recognition. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 680–687.

[14] Justine Sherry, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. 2015. Blind-Box: Deep Packet Inspection over Encrypted Traffic. *SIGCOMM Comput. Commun. Rev.* 45, 4 (Aug. 2015), 213–226. https://doi.org/10.1145/2829988.2787502

[15] Ly Vu, Cong Thanh Bui, and Quang Uy Nguyen. 2017. A Deep Learning Based Method for Handling Imbalanced Problem in Network Traffic Classification. In *Proceedings of the Eighth International Symposium on Information and Communication Technology* (Nha Trang City, Viet Nam) *(SoICT 2017)*. Association for Computing Machinery, New York, NY, USA, 333–339. https://doi.org/10.1145/3155133.3155175

[16] W. Wang, Y. Sheng, J. Wang, X. Zeng, X. Ye, Y. Huang, and M. Zhu. 2018. HAST-IDS: Learning Hierarchical Spatial-Temporal Features Using Deep Neural Networks to Improve Intrusion Detection. *IEEE Access* 6 (2018), 1792–1806.

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

[18] Y. Zeng, H. Gu, W. Wei, and Y. Guo. 2019. $Deep - Full - Range$: A Deep Learning Based Network Encrypted Traffic Classification and Intrusion Detection Framework. *IEEE Access* 7 (2019), 45182–45190.