TLS Encrypted Application Classification Using Machine Learning with Flow Feature Engineering

Onur Barut
University of Massachusetts Lowell
Lowell, MA
Onur_Barut@uml.edu

Rebecca Zhu
Nashua High School South
Nashua, NH
rebeccazhu2003@gmail.com

Yan Luo
University of Massachusetts Lowell
Lowell, MA
Yan_Luo@uml.edu

Tong Zhang
Intel Corporation
Santa Clara, CA
tong2.zhang@intel.com

ABSTRACT

Network traffic classification has become increasingly important as the number of devices connected to the Internet is rapidly growing. Proportionally, the amount of encrypted traffic is also increasing, making payload based classification methods obsolete. Consequently, machine learning approaches have become crucial when user privacy is concerned. For this purpose, we propose an accurate, fast, and privacy preserved encrypted traffic classification approach with engineered flow feature extraction and appropriate feature selection. The proposed scheme achieves a 0.92899 macro-average F1 score and a 0.88313 macro-averaged mAP score for the encrypted traffic classification of Audio, Email, Chat, and Video classes derived from the non-vpn2016 dataset. Further experiments on the mixed non-encrypted and encrypted flow dataset with a data augmentation method called Synthetic Minority Over-Sampling Technique are conducted and the results are discussed for TLS-encrypted and mixed flows.

CCS CONCEPTS

• **Computing methodologies** → **Machine learning**; • **Security and privacy** → **Security protocols**; • **Networks** → **Network privacy and anonymity**.

KEYWORDS

flow feature extraction, feature selection, machine learning, deep learning, encrypted traffic analysis

ACM Reference Format:
Onur Barut, Rebecca Zhu, Yan Luo, and Tong Zhang. 2020. TLS Encrypted Application Classification Using Machine Learning with Flow Feature Engineering . In *2020 the 10th International Conference on Communication and Network Security (ICCNS 2020), November 27–29, 2020, Tokyo, Japan.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3442520.3442529

ICCNS 2020, November 27–29, 2020, Tokyo, Japan

ACM ISBN 978-1-4503-8903-7/20/11…$15.00
https://doi.org/10.1145/3442520.3442529

1 INTRODUCTION

As applications on the Internet proliferate, network traffic classification becomes more important in network security and management such as quality of service (QoS) control, resource allocation, and malicious flow detection. Traffic classification has gained substantial attention for Internet Service Providers (ISP) in order to properly maintain certain networks as well as improve upon the existing networks. Many network flows from major applications such as WhatsApp have the option to be encrypted in order to protect the users' privacy. In addition, malicious users tend to use encryption to hide their actions. Therefore, categorizing encrypted network flows to application types, while protecting the privacy of users, remains an imperative area for research.

Transport Layer Security (TLS) [14] is the cryptographic protocol used to provide communication security in many modern applications. It uses symmetric cryptography to encrypt data. At the handshaking stage of two parties initiating TLS, a shared secret is negotiated and the keys for the symmetric encryption are generated uniquely. The parties settle on the encryption algorithms before their first message is exchanged. The network applications using TLS have secure, authenticated, and reliable connections, which are based on encrypted packet payloads and plain packet headers. While it is computationally feasible to analyze the packet headers in depth, decrypting the payload is computationally challenging and violates the users' privacy. Therefore, classification of flows based on header information, even for TLS encrypted flows, is possible and has recently drawn tremendous interest especially in the machine learning community.

In previous researches, port-based and payload inspections were conducted for network traffic analysis [10]. With the increase in the number of encrypted flows, payload based deep packet inspection (DPI) methods become obsolete, urging the network traffic analysis community to utilize machine learning models that rely on statistical flow features rather than the payload itself.

With machine learning, there are several ways to classify encrypted network traffic including models that use extracted features and deep learning models that use the raw bytes of packets. For feature extraction, a domain expert deals with the raw traffic data and extracts relevant flow features by computing statistics such as the number of bytes, the number of packets, the time duration

of the flow, etc. For models that use feature extraction, flow features are extracted from the headers in the Ethernet frame, the IP datagram, and the TCP/UDP segment. The payload of the packet itself is not included in the feature extraction phase, preserving the user's privacy. Without feature engineering, the classification models utilize all of the features in predicting the network traffic. Utilizing feature selection can sometimes improve the accuracies of the trained models performances if the features are carefully selected [26]. Thus, a feature selection phase is implemented so that the features can be analyzed. Deep learning models, however, do not need feature extraction as they can learn the representations of the data by themselves through their inner neural network layers. Due to their innate feature extraction, deep learning models have become prevalent in classifying images and audio, and recently on traffic data. However, deep learning models have two serious drawbacks that hinder their end-to-end implementation on encrypted traffic analysis. Firstly, they require a large amount of data to train and many prior research works include the packet payloads as input to deep learning models to achieve a higher accuracy, thus potentially violating user privacy [9, 13, 16, 18, 19, 23]. Secondly, deep learning models are computationally onerous in both the training and inference phases.

We are motivated to answer two important open questions on encrypted traffic classification: (1) How effective are the features extracted from encrypted flows in applying machine learning models? (2) Can deep learning models be applied onto flow features, as opposed to the raw payload bytes, to improve the classification accuracy of TLS encrypted flows? Recently, the usage of machine learning (ML) models in malware detection and traffic classification has been explored [9, 17, 19, 23]. These ML models, such as Random Forest (RF) and Multilayer Perceptron (MLP), have been applied to non-TLS flow features. We plan to evaluate these approaches on TLS encrypted flows. In addition, we investigate the performance in network traffic classification through deep learning models such as 1D Convolutional Neural Network (1D CNN). We compare a baseline ML classifier with a CNN model in terms of accuracy and the time to execute the prediction. The data and the source code will be publicly available upon publication.

We make the following contributions in this paper:

- We determine that selecting a smaller number of features but more discriminative features improves the macro-average F1 score of RF models by 5% on the TLS encrypted subset of the dataset.
- We observe that TLS features are useful in classifying encrypted traffic but they degrade the performance on the whole dataset if the proportion of encrypted traffic in the mixed dataset is too small.
- We show that Synthetic Minority Oversampling Technique, a data augmentation method, is helpful to reduce bias and to increase the performance when used with all features available whereas it does not prevent bias but increases the macro-average F1 and mAP scores on the narrower dataset with only 10 features.
- We conduct a preliminary study of small-scale deep learning models trained with TLS features, which serves as an important framework when a significantly larger amount of labeled TLS flows are available.

The rest of the paper is organized as follows: Section 2 provides an overview of related works regarding network traffic classification. Section 3 describes our proposed approach in designing this research, including the experiment setup and data preprocessing. In Section 4, we evaluate and discuss our results. Finally, we conclude our paper in Section 5 and discuss future work.

