A novel QUIC traffic Classifier based on Convolutional Neural Networks

Van TONG*, Hai Anh TRAN†, Sami SOUIHI* and Abdelhamid MELLOUK*
*LISSI-TincNET Research Team
University of Paris-Est Crteil (UPEC), France
Email: van.tong@tincnet.fr, sami.souihi@u-pec.fr, mellouk@u-pec.fr
†Bach Khoa Cybersecurity Centre
Ha Noi University of Science and Technology, Hanoi, Vietnam
Email: anhth@soict.hust.edu.vn

***Abstract*—Nowadays, network traffic classification plays an important role in many fields including network management, intrusion detection system, malware detection system, etc. Most of the previous research work concentrates on features extracted in the context of non-encrypted network traffic. However, these features are not compatible with all kind of traffic characterization. Google's QUIC protocol (Quick UDP Internet Connection protocol) is developed robustly and implemented in many services of Google. Nevertheless, the emergence of this protocol imposes many obstacles for traffic classification due to the reduction of visibility for operators into network traffic, so the port and payload-based traditional methods cannot be applied to identify the QUIC-based services. To address this issue, we proposed a novel technique for traffic classification based on the convolutional neural network which combines the feature extraction and classification phase into one system. The proposed method uses the NetFlow and packet-based features to improve the performance. In comparison with current methods, the proposed method can identify some kind of QUIC-based services such as Google Hangout Chat, Google Hangout Voice Call, YouTube, File transfer and Google play music. Besides, the proposed method can achieve the micro-averaging F1-score of 99.24 percent.**

***Index Terms*—Traffic classification, QUIC, CNN, NetFlow**

I. Introduction

Traffic classification [1] [2] [3] plays an important role in troubleshooting the network issues for Internet service provider (ISPs). Network operator need to define the type of services in their network to quickly react to different issues in the support of various enterprise goals. Traffic classification can be used in intrusion detection systems [4] for detection of denial of service attacks, botnet detection [5], customer's usage identification, etc. Besides, traffic classification is used for the estimation of capacity for the network systems, adaptive network based the QoS (Quality of Service) of network traffic or lawful interception.

The traffic classification approaches are divided into three main categories: port-based, payload-based and flow static-based methods. The port-based method which uses TCP or UDP port numbers is one of the fastest and simplest approach to continuous network monitoring. However, the accuracy is not high because many services use unpredictable ports [6], ports used by another application or the use of network address port translation. To solve the drawbacks of port-based flow traffic classification, the payload-based method is proposed. This method inspects the packet's payload to identify the signature or syntax of each application's packet payload. The payload-based methods provide the high detection rate, but this approach is limited by some issues related to updating the signature. With the rapid development of machine learning, many research works investigate and apply the flow static-based method [7]–[9] to classify the network traffic. This approach relies on the information obtained from the packets such as the number of byte in packet payload, packet interval time, the direction of the packet, etc. Then, these features are trained in the machine learning algorithms to classify the network traffic into different kind of services. The flow static-based method can deal with the issues of port and payload-based methods including the invisibility of payload or dynamic port.

Recently, Google has developed Quick UDP Internet Connection (QUIC) [10]–[12], a new transport layer network protocol on the top of UDP. QUIC has many advantages related to connection establishment, congestion control, multiplexing without head of line blocking and connection migration. Besides, QUIC also provides the security protection equivalent to TLS (Transport Layer Security). The emergency of QUIC has made the traditional traffic classification methods, especially port and payload-based methods, unsuitable.

In this paper, we propose a new flow static-based method based on convolutional neural network (CNN) to detect different QUIC-based Google's services including Google Hangout (Chat, Voice call), File transfer, YouTube and Google Play Music. First, we apply NetFlow-based features and random forest algorithm to define the Google Hangout' services due to the difference of network flow of these services. Then, the second stage uses the packet-based features and convolutional neural network to classify the remainder into file transfer, YouTube or Google play music.

The remainder of the paper is as follows. Section II presents some related works and background related to traffic classification and CNN. We introduce some characteristic of QUIC-based services in Section III. Section IV describes our ap-

proach including the detail on NetFlow-based feature, packet-based feature and CNN-based traffic classification method. Section V shows experimental results and discusses some related contents. The paper concludes with Section VI which highlights some future work.

