Flow Level, HTTP-2 Classification with Machine Learning Algorithms

Yekta Demirci
Electrical and Computer Engineering
University of Waterloo
Waterloo, ON , Canada
ydemirci@uwaterloo.ca

***Abstract*—In this work, flow level HTTP2 traffic is classified from its predecessor HTTP1. Firstly, flow level data-set is constructed from a publicly available web traffic collection, then several features and Hyper Transfer Protocol(HTTP) version of each flow are found to build ground truth by deep packet inspection(DPI). Afterwards, the obtained flow level traffic is used to train different machine learning algorithms; K nearest neighbours(KNN), Support Vector Machines(SVM), Decision Tree(CART), Random Forest and Artificial Neural Network(ANN). Finally, the predictive power of each model is compared by their accuracy and confusion matrices.**

***Index Terms*—HTTP, traffic classification, machine learning**

I. Introduction and Motivation

Understanding the behaviour of the web has always been an hot topic in computer science.[1][2][3]. Classifying the ongoing network traffic would help operators to offer more efficient[4] and cheaper service to customers, scientists to build better architectures[5] and developers to check quality of an implementation and even to detect anomalies[6] and improve security. The classification can be done in various levels. In this work, application layer protocols HTTP is investigated.

HTTP has been one of the most used application layer protocol in computer networks. Through the evolution of the web, HTTP has also faced several modifications an upgrades. As the web traffic gradually increased, growing demand and workload required new solutions and updates to the currently used infrastructure. To answer such needs, several changes has applied to HTTP and and eventually HTTP2 has merged.

HTTP2 differs from HTTP1.x in multiple ways[7]. One of the major change is; it can send multiple requests in parallel with a single Transmission Control Protocol(TCP) connection. This allows asynchronous data download. Secondly, it adopts binary header structure, so the overhead is decreased. Finally, HTTP2 allows servers to push data. If the server anticipates there are referenced files by the requested one, it may push them instead of waiting for additional requests. Clearly, such major updates would change the behaviour of the web.

This paper demonstrates two main objectives. The first one is comparing the characteristics of HTTP1 and HTTP2 traffic. For this purpose, publicly available large data-set which is collected from a real network is used. From the data-set, size and duration information are extracted and differences between HTTP1 and HTTP2 are compared to see their differences. The second objective is showing the predictive power of several classifying algorithms over HTTP1 and HTTP2, just by using flow level information. The used features are relatively easy to collect, therefore the used methods can be applied in real life when DPI is not available. However, DPI is used in this project to label the flows.

The following sections are organized as follows. Section II gives a discussion about the HTTP and related works. In section III, used classification algorithms and the data-set are explained. Then, in section IV, comparisons between HTTP1 and HTTP2 flows are shown and the outcomes of the classifiers are analyzed. Finally, in section V, the conclusion is given.

II. Background and Related Work

A. HTTP overview

In the early days of computer networking, HTTP was designed to define a protocol to be used between a client and a server to retrieve and deliver basic text files. As the usage of web increased and evolved, higher payloads are started to be sent. In order to meet the needs, HTTP has also evolved. In HTTP1, 1 request could be made with a single connection. Then in time, in order to increase the efficiency, *persistent connection* is introduced by HTTP1.1 where the same connection could be reused to make additional requests. Just with a few upgrades like *pipeline*, HTTP1.1 has been used for many years. Finally in 2015[7], HTTP2 is introduced. The major differences between the old school HTTP1 and new HTTP2 could be listed as multiplexing, binary headers and server pushing. By multiplexing, several virtual bidirectional streams can be formed within a single TCP connection. This enables asynchronous data transfer with a single TCP connection. The second major difference is the header compression. In HTTP1.x, ASCII encoding is used in headers. In HTTP2, the headers are encoded in binary format which can be thought as compression. Finally in HTTP2, servers can push objects where in conventional way, HTTP1.x pulls data by sending requests. Pushing helps to send files faster because in a normal web-page, a base file references some other files. After, a client retrieves the base file, it is likely to request other referenced ones. In HTTP2, the server can foresee this situation and it may push the referenced objects in advance if the client accepts push feature.

B. Related work

The internet network classification can be done in various ways depending on the layer. Generally speaking, two major approaches can be listed as flow based or packet based. Packet based scenario provides more information about traffic like used headers, protocol structures and payload, yet accessing packet information is not easy, primarily due to privacy concerns. In the case of flow based inspections, IP and port numbers of sources and destinations are enough to create flows. Flow duration, size of flow(in terms of byte or packet) could be used as features and payload information is not necessary. In this paper, flow level information with relatively easy to obtain features are used. By this way, the used classification algorithms could be applied to a web traffic without using any DPI.