2 RELATED WORK

2.1 Transport Layer Security

TLS is a cryptography protocol that encrypts data sent over the Internet [14]. It supersedes the Secure Socket Layers (SSL) for network security. While it is used for secure email sending and secure file uploading, TLS is most prominently utilized for secure web browsing. In the TCP/IP network stack, TLS is a layer that fits between the transport and application level, making it easy to add on top of TCP services which results in TLS's increasing popularity. TLS uses a handshake protocol between the web server and web client to establish a connection and negotiate a secret key used to encrypt and decrypt the information. Through symmetric encryption, where the same key is used to encrypt and decrypt, and public key encryption, TLS protects the users' privacy. Thus, if there is an eavesdropper or hacker, the secret key is unknown to the hacker, and the information is protected. Some of the TLS features used in research works include the number of TLS packets, the number of TLS extensions, and the number of ciphersuites.

2.2 Encrypted Traffic Classification

Previous works in classifying encrypted network traffic involve using machine learning models to predict network traffic. Deep learning models including Convolutional Neural Network (CNN) and Long Short Term Memory (LSTM) have shown their effectiveness in many applications including computer vision [1], human activity recognition [2] and Internet traffic classification [9, 15, 19, 22, 25]. Lotfollahi et al. [9], for example, use a CNN and stacked autoencoder neural network in order to identify applications and classify traffic using network flow data. In training the model, they keep the IP header as well as the first 1480 bytes of each IP packet to perform packet classification. Through their work, they conclude that the CNN results in 0.93 precision and a F1 score of 0.95 when using the "ISCX VPN-nonVPN" dataset. Similarly, Yang et al. use an autoencoder to extract features in TLS/SSL encrypted traffic data and a CNN that achieves state-of-the-art results [22]. In another research, Zhang et al. [25] combine LSTM and CNN to create a new neural network called Stereo Transform Neural Network (STNN). The model achieves an average F1 measure of 0.95 and an average accuracy of 99.5% on TLS/SSL encrypted traffic classification. Vu et al. [15], on the other hand, use LSTM, a type of recurrent neural network, for a time series analysis of traffic data. Based on packet payload features, their LSTM model achieves a 98% accuracy on ISCX VPN-nonVPN dataset. Wang et al. [19] use an end-to-end framework to classify traffic data. In other words, they use raw traffic as input and final labels as output as opposed to hand-designed features for input. They train 1D CNN on the ISCX VPN-nonVPN dataset. However, they do not address the imbalance in the training data when training the 1D CNN.

Other researchers propose training a artificial neural network (ANN) to classify typical P2P protocol such as Kazaa and BitTorrent.

Ting et al. [8] find that utilizing ANN achieve greater accuracy. Gao et al. [6] apply deep belief networks (DBN) on the KDD Cup 1999 dataset with a specific focus on intrusion detection. Utilizing feature selection, Gao et al. conclude that their model outperforms a Support Vector Machine (SVM).

Another area garnering increased attention in encrypted traffic identification is mobile app traffic data. Using their own dataset on Android traffic, Wang et al. [20] employ several deep learning models including a Stacked Denoising Autoencoder (SDAE), 1D CNN, and a bidirectional LSTM to identify encrypted traffic. They conclude that the 1D CNN classifier performs the best with a 91.8% accuracy and a macro-average F-measure of 90.1%. All their models, however, achieve high accuracy regardless of TLS encryption.

While these research works focus on using deep learning models to classify traffic data, other researchers such as Wang et al. [17] employ traditional machine learning models. They train a Random Forest (RF) classifier to identify smartphone apps from encrypted data. In another study, Taylor et al. [12] improve upon Wang's framework by adding 110 applications to classify, eventually achieving a 73% accuracy in the multi-class classification. To improve the RF accuracy on certain datasets, Yamansavascilar et al. [21] use feature selection, citing an improvement of 2% in their RF's accuracy. For feature selection, Yamansavascilar manually selected 111 flow features to use from the dataset. In some research works, including Gil et al. [4], they analyze specific features in traffic flow such as duration of the flow, forward and backward inter-arrival time, etc. to train a k-nearest neighbor (k-NN) and C4.5 decision tree algorithm. The C4.5 algorithm they employ achieves 88% recall on their dataset. While other studies focus on packet-based and payload-based features for classification, Gil et al. utilize time-based attributes for traffic classification. Similarly, Ding and Li [11] propose a hybrid method to classify traffic data in real time using C4.5 decision tree algorithms and then classify the services using a RF classifier. Through their hybrid model, Ding et al. achieve a 95% accuracy. Another traditional machine learning model utilized in classifying encrypted traffic is Naive Bayes classifier, which is employed by Zhang et al. [24].

3 PROPOSED APPROACH

We propose a feature selection mechanism before training the model to improve the performance of encrypted application classification using the non-vpn2016 dataset. The overall proposed flowchart is provided in Figure 1. Firstly, Metadata and TLS features are extracted using the flow feature extraction tool. Then, TLS-encrypted flows are extracted from the whole dataset. After, we filter out the flows without any packets towards the client and end up with a pure bidirectional flow data, as the number of packets received by the client should play an important role in application classification, assuming that this classification model is deployed in the client end of the network. Then, we read Metadata and TLS features separately and combine them to create the data matrix. After that, label information is added according to the filename of network capture and 20% of each class is reserved as the validation set while the remaining samples are used to train the classifier. Finally, we use the validation set to obtain results in macro-average F1 and mAP scores.

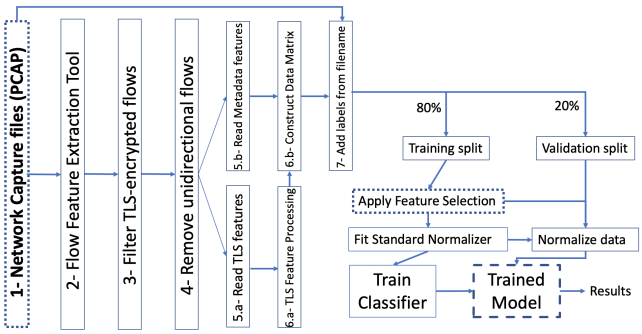

Figure 1: Proposed approach for TLS-encrypted flow classification

Deep learning models are widely used in network traffic classification. Most of the previous researches deploy CNN and LSTM models with the raw traffic capture data as input and exploit the models to learn their own representations internally. In the light of the previously reported state-of-the-art classification performance on network traffic using deep models on top of the raw data, we are inspired to implement similar deep models on top of the extracted features to see if the model can produce more accurate results. Therefore, we compare the effect of feature selection methods by implementing deep learning models, 1D CNN and 2D CNN, on top of the extracted features. Moreover, we implement the Synthetic Minority Over-Sampling Technique (SMOTE) to overcome the bias due to the imbalance of the dataset.

The non-vpn2016 dataset is a mixture of both TLS-encrypted and non-encrypted flows. Along with the TLS-encrypted traffic classification, we also attempt to increase the classification performance of all the flows. For that purpose, the proposed approach in Figure 1 can still be applied by only removing the TLS-encrypted flow extraction step.

3.1 Feature Selection Methods

We evaluate four different feature selection methods to select the most powerful features for an efficient classification. Firstly, the Random Forest algorithm provides feature importance which can be used to evaluate the importance of the features. Secondly, the correlation between the feature and the label is used as another feature selection method. Thirdly, Principal Component Analysis (PCA) is implemented to select features with high relevance. Finally, a novel method using Jensen-Shannon Divergence values is proposed for feature selection.