II. Background and Related work

A. Convolutional network

Convolutional network [13] [14] is known as convolutional neural networks (CNN), a specialized artificial neural network for data processing. It can be applied in some applications such as pattern recognition, DGA botnet detection or traffic classification [3]. Artificial neural network (ANN) [14] is computing system inspired by the biological neural networks. ANN comprises a high number of interconnected nodes to learn the input and optimize the final output. The basic structure of ANN contains three main layers consisting of input layer, hidden layer, and output layer. The hidden layer will make the decision from previous layer and weight in order to improve the final output.

CNN comprises three main layers including convolutional layer, polling layer, and fully-connected layer. The convolutional layer is the core building block of CNN. This layer consists of the various kernels which have a small receptive field. The important characteristic is that the receptive field will be extended through the full depth of the input. We can detect one kind of feature of the input with one kernel, so the convolutional layer with many kernels can extract a variety of features at many spatial locations. The pooling layer replaces the output with the summary statistic of nearby output. There are two main pooling functions including max pooling and average pooling function. The pooling layer help to make the representation become approximately invariant to small translation of the input. In fully-connected layer, each unit has the connections to all activation of previous layers. It is the multi-layer perceptron neural network (MLP) [15] which illustrate the relationship between the previous layer with the output.

B. Related work

In this section, we present some current related work in the area of NetFlow-based method and convolutional neural network. Besides, we describe the motivation to consider the traffic classification using NetFlow-based features, packet-based features, and convolutional neural network.

Williams et al. [7] presented the traffic classification using machine learning algorithms to detect the FTP, Telnet, SMTP, DNS, and HTTP. Some NetFlow-based features are extracted to classify the network flows containing the protocol, flow duration, flow volume in bytes and packets, packet length, and inter-arrival time. Besides, they also investigate the performance of different machine learning algorithms including Naive Bayes, C4.5, Bayesian Network and Naive Bayes Tree.

Bashir1 et al. [16] proposed the NetFlow-based approach to identify the BitTorrent activities. Although the performance achieves over 90 percent, this approach only concentrates on BitTorrent applications.

Lotfollahi et al. [17] proposed "Deep Packet" to classify the network traffic into different kind of traffic characterization including email, chat, FTP, Skype, torrent, etc. They extract the features from the packets and feed into the convolutional neural network to identify the traffic characterization. However, the using of 1500 bytes in the packets can lead to uncertain results because there are some similar attributes in the same services.

Lopez-Martin et al. [3] presented the new traffic classification based on the convolutional neural network to detect and classify 108 services. In this research work, each flow comprises 20 packets, and each packet contains 6 features including source port, destination port, the number of bytes in packet payload, TCP window size, inter-arrival time and direction of the packet. However, the authors use the destination port which is similar to the same services to identify the different services in the network.

All of the research work presented above only consider and investigate the traffic classification using the convolutional neural network and NetFlow-based methods in the context of non-encrypted network traffic. Recently, the emergence of QUIC results in many obstacles for traffic classification because the payload of the packet is encrypted. Besides, all services using QUIC comprise the similar port (443), so some traditional approaches cannot apply to identify various kind of services in the network.

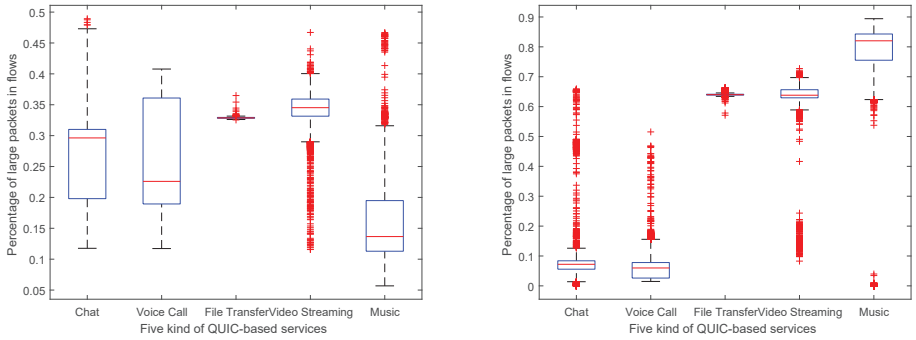
In this paper, we propose the **new traffic classification based on the convolutional neural network** which consists of two main traffic classification stage and combines the **netflow and packet-based features** to predict the **QUIC-based services**. This approach can help to **identify the services** and construct the **adaptive network based on QoS**.