In the literature, several different algorithms are proposed for classification. Even though, most of the used algorithms are simple Machine Learning(ML) approaches[8], some authors also come up with their own algorithms. Such an example is to classify P2P traffic, mainly to identify BitTorrent users.[9]. In a different work[10], the authors clustered application layer traffic by using unsupervised ML algorithms: K-means and AutoClass clustering. They claimed unsupervised methods are more beneficial since the data is not needed to be labeled beforehand to train models.This eases training procedure, because labelling the traffic data is not trivial, the content might be encrypted or the packet information may not be available. Then, the same authors offered a new approach[11] using expectation maximization based clustering which gave better accuracy than their previous work.

Using encryption is a way to secure content and increase privacy. HTTP2 is defined for both HTTP and HTTPS, however it is mentioned that implementation of HTTP2 must use Transport Later Security(TLS) version 1.2 [12]. TLS is an Internet Engineering Task Force(IETF) standard to provide end to end communication security over networks to prevent eavesdropping and forgery. Therefore, classifying HTTP2 can be considered as classifying encrypted traffic. In a paper[13], authors used graph-comparison approach only by using visible behaviour of the network, such as size, timing and direction of the packets which are similar to the features used in this work.

III. System Design

A. Data-set

Collecting internet traffic data is a challenging task due to several reasons. The first obvious reason is privacy concern. Since the traffic carries several authentication information like bank, social media or email account IDs and passwords, unauthorized interception, access, use, and disclosure of the communication channels are not allowed[14]. The second reason is, the nodes where large amount of data flows are controlled by Internet Service Providers(ISP) and the data is not publicly available. Even if all the legal requirements are solved and necessary permissions are obtained, monitoring and gathering large scale data requires careful maintenance of disk-bandwith, CPU capacity and data reduction[15].

In this paper, publicly available ISCXVPN2016[16] data-set is used. The traffic was captured in 2016 by using *Wireshark* and *tcpdump*, generating 28GB of data where both regular and Virtual Private Network(VPN) sessions are used to obtain 14 different traffic categories like VOIP, VPN-VOIP, P2P, VPN-P2P. In the original work[17], the data was used to understand effectiveness of flow-based time-related features to detect VPN traffic and classify its type like browsing, streaming etc.

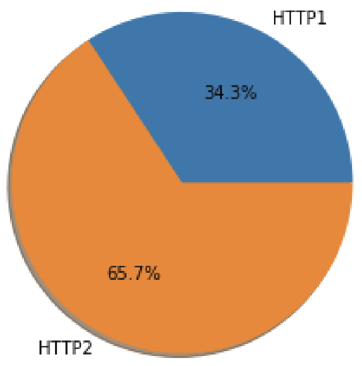

Fig. 1. Distribution of HTTP1.x and HTTP2 of ISCXVPN2016 data-set

Beside ISCXVPN2016 data-set, in order to benchmark, a self monitored data-set is also collected via *Wireshark*. However, the distribution of HTTP1.x and HTTP2 protocols are not homogeneous enough and the total obtained flow number is 10 times smaller than ISCXVPN2016 data-set. Therefore self obtained data-set is not used to train and test the classifiers.

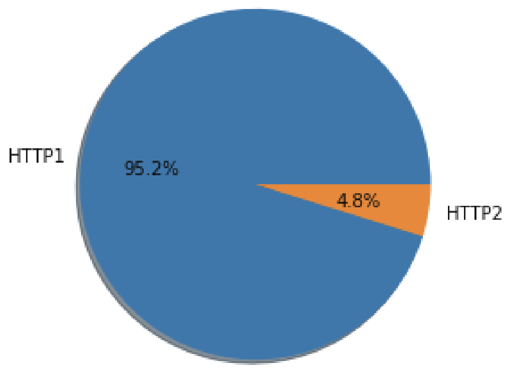

Fig. 2. Distribution of HTTP1.x and HTTP2 of self collected data-set

From ISCXVPN2016 data-set, by using well-known 4 tuple information, source-destination, port-IP numbers unidirectional flows are constructed. Generally, flows are defined as bidirectional for internet traffic classifying purposes[18][19], yet there are also examples of unidirectional flow use[20]. During the construction of flows, because of insufficiency of *RAM*, only the *pcap* files less than 500MB are used. Flows are labelled according to the hints provided by TLS,HTTP1 handshakes and more than 1000 flows are obtained.That was the part when DPI was used.

Features used for classifiers are generally the key for the performance, yet it is not clear what features to use upfront. In a work where authors classified HTTP2 traffic[21], 17 candidate features are extracted and after applying correlation-based feature selection, they claimed that the following features present the most predictive power:(i) number of bytes, (ii)number of packets, (iii) average bytes per packet and (iv) flow duration. In this work, the same features are extracted while building the flows. Because these features do not require deep packet investigation and this would make the used models more applicable.