3.1.1 Random Forest as feature selector. The Random Forest classifier has a feature importance attribute that weighs the features according to the contribution to the classification. Therefore, we use the feature importances assigned by the Random Forest as a feature selection method.

3.1.2 Feature selection using correlation. The correlation of two distributions gives an indication of how much one of the distributions is affected by a change in the other distribution. In other words, if a change in the value of a feature triggers a similar change

in the value that represents the label of the class, then we may assume the feature possesses discriminative information.

3.1.3 Feature selection with PCA. We apply a feature selection method using the eigenvalues of the covariance matrix obtained from the training set. We get the contribution from each eigenvector to the principal component and sort the features accordingly.

3.1.4 Feature selection with Jensen-Shannon Divergence. Jensen-Shannon Divergence computes how similar or different the given two distributions are. The computed value is a continuous number in the interval of $[0, 1]$ where 0 means the two distributions are identical and 1 means the two distributions are completely different. Using this score, we compute the distribution of each feature in one class to other classes one by one, and get the average of them to obtain a value between 0 and 1.

3.2 Data and Preprocessing

3.2.1 Dataset description. We get the raw traffic data from the publicly open CIC repository. The ISCX VPN-nonVPN2016 dataset [4] has different classes including "facebook_audio", "facebook_chat", "skype_audio", "skype_chat" etc. with and without VPN. Similarly, the ISCX Tor-nonTor2017 dataset [7] includes similar classes with and without Tor traffic. First, network flow features including Metadata, TLS, DNS and HTTP header features are extracted using our feature extraction tool. Then, a binary Tor-nonTor classification dataset is generated according to Scenario A in [7] and a seven-class network traffic classification dataset is obtained with "P2P", "Audio", "Chat", "Email", "File_Transfer", "Tor", and "Video" labels by combining relevant sub-classes into the broader groups using the non-vpn2016 dataset.

Non-vpn2016 and Tor-nonTor2017 datasets contain 163831 and 51574 flow samples, respectively, with many different Metadata features along with protocol-defined features such as TLS, DNS and HTTP if used in the flow. In this study, we include only TLS defined protocol features along with Metadata features for the encrypted traffic analysis experiments. A small percentage, in other words, only 1578 flows are TLS encrypted in non-vpn2016 dataset. Since Tor-nonTor2017 Scenario A is a binary classification dataset, we use it to expand our results in feature selection and evaluation experiments only.

3.2.2 TLS feature processing. Machine learning models require a data matrix as input. This input data matrix is usually a two dimensional array whose rows correspond to flow samples and columns correspond to the features. Metadata features are numerical features that can be extracted for any type of flow and therefore can be easily loaded into a data matrix. Some examples of metadata features include, but are not limited to, the source port number, the destination port number, the number of packets inbound, the number of packets outbound, the number of bytes inbound, the number of bytes outbound, the time duration of the flow, etc. On the other hand, TLS features contain both numerical features such as the number of ciphersuites offered by the client and server, the number of TLS encrypted packets in the flow, etc. as well as variable-size non-numerical feature arrays such as the ciphersuites offered by the client, the supported TLS extensions of the client, etc. Hence, it is important to find a way to convert the variable-size non-numerical features into fixed length numerical features to be able to load the TLS information into the data matrix.

Firstly, we import TLS features along with metadata features. Tls_cnt, tls_cs_cnt, tls_ext_cnt, tl_key_exchange_len, tls_svr_cnt, tls_svr_cs_cnt, tls_svr_ext_cnt and tls_svr_key_exchange_len features contain single integer values; therefore, it is straightforward to load those features into data matrix. Each of these features occupies a single column and the corresponding value is loaded to the corresponding column.

Tls_cs, and tls_ext_types are features with variable sizes. For example, the tls_cs feature for a single flow sample may contain 'c00a', '00af' and other ciphersuites as an array of strings. Tls_ext_types is also similar to tls_cs. Therefore, we first analyze those ciphersuites and extension types and get top-N most common features, say N=10, for each class in the dataset. Then, we combine those top-10 common ciphersuites for each class. It usually makes a number larger than 10, for example 16, because different classes may contain different top common ciphersuites. We then place each of these 16 ciphersuites as a separate column in data matrix, and if the flow contains any of those ciphersuites in tls_cs feature, we put 1 for the corresponding column. If not, we put 0. At the end of the line, we add another column as the 17^{th} column that represents the ciphersuites that are not among the top-N. For example, a flow sample containing 11 other ciphersuites that are not among the top commons has 11 for the 17^{th} column of tls_cs feature in the data matrix. The same approach applies for the tls_ext_types feature.

Tls_svr_cs feature is a single-valued feature with a ciphersuite selected among the advertised ciphersuites in tls_cs. Therefore, we have a binary feature for this. If the selected ciphersuite is among the common-N ciphersuites in tls_svr_cs, then we put 1, otherwise we put 0 for the corresponding column in the data matrix.

Tls_svr_ext_types is almost the same as the tls_cs and tls_ext_types. So, the same procedure also applies for this feature as well.

Tls_len and tls_svr_len are variable length integer arrays with payload sizes. For example, tls_len: [541, 45, 234, 21] and tls_svr_len: [67, 54, 256]. For these two features, we allocate 4 columns for each to represent this information in the data matrix: (1) the length of the array, (2) the minimum value in array, (3) the maximum value in array, and (4) the mean value of array.

If a flow sample does not contain any TLS features, or in other words, the flow is not TLS encrypted, we then insert all zeros for the corresponding TLS columns in the data matrix.

Secondly, we observe that there are many flows whose number of inbound packets feature is zero. In other words, the flow is not bidirectional. In a scenario where the proposed application classification model is deployed in the client end of the network, then it is reasonable to assume that the flows in the dataset that have no incoming packet to the client can be considered as unacknowledged queries. Therefore, we filter out those flows and come up with 13635 pure bidirectional flows in the dataset.

3.2.3 Data augmentation using SMOTE. Many of the datasets are naturally imbalanced due to the data collection setup [5]. SMOTE [3] is a method to increase the numbers of undersampled instances in the dataset by computing the synthetically generated features according to the k-nearest neighbors of the instance in the feature

space. We utilize the imbalance-learn library for python to apply SMOTE in the non-vpn2016 dataset. By the default parameters, the SMOTE method generates synthetic samples for all classes except for the one with the majority in the dataset, making the number of samples for each class equal to that of the majority. In our experiments, we observe a significant imbalance in the non-vpn2016 dataset and augment the minority classes with SMOTE to mitigate the bias in the prediction.

3.3 TLS-Encrypted Flow Classification

The non-vpn2016 dataset contains both TLS-encrypted and non-encrypted flows. We define a flow as TLS-encrypted if the flow contains TLS features in the input data. We observe that after pre-processing, there are 1360 TLS-encrypted flows, i.e. about 1% of the entire dataset. We create a TLS-encrypted subset to perform feature selection and compare the results with deep learning algorithms on the TLS-encrypted dataset.