III. Characteristic of some QUIC's services

It is expected that there is the difference between some QUIC-based services (Fig. 1). In this paper, we use the collected dataset [18] containing Voice call, Chat (Google Hangout), Video streaming (YouTube), Google play music and File transfer.

When we capture the network flow traffic, we found some interesting characteristic of QUIC-based services. In Voice call of Google Hangout services, lots of UDP packets exist in the network flows when an end-user call to some automatic PBX phone number. I think that Google Hangout only uses QUIC in the front-end to establish and finish the connections. When the connection is established, the data is transferred between two end-users using UDP protocol. The reason for it is that QUIC has recently implemented in some Google's services, so the server of the companies whose automatic PBX phone number belong to is not implemented and supported QUIC.

In the other services, QUIC is used for establishing, finishing the connections and transferring data between end-users. However, there is some significant difference between these



(a) Small packets.

(b) Large packets.

Fig. 1: Percentage of small and large packets in flows.

services. In Chat of Google Hangout services, all packets transferred between end-users are supported by QUIC, but the data transferred is not sufficient. Therefore, the majority of packets in flows is small packets, which the length of packets is less than 150 bytes. It is similar to Voice call service of Google Hangout. In the Fig. 1a and 1b, the average percentage of small and large packets in the flows of Chat and Voice call are smaller than the figure for others. However, the standard deviation of the percentage of small packets in flows of Chat and Voice call services are larger than the figure for three other services.

In the file transfer, video streaming and Google play music, the server and end-users transfer large data frequently, so there is the high number of large packets, which the length of packets is bigger than 1000 bytes, in the network flows. As in Fig. 1b, the average of large packets in the flows of these services are over 60 percent. Moreover, the end-users frequently send the small packets to the servers, so the number of small packets of these services is approximately 30 percent, except for Google play music (approximately 15 percent).

IV. The Novel Traffic Classification Method Based on the Convolutional Neural Network

In this paper, we propose a new traffic classification method based on the convolutional neural network which comprises two multiclass classification stages. The first stage using NetFlow-based features to define the Chat and Voice call of Google Hangout. Then the remainder continues to go through the second multiclass classification stages to classify the network flows into the file transfer, video streaming or Google play music.

A. Netflow-based features

In the first multiclass classification stage, we use the NetFlow-based features to identify Chat and Voice call service of Google Hangout. As in section III, we found that there is some difference between five kinds of QUIC-based services, especially packet length and payload length. Therefore, we extract 8 features from the flows:

- From client to server
  - Percentage of small packets in the flows.
  - Percentage of medium packets in the flows.
  - Percentage of large packets in the flows.
  - Average payload length.
- From server to client
  - Percentage of small packets in the flows.
  - Percentage of medium packets in the flows.
  - Percentage of large packets in the flows.
  - Average payload length.

The small packet, medium packet and large packet which the packet length range from 0 to 150, 150 to 1000 and over 1000 bytes, respectively.

B. Pre-processing

In this paper, we use the collected dataset which is captured at data link layer. The dataset will be processed in the pre-processing phase. There are four main steps in the pre-processing phase including data link header removal, byte conversation, normalization and zero padding.

The data-link header contains some information related to the physic layer which plays an important role in forwarding the frames in the network. However, this information is useless for traffic classification, so the data-link header will be filtered in the data link header removal step. Besides, we only use the payload of QUIC packet because we found that other information in QUIC packet is useless for the classification. Then the packet in the dataset will be converted from bit to byte in order to reduce the input size. For better performance, all packet bytes are normalized using dividing by 255, the maximum value for a byte. The CNN requires the same input length while the packet length in the dataset varies from over 50 to 1392 bytes. Therefore, the dataset will be added some zero values in the zero padding step to have the similar length of each packet. The packet with packet length less than 1400, are padded zero at the end. Finally, each packet comprises 1400 values corresponding to 1400 features.

C. The proposed method

In this section, we present the traffic classification based on the convolutional neural network (Fig. 2a). To identify the

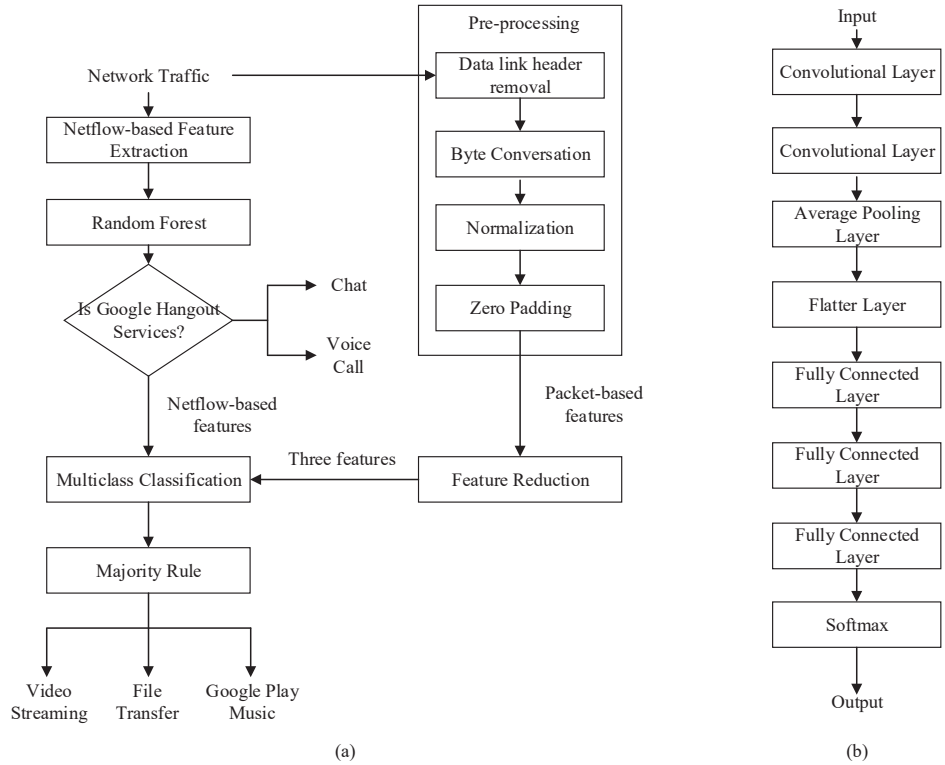


Fig. 2: (a) Traffic classification method using the netflow and packet-based features; (b) The multiclass classification.

Google Hangout services, the network flow will be processed in the NetFlow-based feature extraction. This step extracts 8 NetFlow-based features (as described in the previous subsection) representing for each flow. Then, these features are feed into the random forest algorithm [19] to classify the network flows into the Chat, Voice call service of Google Hangout and the third group (File transfer, Video streaming, and Google play music). As described in section III, there is the difference between the Google Hangout services and the remainder, so we use the random forest as a three-class classifier. Besides, the network traffic is processed in the pre-processing stage to extract 1400 features representing for each packet. Afterward, these packet-based features continue to go to the feature reduction to decrease the size of packet-based features. The feature reduction consists of three main layers including the convolutional layer, average pooling layer, and fully connected layer. This stage converts 1400 features of each packet into three features.