4 EVALUATION AND DISCUSSION

Accuracy in the validation set is the most popular metric for a classification evaluation. However, accuracy by itself does not provide a detailed insight about the performance of a model for a multi-class problem, especially if the dataset is imbalanced. Therefore, we utilize the recall, precision, macro-average F1 and macro-average mAP scores whose formulae are given in equation (1) and (2)

$$Recall = \frac{TP}{(TP + FN)}, Precision = \frac{TP}{(TP + FP)} \quad (1)$$

$$F1 = \frac{2 * precision * recall}{(precision + recall)}, mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \quad (2)$$

where N is the number of samples, TP is true positive, FN is false negative, FP is false positive and AP_i is the average precision for the i^{th} class. In our experiments, we use macro-averaged F1 and mAP scores because macro-averaged scores are not affected by the imbalance in the dataset. Therefore, macro-averaged scores gives an unbiased result about the model performance.

In our experiments, the Scikit-learn open-source Python library is utilized to calculate the metrics aforementioned. Both encrypted and non-encrypted flows, also referred to as the whole flows of the non-vpn2016 dataset, are used to evaluate the feature selection methods. All of the experiments are conducted on a machine with Intel® Core™ i7-6700HQ CPU at 2.60GHz processor, 16 GB RAM and a GPU GeForce GTX 1060 with 6 GB memory.

4.1 Feature Evaluation and Selection

Four different feature evaluation methods are implemented for the feature selection phase. Random Forest and k-nearest neighbor classifiers are used to evaluate the performance, as they provide the most accurate results for the non-vpn2016 dataset. Since the number of TLS-encrypted flows is very small, we use the whole dataset to evaluate different feature selection methods. Figure 2 shows the macro-average F1-score of each method on the validation set according to the different number of features using RF and kNN classifiers. We observe that both PCA-based and correlation-based feature evaluation methods degrade the classification performance

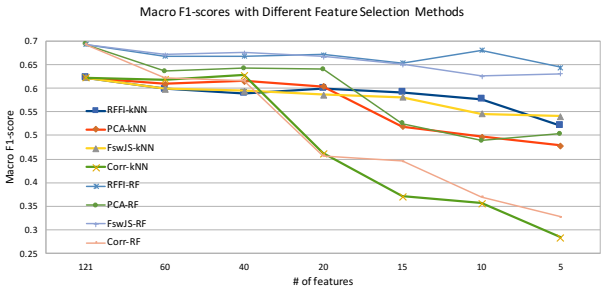

Figure 2: Comparison of different feature selection methods on the non-vpn2016 dataset

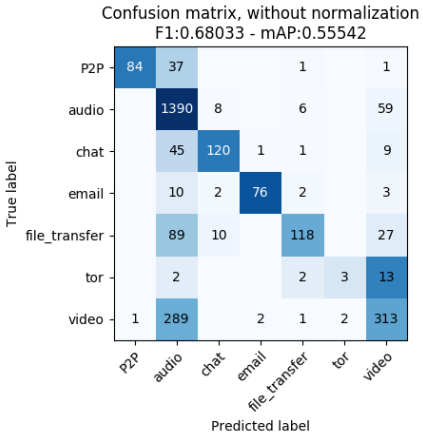

Figure 3: Confusion matrix of the RF model trained with top-10 features on the whole non-vpn2016 dataset

drastically when a smaller number of features are selected to train the model. On the other hand, the proposed feature selection methods using Jensen-Shannon (FSwJS) Divergence and Random Forest feature importance (RFFI) with the RF classifier produce more accurate results than the other methods. However, we observe that both FSwJS and RFFI cannot exceed the macro-average F1-score that is achieved with the all Metadata features used.

The most accurate model after feature selection is obtained using the RF classifier with the top-10 features selected by the RFFI. Although RF model with all 121 features achieves 0.6925 macro-average F1-score, RF model with top-10 features of RFFI reaches to 0.6803. The list of important top-10 features selected by RFFI is given in Figure 4. The two most important features for the non-vpn2016 application classification dataset are the port number of the source and the mean of the payload sizes in the reverse direction of the flow. Similarly, the port number of the destination and the duration of the flow, time_length, are evaluated as the third and fourth important features, respectively. Other important features selected in the top-10 are related to the payload sizes in both direction and the number of bytes sent and received.

The confusion matrix of the Random Forest model trained with the top-10 features on the whole non-vpn2016 dataset is given in Figure 3. The most obvious observation is that the number of flows

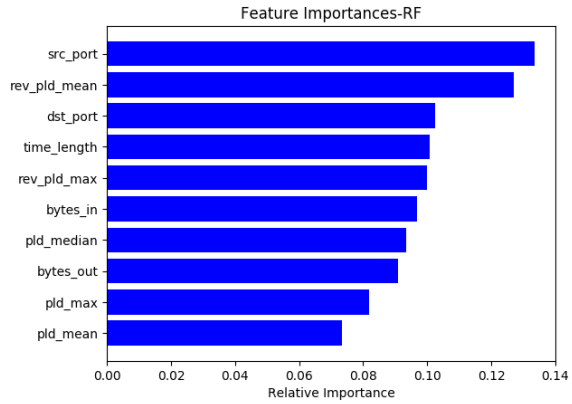

Figure 4: Top-10 important features selected by RF model on the whole non-vpn2016 dataset

belonging to the Audio class is much higher than the other classes. The same ratio of sample numbers between classes also applies to the training set. As a result of this imbalance, the model becomes biased and favors the Audio class more than others. Except for the Tor class, all other classes have significant misprediction to the Audio class because of this imbalance. The reason why the Tor class is mispredicted as Video is because the majority of the samples in the Tor class are actually samples for video flows, namely tor_youtube and tor_vimeo obtained in the Tor browser.

The ISCX Tor-nonTor2017 [7] dataset from the CIC repository is also utilized to evaluate the proposed feature selection methods following the same approach given in Figure 1. In the first scenario, the Tor-nonTor binary classification is performed. In the second scenario, 8 different traffic types of nonTor (Audio, Browsing, Chat, Email, File-transfer, P2P, Video, VoIP) dataset are classified. Since RFFI and FSwJS based feature selection methods perform better in non-vpn2016 dataset, only these two feature selection methods with kNN and RF classifiers are used. Table 1 shows the feature selection results with kNN and RF classifiers for all Metadata, Top-10, and Top-5 selected features, respectively. Top-10 feature selection using RFFI and RF classifier yields a perfect classification result on the validation set with both recall and precision for Tor and nonTor classes equal to 1.0. These results achieve higher accuracy when compared to all 121 Metadata used to train the model. Moreover, when compared to C4.5 proposed in [7], the top-5 features selected in our proposed method using RF model achieves higher precision for both Tor and nonTor classes as well as recall for nonTor class while recall for Tor class is lower by only around 2%. Similarly, the top-5 features selected by RFFI using the kNN classifier achieves higher recall and precision for both Tor and nonTor classes except recall for the Tor class where the difference is less than 1%.