After the first classification of random forest algorithm, the NetFlow-based features and three feature obtained from the feature reduction are combined and transferred to the multi-class classification. The structure of this stage, which is the one-dimensional convolutional neural network, is described as in Fig. 2b. The architecture of one-dimensional convolutional neural network contains five essential layers:

- Convolutional layer: The first convolutional layer is a one-dimensional layer with 550 kernels of size 5, the stride of 1. The input vector has 11 features, so the output of this layer is a two-dimensional tensor with the size of 7 x 550. The second convolutional layer is a one-dimensional layer with 100 kernels of size 4, the stride of 1. The output of this layer is a two-dimensional tensor with the size of 4 x 100.
- Average pooling layer: After going through the first two convolutional layer, the data is feed into the one-dimensional average pooling layer with the size of 2, the stride of 1. The output of this layer is a two-dimensional tensor with the size of 3 x 100.
- Flatter layer: This layer flatten the output obtained from the average pooling layer into a one-dimensional vector with 300 features.
- Fully connected layer: There are three fully connected layers with 100, 500, and 3 neurons, respectively.
- Softmax: This layer is a generalization of the logistic function that squashes an n-dimensional vector into to an n-dimensional vector with element values from 0 to 1.

In CNN, the convolutional layer, average pooling layer, and flatter layer are similar to the feature extraction stage. Besides, the fully connected layer and softmax are the classifier that indicates the relationship between the input feature and the output to classify the network flows into different kind of services.

In multiclass classification, we combine the NetFlow-based feature and the packet-based features in order to identify the services. In each flow, we extract the packet-based features

of 10 packets and add the NetFlow-based features into each packet, so each packet is represented by the NetFlow and packet-based features. The majority rule stage classifies the flows based on the ten packets of each flow. The majority rule is the decision rule that selects the alternatives with high votes. If the majority of packets among 10 packets are classified as any service, the flows will be assigned to that service. If the vote of three services is equal, we use the probability of the packet classified to identify the services. The detail of majority rule is described as in Algorithm 1. The label(id) is the label of flows after classification, and $prob_{ij}$ are the probability of a $packet_i$ classified as $service_j$ in the flows. argmax is the function that returns the index of the highest value in the input.

Algorithm 1 Majority rule

Require: id, $FlowId$, $label$, $FlowLabel$, $service$, $Count$, $prob$, $Prob$, a

while id in $FlowId$ **do**
  for j = 1 to 3 **do**
    for i = 1 to 10 **do**
      if $label(id)$ is $service_j$ **then**
        a_{ij} = 1
      else
        a_{ij} = 0
      end if
    end for

$$Count_j = \sum_{i=1}^{10} a_{ij}$$

$$Prob_j = \sum_{i=1}^{10} prob_{ij}$$

  end for
  if all $Count_j$ unique in $Count$ **then**
    $FlowLabel(id)$ = argmax($Count$)
  else
    $FlowLabel(id)$ = argmax($Prob$)
  end if
end while

V. Experimental results

A. Dataset specification

In the experiment, we build the testbed and collect the network flows to build the real dataset [18] during three weeks (starting March 2018). To capture the network flows of video streaming, chat, and voice call services, we use Selenium WebDriver [20] in Google Chrome running on Ubuntu 16.04 OS. Besides, we also use quic-go [21] to transfer some files from server to client using QUIC and capture the network flows of file transfer services. We captured approximately 150GB of network traffic including over 20,000 flows with five kinds of QUIC-based services. The detail is described as in Table I. In the first classification using random forest algorithm, the dataset is divided into 5-fold. Concretely, the training phase comprises 20 percent of the dataset, and the testing phase accounts for 80 percent of the dataset. In the multiclass classification, the dataset is also divided into 5-fold. The training, validation, and testing comprise 45, 5, 50 percent of the dataset, respectively.

TABLE I: Dataset specification.

Services	Number of flows
Chat	2,783
Voice call	2,608
File transfer	4,451
Video streaming	5,844
Google play music	4,349

Our proposed method is written in Python using Keras library [22] and scikit-learn tools [23]. Besides, all experiments are implemented in a workstation (Intel(R) Xeon(R) CPU E5-2640 v3 @ 2.6 GHz and 16 GB of RAM) running Ubuntu 16.04.

B. Performance measures

In this section, we present some measures to evaluate the performance of our proposed method including Precision, Recall, and F1-score. Precision is the percentage of relevant flows that are retrieved, while recall is the percentage of retrieved flows that are relevant. F1-score represents a harmonic mean between precision and recall. The detail of these parameters are described as in equation 1, 2, 3.

$$Precision = \frac{TP}{TP + FP}. \tag{1}$$

$$Recall = \frac{TP}{TP + FN}. \tag{2}$$

$$F1 - score = \frac{2}{1/Recall + 1/Precision}. \tag{3}$$

The quality of overall classification can be evaluated in two ways [24]. In macro-averaging, a metric is averaged over all classes that are treated equally. Micro-averaging is based on the cumulative True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN) of all dataset.

C. Experimental results

In our proposed method, random forest and CNN work together in combination. Its basic motivation is to retain the high accuracy on different kind of QUIC-based services. This section is dedicated to evaluating the performance of our proposed method and highlighting its advantages over different scenarios including different parts of the dataset and a various loss function.

First, we use the random forest algorithm and NetFlow-based features to identify the Google Hangout services containing Chat and Voice call. Table II indicates the performance of the random forest algorithm using netflow-based features. The NetFlow-based features consist of 8 features related

to the proportion of packets in the flows, and the average packet length in the flows (Section III). The performance of Chat and Voice call service are impressive, over 96 percent. However, the recall of Chat service is the lowest with 96.54 percent. As in Fig. 1, Chat, and Voice call service have some similar region, so some flows of Voice call service are identified into Chat service. This leads to the lower result for Chat service. Especially, the performance of third group (file transfer, video streaming, and Google play music) are the highest with over 99.5 percent. The reason for it is that there is the slight difference in 8 NetFlow-based features of Chat, Voice call service, and third group. As a result, the micro and macro-averaging precision, recall and f1-score achieve over 98 percent.

TABLE II: Precision, Recall and F1-score for random forest algorithm using netflow-based features.

Class	*Precision*	*Recall*	*F1-score*	*Number of flows*
Chat	0.9871	0.9654	0.9762	2,227
Voice call	0.9871	0.9880	0.9875	2,086
Third group	0.9957	0.9997	0.9977	11,351
Micro-averaging	0.9934	0.9934	0.9934	15,664
Macro-averaging	0.9899	0.9843	0.9871	

Second, the remainder of the dataset are investigated in two scenarios to define the file transfer, video streaming or Google play music. In the first scenarios, we investigate the influence of different features on our proposed method. We implement five dataset of possible features and appreciate which features indicate the high performance. The Figure 3 indicates five dataset including the first 300, 600, 900, 1,200 and 1,400 features in payload of QUIC packets. There is an upward trend in the micro and macro-averaging precision, recall and f1-score in five datasets. The performance with the dataset of the first 300 and 900 features are not good, only over 70 percent in the micro and macro-averaging f1-score. The precision of video streaming service with the dataset of the first 300 features is 64.34 percent. Some flows of video streaming service are classified as flows of file transfer service, so this reduces the recall of file transfer service (only 26.52 percent) in the experiment of the first 300 features. It is similar to the dataset of the first 900 features. When the number of features increases to over 1,200, the precision, recall, and f1-score are above 94 percent. Especially, the performance with the dataset of 1400 features is the highest with over 97 percent in five kinds of the dataset. The discriminant power in 1400 features are larger than the figure for others, so the micro and macro-averaging precision, recall and f1-score are the highest, with approximately 99 percent.

In the second scenario, we investigate the influence of some loss functions [25] on the convolutional neural network (Figure 4). As above analyzation, the performance of the dataset with 1400 features are the highest in five kinds of the dataset, so we use this dataset in the second scenario. We implement three kind of loss functions containing `categorical_hinge (Hinge)`, `mean_squared_error (MSE)` and `sparse_categorical_crossentropy (SCCE)`. In this scenario, we evaluate the performance of three kinds of loss function to select the appropriate loss function for our proposed method. Hinge loss is the loss function which is notably used in Support Vector Machine (SVM) for ”maximum-margin” classification. In Fig. 4, the performance of hinge loss is not good because the micro and macro-averaging precision, recall, and f1-score are only 20 percent. Especially, the flows of file transfer cannot be detected. Mean squared error is the loss function that measures the average of the squares of the errors. The performance of MSE loss is higher than the figure for hinge loss. Moreover, the flows in video streaming services is lower than the figure for other services. The reason for it is that flows of video streaming are treated as the flows of Google play music, so the recall of Google play music is only 94.73 percent. A noticeable feature is that the performance of SCCE is the highest in three loss function. The precision, recall, and f1-score of three services are over 96 percent. As a result, the micro and macro-averaging of SCCE are the largest with approximate 99 percent.