As the second scenario, nonTor traffic is classified into 8 labels with the top-10 features selected by RFFI as it yields the most useful features in our previous experiments included in this paper. Recall (Rc), and Precision (Pr) for the given number of features (#) are provided for the RF and kNN classifiers for all classes in Figure 5. Each classifier has its own advantages. For example, the kNN method with all Metadata features achieved higher recall and precision values for Chat, Email and Video classes than the RF classifier. For

Table 1: Recall and Precision values for different methods on tor-nonTor classification

#Features	Metric	RFFI-kNN	FSwJS-kNN	RFFI-RF	FSwJS-RF
121	Rc-Tor	0.90909	0.90909	0.93939	0.93939
	Rc-NonTor	1.0	1.0	0.99981	0.99981
	Pr-Tor	1.0	1.0	0.93939	0.93939
	Pr-NonTor	0.99971	0.99971	0.99981	0.99981
10	Rc-Tor	0.96967	0.87879	1.0	0.96968
	Rc-NonTor	1.0	0.99932	1.0	1.0
	Pr-Tor	1.0	0.80556	1.0	1.0
	Pr-NonTor	0.99990	0.99961	1.0	0.99990
5	Rc-Tor	0.87879	0.51515	0.90909	0.81818
	Rc-NonTor	0.99990	0.99932	0.99990	1.0
	Pr-Tor	0.96667	0.70833	0.96774	1.0
	Pr-NonTor	0.99961	0.99845	0.99971	0.99942
#Features	**Metric**	**[7] C4.5**	**[7] kNN**	**[7] ZeroR**	
5	Rc-Tor	0.93	0.88	0.0	
	Rc-NonTor	0.99	0.98	1.0	
	Pr-Tor	0.95	0.85	0.0	
	Pr-NonTor	0.99	0.98	0.89	

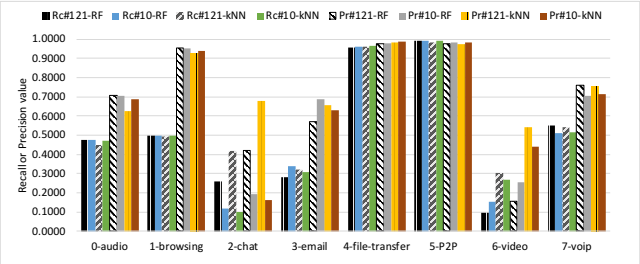

Figure 5: Recall (Rc) and Precision (Pr) on the non-tor2017 dataset

the RFFI selected top-10 features, similar but less accurate results are obtained, as expected. Again, the kNN method classifies File-transfer, Video and VoIP classes with higher recall and precision than the RF model. Audio and P2P recall and precision values with the kNN are increased up to 3% with the top-10 features while Chat recall and precision values dramatically reduce. For the RF model, similar to the kNN, Chat recall and precision values drop from 0.26 to 0.12 and 0.42 to 0.19, respectively, while Email and Video recall and precision values improve up to 11%. Evaluation metrics for other classes remain almost the same with minor reductions with the top-10 features. Overall, Chat, Email and Video class performances for the classifiers are the worst while File-transfer and P2P class performances achieve more than 95%.

4.2 Contribution of TLS Features

The non-vpn2016 dataset consists of four different types of features as described in section 3.2.1. TLS features play an important role when it comes to TLS-encrypted network flow classification. However, in the non-vpn2016 dataset, the encrypted flows only cover around 1% of the whole data. Therefore, training a model with TLS features may not contribute to the overall accuracy. In order to test this, we compare the Random Forest classifier results with and without TLS features. Figure 6 displays the macro-average F1-scores of RF classifiers trained with and without TLS features and with different numbers of features. Top important features are

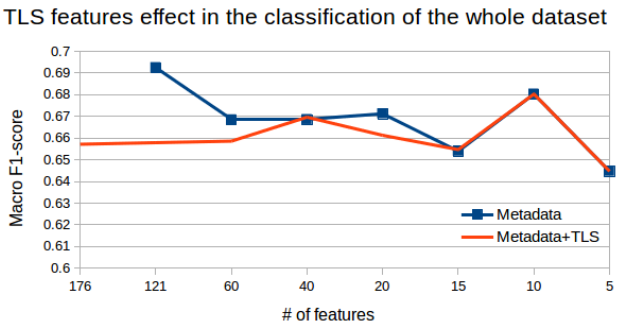

Figure 6: Contribution of TLS features to the classification of the whole non-vpn2016 dataset

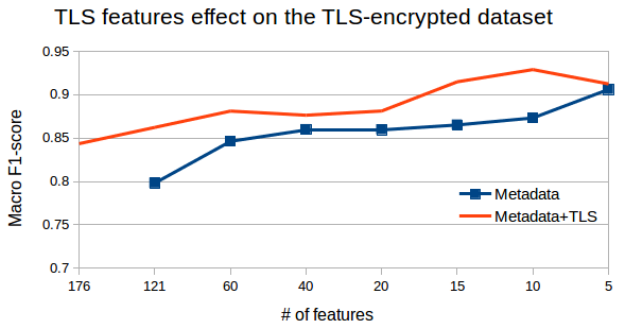

Figure 7: Contribution of TLS features to the classification of the TLS-encrypted subset of the non-vpn2016 dataset

selected by the RFFI method. In Figure 6, we observe that taking into account TLS features does not provide additional information to the model for a better classification. Interestingly, the macro-average F1-scores obtained with the Top-10 and the Top-5 features are exactly the same with the two sets of input features because the feature selection method interprets that TLS features do not play an important roles for the current classification and does not select them as top important features. The main reason behind it is that the number of encrypted flows in the whole dataset is too small such that the values of the TLS features for the majority of the samples are zero. This causes the information that can be extracted from TLS features to become noisy for the dataset.

4.3 TLS-Encrypted Traffic Classification

While the TLS features do not play an important role in classification with regards to the whole dataset comprised of non-encrypted and encrypted data, when only TLS-encrypted data is used, TLS features play an imperative role in classification. The TLS-encrypted subset of the non-vpn2016 dataset is used to analyze Metadata features versus Metadata and TLS features to evaluate the TLS features in a fair manner. To test if the TLS features affect the accuracy of the Random Forest models on the TLS-encrypted subset, we compare the results of the Random Forest classifier with and without TLS features in addition to the RFFI feature selection method.

We first notice that there is not a single P2P flow that is TLS-encrypted. Additionally, there are only two TLS-encrypted flows for the Tor class. Since SMOTE requires, in our experiments, at least three other samples in order to find the nearest neighbors for data augmentation, we remove the Tor class samples in the TLS-encrypted subset and obtain a dataset with four classes. While SMOTE does not affect the model in this section, we need an unbiased comparison between the current model and the model with SMOTE augmented data, so we must filter out the Tor class.

Figure 7 details the macro-average F1-scores of both RF classifiers. We observe that in both cases, selecting a smaller number of features but more discriminative features generally increases the macro-average F1-scores. In other words, when the uninformative features are eliminated during feature selection, the results obtained from the RF model are more accurate. While the RF classifier without TLS features achieves its highest accuracy with the Top-5 features, the

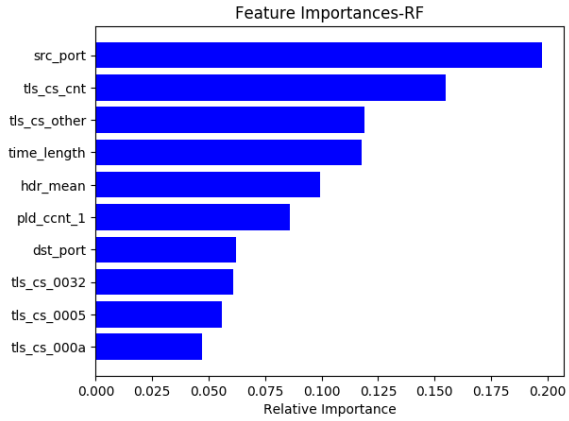

Figure 8: Top-10 important features selected by RF model on the TLS-encrypted subset of the non-vpn2016 dataset

RF classifier with TLS features achieves its highest accuracy with the Top-10 features. Using only the Top-5 Metadata features, the RF model achieves a 0.90615 macro-average F1-score as opposed to the 0.92899 macro-average F1-score from the Top-10 Metadata and TLS features from the RF model. However, with any number of features, the RF classifier with TLS features outperforms the RF classifier without TLS features. This means that there are TLS features selected by the RFFI feature selection in each of the trials so that the RF classifier's accuracy increases. Figure **??** shows the top-10 features selected by RFFI. We see that while there are some common features such as source port, destination port and time duration of the flow, there are also TLS features in the Top-10. Top selected TLS features are mostly from the ciphersuites advertised by the client. The most important TLS feature which comes after the source port feature is the number of ciphersuites advertised by the client. The other important TLS features are some of the most common client advertised ciphersuites such as 0032, 0005 and 000a. Thus, TLS features play an important role in classification when classifying the TLS-encrypted flows.

The confusion matrix obtained with the Top-10 features in the TLS-encrypted subset is given in Figure 9. Overall, the random forest classifier achieves a successful classification. The only apparent

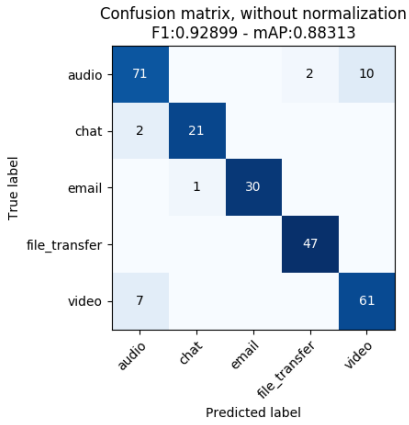

Figure 9: Confusion matrix of Random Forest classifier on the TLS-encrypted subset of the non-vpn2016 dataset

misclassification occurs between Audio and Video classes. More than 10% of the Audio class in the validation set are misclassified as Video and around 10% of the Video class are mispredicted as Audio. This misclassification may be explained because any video also contains an audio track.

4.4 Deep Learning with Flow Features

Deep learning methods are proven to perform very successfully with a large amount of training data, referred to as big data. The size of the data is considered big if the data has many features or the dataset contains a lot of samples, or both. After observing that adding TLS features does not contribute to the classification performance for the non-vpn2016 dataset, one might think of implementing more complex models, such as deep neural networks, which integrate the classifier and feature selection in one process.

In this section, we evaluate the effect of deep learning methods using convolutional neural networks as they are proven to be successful in different related studies. We compare the classification performance in terms of macro-average F1-score, macro-average mAP score, and classification speed in terms of flow per second for the whole dataset and TLS-encrypted subset, respectively. We implement 1D CNN and 2D CNN networks that have similar architecture to the famous LeNet-5 that contains two convolutional layers followed by pooling layers, two fully connected layers and the final output layer. Hyperparameters for these networks are selected with grid search and the final set of hyperparameters that gives the best results on the validation set is given in Table 2.

Two different CNN models are implemented on top of the extracted flow features and the results are compared with the Random Forest classifier. The same validation data is used to test all the models to measure the classification speed in terms of flow per second. We run the inference 100 times and get the average to find out how many seconds one inference takes. The number of flow per second is then obtained by dividing the number of flows in the validation set by the average execution time of the classifier. The results, along with the macro-average F1 and mAP scores are given in Table 3. The best macro-average F1 score of 0.6925 is obtained using all of the Metadata features with the RF classifier. On the other hand, the best macro-average mAP score of 0.55542 is achieved by the RF model trained using only the Top-10 important features selected using RFFI. However, the RF model with Top-10 features is slightly slower than the RF model trained on the whole 121 Metadata features. This means that using only 10 features for Random Forest forces the trees in the forest to be deeper which causes the classifier to perform slower.

Table 2: Hyperparameters set for the CNN models

	1D CNN	2D CNN
Learning Rate	0.001	0.001
Decay Rate	1e-5	1e-5
Dropout Rate	0.5	0.5
Reg. constant	1e-5	1e-5
Batch Size	100	100
Epochs	1000	1000
Filters	128	128
Kernel Size	4x1	4x4
Strides	1	1
CNN Layers	2	2
# of params	27479	23767

Although CNN models are faster in terms of flow classification per second, it should be kept in mind that the CNN models are trained on GPU and tested on CPU while the RF model is trained and tested on CPU. Despite being faster on GPU, the trained 1D and 2D CNN models cannot achieve as accurate results as the Random Forest models. Additionally, the RF models outperform the kNN classifiers when using all of the metadata and when using the Top-10 features. However, the flow rate achieved by the kNN classifier using the Top-10 features is the highest among the other models.

Table 3: Performances on the whole flows in the non-vpn2016 dataset

	Macro F1	Macro mAP	Flow per Sec.
All Metadata (RF)	0.69250	0.55076	15661
Top-10 of Metadata (RF)	0.68033	0.55542	14990
All metadata (kNN)	0.62223	0.44749	567
Top-10 of Metadata (kNN)	0.57721	0.40574	23494
Metadata+TLS (1D CNN)	0.30600	0.25370	18265
Metadata+TLS (2D CNN)	0.23112	0.21030	20703

For the 1D and 2D CNN models on the TLS-encrypted dataset, we must keep in mind that there are only approximately 1% of the flows which are encrypted, resulting in a low number of flows in our TLS-encrypted dataset. Therefore, a deep model may perform well during the training phase, but not during the testing phase due to overfitting. To prevent this, sufficiently small sized CNN models are implemented. The results of the 1D CNN and 2D CNN models are shown in Table 4. The macro-average F1 and macro-average mAP scores for both the 1D CNN and 2D CNN are lower than those of the RF and kNN classifiers. In particular, the macro-average F1 and macro-average mAP scores of the 1D CNN are 0.63256 and 0.57247 respectively. On the other hand, the macro-average F1 and macro-average mAP scores of the 2D CNN are 0.76728 and 0.67226 respectively. Thus, the 2D CNN outperforms the 1D CNN; however, both models achieve less accurate results than the RF and kNN classifiers. Compared to the 1D CNN and 2D CNN models trained on the whole non-vpn2016 dataset, the 1D CNN and 2D

CNN models trained on the TLS-encrypted subset have significantly improved results, with the macro-average mAP score of the 2D CNN improving from 0.2103 to 0.67226. This means that TLS features in addition to the Metadata features are providing discriminative information to the deep learning models for a better TLS-encrypted flow identification.