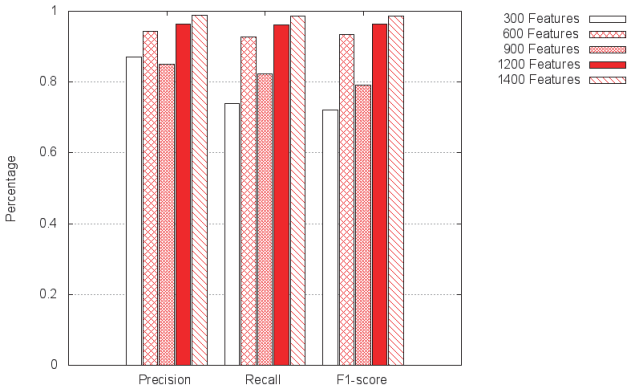


Fig. 3: Macro-averaging precision, macro-averaging recall and macro-averaging f1-score in different datasets.

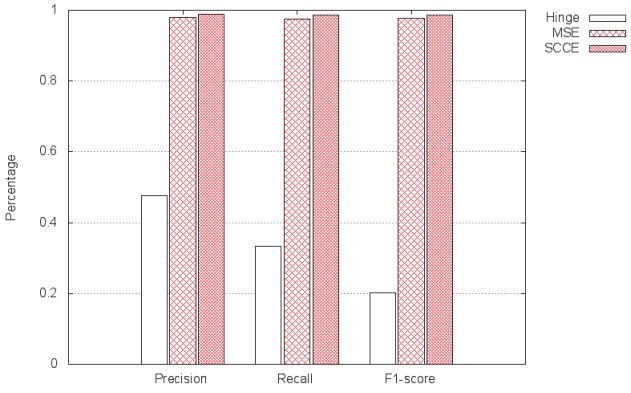


Fig. 4: Macro-averaging precision, macro-averaging recall and macro-averaging f1-score in different loss functions.

Through two scenarios, we found that the dataset with 1400 features and the SCCE loss function are appropriate for our proposed method. Figure 5 indicates the overall results of our proposed methods with the dataset of 1400 features and SCCE loss function on five kinds of QUIC-based services. The micro and macro-averaging of precision, recall, and f1-score are approximately 99 percent, an impressive result. These results are calculated based on scikit-learn [23] using the results of random forest stage and multiclass classification stage.

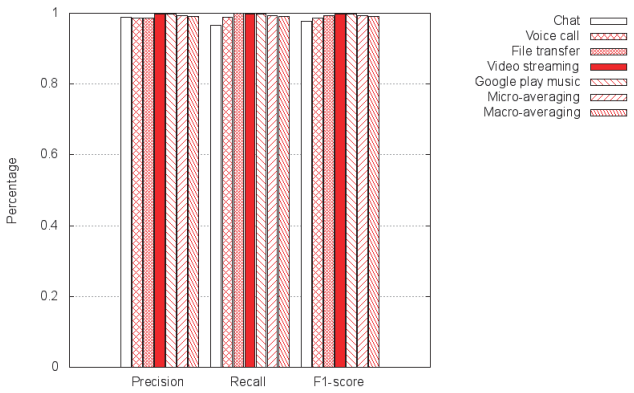


Fig. 5: Precision, Recall and F1-score of our proposed method on five kind of QUIC-based services.

VI. Conclusion

This paper presents the novel traffic classification method using the convolutional neural network, NetFlow, and packet-based features to identify some QUIC-based services. There are two main classification stages in our proposed method. The first stage uses random forest and netflow-based features to detect the Google Hangout services. The second stage uses the convolutional neural network and combines the NetFlow, packet-based features with some alternatives to classify the network flows into video streaming, Google play music or file transfer. The experiments demonstrate that our proposed method can detect five kinds of QUIC-based services with high accuracy (approximately 99 percent).