The deep learning models achieve slightly lower macro-average F1 and macro-average mAP scores than the RF classifier trained using all Metadata and TLS features. In fact, the RF classifier with the top-10 Metadata and TLS features outperforms all three other models. Additionally, kNN with top-10 features is faster in terms of classifying flows per second; however, its F1 and mAP scores are far lower than best performing RF classifier.

Table 4: Performances on the TLS-encrypted flows in the non-vpn2016 dataset

	Macro F1	Macro mAP	Flow per Sec.
All Metadata+TLS (RF)	0.84364	0.75403	17197
Top-10 of Metadata+TLS (RF)	0.92899	0.88313	20658
All Metadata+TLS (kNN)	0.77466	0.66722	2752.82
Top-10 of Metadata+TLS (kNN)	0.89298	0.82733	25455
Metadata+TLS (1D CNN)	0.63256	0.57247	18673
Metadata+TLS (2D CNN)	0.76728	0.67226	21139

4.5 Bias Reduction with SMOTE

We observe a bias in the prediction of classes due to the imbalance in the samples of the non-vpn2016 dataset. Therefore, we apply SMOTE to increase the number of flows of the undersampled classes to reduce this bias in prediction. Table 5 shows the macro-average F1 and mAP scores of the different models and different number of features with and without SMOTE. Despite the decrease in the F1 and mAP scores for the RF classifier trained with the whole Metadata features, the other five approaches with SMOTE yield a boost in the classification accuracy. We observe a slight increase of F1 and mAP scores with the RF classifier using the top-10 features and the kNN classifiers; however, the increase of the scores for the two CNN models are almost 13% and 20% for F1 score and around 6% and 9% for mAP score in the 1D CNN and 2D CNN respectively. Even with SMOTE, the CNN models and the kNN models are still less accurate than the RF classifier with the top-10 features.

In order to understand the reason behind the decrease in the F1 and mAP scores for the RF model with the all Metadata features, the confusion matrix in Figure 10 is examined. We observe that SMOTE has reduced the bias to the Audio class and helps the model produce more accurate results for the other classes. For example, the Tor class is now accurately classified with 70% recall rate. However, the Audio class recall rate heavily drops from around 95% to 55%.

Figure 11 shows the confusion matrix obtained using the RF classifier trained with the top-10 features on the SMOTE augmented non-vpn2016 dataset. Even though both of the F1 and mAP scores are slightly increased, there is still a bias towards to the majority class in the predictions. Here we can conclude that SMOTE with small number of features is not effective to prevent bias but has a potential for a boost in the performance of the classifier with the imbalanced dataset.

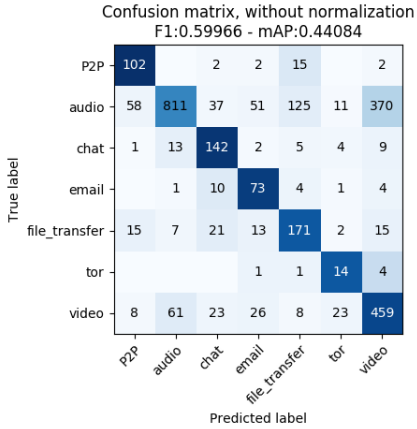

Figure 10: Confusion matrix obtained with Random Forest classifier with all Metadata features on the SMOTE augmented non-vpn2016 dataset

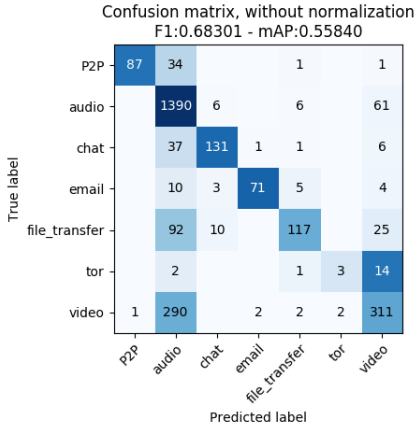

Figure 11: Confusion matrix obtained with Random Forest classifier with top-10 features on the SMOTE augmented non-vpn2016 dataset

Table 5: Effect of SMOTE to the classification of the whole flows in the non-vpn2016 dataset

	Macro F1		Macro mAP	
SMOTE:	**No**	**Yes**	**No**	**Yes**
All Metadata (RF)	0.69250	0.59966	0.55076	0.44084
Top-10 of Metadata (RF)	0.68033	0.68301	0.55542	0.55840
All Metadata (kNN)	0.62223	0.63505	0.44749	0.46335
Top-10 of Metadata (kNN)	0.57721	0.58218	0.40574	0.41334
Metadata+TLS (1D CNN)	0.30600	0.43952	0.2537	0.31267
Metadata+TLS (2D CNN)	0.23112	0.43084	0.2103	0.30392

We see similar results when SMOTE is used on the TLS-encrypted subset of the non-vpn2016 dataset. These results are detailed in Table 6. Firstly, the macro-average F1 and macro-average mAP scores for the RF classifiers and kNN with top-10 features suffer a decrease in value when SMOTE is applied. To quantify, the macro-average

F1 and macro-average mAP scores were 0.92899 and 0.88313 respectively before applying SMOTE and decreased to 0.90855 and 0.85368 respectively after applying SMOTE when using the RF classifier with the top-10 Metadata and TLS features. On the other hand, when SMOTE is applied to the deep learning models and kNN with all Metadata and TLS features, the macro-average F1 and macro-average mAP scores increase. While the 2D CNN's result and both kNN classifiers' results slightly increase, the 1D CNN's performance increases more: the macro-average F1 score increases from 0.63256 to 0.81779 and the macro-average mAP score increases from 0.57247 and 0.72035. Thus, SMOTE can improve the accuracy of a deep learning classifier even with small number of samples in a dataset but it may not be sufficient to achieve the highest accuracy.

Table 6: Effect of SMOTE to the classification of the TLS-encrypted flows in the non-vpn2016 dataset

	Macro F1		Macro mAP	
SMOTE:	**No**	**Yes**	**No**	**Yes**
All Metadata+TLS (RF)	0.84364	0.83181	0.75403	0.73905
Top-10 of Metadata+TLS (RF)	0.92899	0.90855	0.88313	0.85368
All Metadata+TLS (kNN)	0.77466	0.79855	0.66722	0.69072
Top-10 of Metadata+TLS (kNN)	0.89298	0.83331	0.82733	0.74065
Metadata+TLS (1D CNN)	0.63256	0.81779	0.57247	0.72035
Metadata+TLS (2D CNN)	0.76728	0.82666	0.67226	0.73581

5 CONCLUSION AND FUTURE WORK

In recent years, the demand for accurate, fast, and privacy preserving application classification solutions has increased due to the extended use of encrypted packet traffic on the Internet. However, most of the prior approaches fail to cover all three criteria for classification. By analyzing and selecting flow features from TLS-encrypted flows, we provide an accurate and relatively fast approach while preserving user privacy. The experimental results show that client advertised ciphersuites play significant roles for an accurate encrypted traffic classification. However, the use of TLS features for a mixed dataset is not effective when the number of encrypted flows is too small. In addition, this paper shows that engineered feature extraction and proper feature selection for encrypted traffic classification is still important and that accurate and sufficiently fast classifiers can be achieved by the Random Forest algorithm. Because the CNN models are used on the small sized dataset, we plan to examine a larger dataset using selected flow features and deep learning algorithms in the future.

ACKNOWLEDGMENTS

This work was supported in part by Intel Corporation.

REFERENCES

[1] Onur Barut and A. Aydin Alatan. 2019. Geospatial object detection using deep networks. In *Earth Observing Systems XXIV*, James J. Butler, Xiaoxiong (Jack) Xiong, and Xingfa Gu (Eds.), Vol. 11127. International Society for Optics and Photonics, SPIE, 15 – 23. https://doi.org/10.1117/12.2530027

[2] O. Barut, L. Zhou, and Y. Luo. 2020. Multi-task LSTM Model For Human Activity Recognition and Intensity Estimation using Wearable Sensor Data. *IEEE Internet of Things Journal* (2020), 1–1.

[3] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority over-Sampling Technique. *J. Artif. Int. Res.* 16, 1 (June 2002), 321–357.

[4] Gerard Draper-Gil, Arash Habibi Lashkari, Mohammad Saiful Islam Mamun, and Ali A. Ghorbani. 2016. Characterization of Encrypted and VPN Traffic using Time-related Features. In *ICISSP*.

[5] J. Fontanarava, G. Pasi, and M. Viviani. 2017. Feature Analysis for Fake Review Detection through Supervised Classification. In *2017 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. 658–666.

[6] N. Gao, L. Gao, Q. Gao, and H. Wang. 2014. An Intrusion Detection Model Based on Deep Belief Networks. In *2014 Second International Conference on Advanced Cloud and Big Data*. 247–252.

[7] Arash Habibi Lashkari, Gerard Draper Gil, Mohammad Mamun, and Ali Ghorbani. 2017. Characterization of Tor Traffic using Time based Features. 253–262. https://doi.org/10.5220/0006105602530262

[8] Hu Ting, Wang Yong, and Tao Xiaoling. 2010. Network traffic classification based on Kernel Self-Organizing Maps. In *2010 International Conference on Intelligent Computing and Integrated Systems*. 310–314.

[9] Mohammad Lotfollahi, Ramin Shirali hossein zade, Mahdi Jafari Siavoshani, and Mohammadsadegh Saberian. 2017. Deep Packet: A Novel Approach For Encrypted Traffic Classification Using Deep Learning. *Soft Computing* (09 2017). https://doi.org/10.1007/s00500-019-04030-2

[10] J. Luo, Y. Liang, W. Gao, and J. Yang. 2014. Hadoop based Deep Packet Inspection system for traffic analysis of e-business websites. In *2014 International Conference on Data Science and Advanced Analytics (DSAA)*. 361–366.

[11] Rusheng Ding and Wenmin Li. 2016. A hybrid method for service identification of SSL/TLS encrypted traffic. In *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*. 250–253.

[12] V. F. Taylor, R. Spolaor, M. Conti, and I. Martinovic. 2016. AppScanner: Automatic Fingerprinting of Smartphone Apps from Encrypted Network Traffic. In *2016 IEEE European Symposium on Security and Privacy (EuroS P)*. 439–454.

[13] V. Tong, H. A. Tran, S. Souihi, and A. Mellouk. 2018. A Novel QUIC Traffic Classifier Based on Convolutional Neural Networks. In *2018 IEEE Global Communications Conference (GLOBECOM)*. 1–6.

[14] S. Turner. 2014. Transport Layer Security. *IEEE Internet Computing* 18, 6 (2014), 60–63.

[15] L. Vu, H. V. Thuy, Q. U. Nguyen, T. N. Ngoc, D. N. Nguyen, D. T. Hoang, and E. Dutkiewicz. 2018. Time Series Analysis for Encrypted Traffic Classification: A Deep Learning Approach. In *2018 18th International Symposium on Communications and Information Technologies (ISCIT)*. 121–126.

[16] P. Wang, F. Ye, X. Chen, and Y. Qian. 2018. Datanet: Deep Learning Based Encrypted Network Traffic Classification in SDN Home Gateway. *IEEE Access* 6 (2018), 55380–55391.

[17] Q. Wang, A. Yahyavi, B. Kemme, and W. He. 2015. I know what you did on your smartphone: Inferring app usage over encrypted data traffic. In *2015 IEEE Conference on Communications and Network Security (CNS)*. 433–441.

[18] 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.

[19] 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.

[20] Xin Wang, Shuhui Chen, and Jinshu Su. 2020. Real Network Traffic Collection and Deep Learning for Mobile App Identification. *Wireless Communications and Mobile Computing* 2020 (19 Feb 2020), 4707909. https://doi.org/10.1155/2020/4707909

[21] Baris Yamansavascilar, M. Amaç Güvensan, Ali Gökhan Yavuz, and M. Elif Karsligil. 2017. Application identification via network traffic classification. *2017 International Conference on Computing, Networking and Communications (ICNC)* (2017), 843–848.

[22] Y. Yang, C. Kang, G. Gou, Z. Li, and G. Xiong. 2018. TLS/SSL Encrypted Traffic Classification with Autoencoder and Convolutional Neural Network. In *2018 IEEE 20th International Conference on High Performance Computing and Communications; IEEE 16th International Conference on Smart City; IEEE 4th International Conference on Data Science and Systems (HPCC/SmartCity/DSS)*. 362–369.

[23] H. Yao, C. Liu, P. Zhang, S. Wu, C. Jiang, and S. Yu. 2019. Identification of Encrypted Traffic Through Attention Mechanism Based Long Short Term Memory. *IEEE Transactions on Big Data* (2019), 1–1.

[24] J. Zhang, C. Chen, Y. Xiang, W. Zhou, and Y. Xiang. 2013. Internet Traffic Classification by Aggregating Correlated Naive Bayes Predictions. *IEEE Transactions on Information Forensics and Security* 8, 1 (2013), 5–15.

[25] Y. Zhang, S. Zhao, J. Zhang, X. Ma, and F. Huang. 2019. STNN: A Novel TLS/SSL Encrypted Traffic Classification System Based on Stereo Transform Neural Network. In *2019 IEEE 25th International Conference on Parallel and Distributed Systems (ICPADS)*. 907–910.

[26] Z. Zhao, R. Anand, and M. Wang. 2019. Maximum Relevance and Minimum Redundancy Feature Selection Methods for a Marketing Machine Learning Platform. In *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. 442–452.