Despite the high performance, our proposed method has some disadvantages. The usage of NetFlow-based features leads to the increase of the runtime of processing and classification. We use all packets in the flows, so this will impose some obstacles when the number of packets in flows is huge. In the future, we will construct the larger dataset, deeply investigate the NetFlow-based features, and compare the performance of our proposed method with some traditional traffic that can make our proposed method more reliable.

References

[1] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE Communications Surveys & Tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[2] A. Dainotti, A. Pescape, and K. C. Claffy, "Issues and future directions in traffic classification," *IEEE network*, vol. 26, no. 1, 2012.

[3] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Network traffic classifier with convolutional and recurrent neural networks for internet of things," *IEEE Access*, vol. 5, pp. 18 042–18 050, 2017.

[4] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23-24, pp. 2435–2463, 1999.

[5] W. Lu, M. Tavallaee, G. Rammidi, and A. A. Ghorbani, "Botcop: An online botnet traffic classifier," in *Communication Networks and Services Research Conference, 2009. CNSR'09. Seventh Annual*. IEEE, 2009, pp. 70–77.

[6] T. Karagiannis, A. Broido, N. Brownlee, K. C. Claffy, and M. Faloutsos, "Is p2p dying or just hiding?[p2p traffic measurement]," in *Global Telecommunications Conference, 2004. GLOBECOM'04. IEEE*, vol. 3. IEEE, 2004, pp. 1532–1538.

[7] N. Williams, S. Zander, and G. Armitage, "A preliminary performance comparison of five machine learning algorithms for practical ip traffic flow classification," *ACM SIGCOMM Computer Communication Review*, vol. 36, no. 5, pp. 5–16, 2006.

[8] Y.-s. Lim, H.-c. Kim, J. Jeong, C.-k. Kim, T. T. Kwon, and Y. Choi, "Internet traffic classification demystified: on the sources of the discriminative power," in *Proceedings of the 6th International COnference*. ACM, 2010, p. 9.

[9] C. Wang, T. Xu, and X. Qin, "Network traffic classification with improved random forest," in *Computational Intelligence and Security (CIS), 2015 11th International Conference on*. IEEE, 2015, pp. 78–81.

[10] G. Carlucci, L. De Cicco, and S. Mascolo, "Http over udp: an experimental investigation of quic," in *Proceedings of the 30th Annual ACM Symposium on Applied Computing*. ACM, 2015, pp. 609–614.

[11] R. Hamilton, J. Iyengar, I. Swett, and A. Wilk, "Quic: A udp-based secure and reliable transport for http/2," *IETF, draft-tsvwg-quic-protocol-02*, 2016.

[12] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar *et al.*, "The quic transport protocol: Design and internet-scale deployment," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 2017, pp. 183–196.

[13] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," *arXiv preprint arXiv:1511.08458*, 2015.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[15] L. M. Belue and K. W. Bauer Jr, "Determining input features for multilayer perceptrons," *Neurocomputing*, vol. 7, no. 2, pp. 111–121, 1995.

[16] A. Bashir, C. Huang, B. Nandy, and N. Seddigh, "Classifying p2p activity in netflow records: A case study on bittorrent," in *Communications (ICC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 3018–3023.

[17] M. Lotfollahi, R. Shirali, M. J. Siavoshani, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *arXiv preprint arXiv:1709.02656*, 2017.

[18] V. Tong, "Network flow of quic," *https://drive.google.com/drive/folders/1cwHhzvaQbi-ap8yfrj2vHyPmUTQhaYOj?usp=sharing*, April 2017.

[19] A. Liaw, M. Wiener *et al.*, "Classification and regression by randomforest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[20] Selenium, "Webdriver," *https://www.seleniumhq.org/projects/webdriver/*, April 2017.

[21] lucas clemente, "A quic implementation in pure go," *https://github.com/lucas-clemente/quic-go*, April 2017.

[22] F. CholletKeras, "Keras," *https://github.com/fchollet/keras*, 2016.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *Journal of machine learning research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[24] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information Processing & Management*, vol. 45, no. 4, pp. 427–437, 2009.

[25] F. CholletKeras, "loss function," *https://keras.io/losses/*, 2016.