B. Pre-processing and Classification algorithms

By the light of [21], already a few numbers of features are extracted. However, in order to understand the variation provided by these features, principal component analysis(PCA) and linear discriminant analysis(LDA) are performed over the flows. PCA is a singular value decomposition technique where the features are mapped to principal components by maximizing the variance. In the case of LDA, the intra-class variance is minimized while inter-class variance is maximized. For PCA and LDA, implementations offered by Scikit Learn library of Python are used.[22][23]

Respectively, KNN, SVM, Decision Tree(DT) Random Forest(RF) and ANN are used to classify the traffic. KNN,SVM and ANN are relatively computationally expensive and slower algorithms compared to Decision Trees or Random Forest. Especially, training ANN is not efficient when our goal here is considered. Yet, in order to give an idea and make a comparison, they are all used. It is worth to mention, in a real life use, ANN would not be feasible to implement because of its costly training whereas less computationally expensive algorithms may offer similar performances.

In order to find optimum metrics, the algorithms are trained with several parameters. The data is split into training and test set with 0.8 and 0.2 ratio, respectively. For SVM,DT and RF 10 fold cross validation is used.

In the case of KNN, *euclidean*,*manhattan* and *chebyshev* distance metrics are used with 1 to 31 different neighbours. For SVM, *linear*, *polynomial(degree 3)*, *radial basis function* and *sigmoid* kernels are used with penalty values from 0.1, 0.5, 1, 2, 5, 10, 20, 50 to 60. For DT, depthness of 3, 5, 7, 10 to maximum allowable value are used. Scikit learn adopts optimized version of *Classification and Regression Trees(CART)*, algorithm which is similar to *C4.5*. For RF the same depth parameters as DT are used with 5, 10, 50, 100, 150, 200 trees. Finally, in the ANN, 1 hidden layer is used with 4 neurons. The activation functions in the input and hidden layers are *rectifier* whereas the output layer uses *sigmoid* function. The batch size is set as 25 and 200 epoch is used. Finally, *Adam* optimizer is used with *cross entropy loss* to train the model and different threshold values are used to classify the results of the output node.

For KNN[24], SVM[25], DT[26] and RF[27] Scikit Learn, for ANN[28] Keras libraries offered by Python are used.

C. Evaluation metrics

Accuracy is one of the most popular metrics. However, it may not reflect outcomes in a detailed manner due to class imbalance. After training the classification models with different parameters, the ones which give the highest accuracy are chosen. Then, confusion matrix for each model is found with these parameters. The predicted samples are compared in terms of true discovery rate beside the overall accuracy. True discovery rate tells how much of a class is truly classified. Scikit learn library is used to get confusion matrices. [29]

IV. Results and Discussion

It is clear that HTTP1 and HTTP2 protocols show different characteristics. The first obvious observation is, response flows from the servers carry more packets, bytes and are longer in duration than the client request flows. Secondly, as it can be seen in figure 4 and 5, HTTP2 flows have longer duration and as a result of that, they have more packets. Also there are sharp increases in flow duration graph of both HTTP1.x and HTTP2 server responses. This is probably because of timeout values set by the servers. On the other hand, when size of the flows are considered in terms of bytes, HTTP1.x flows carry more data than HTTP1. To calculate average byte per packet of a flow, total size of flows are divided by the number of packets they have. It is worth to note, HTTP1 packets also carry more byte per packet than HTTP2. This might be due to header compression or implementation level constraints about the maximum allowed byte per packet in HTTP2.

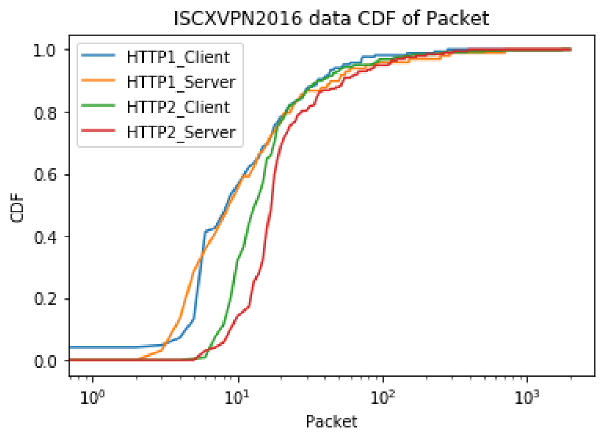

Fig. 3. CDF of packets in flows, ISCXVPN2016 data-set

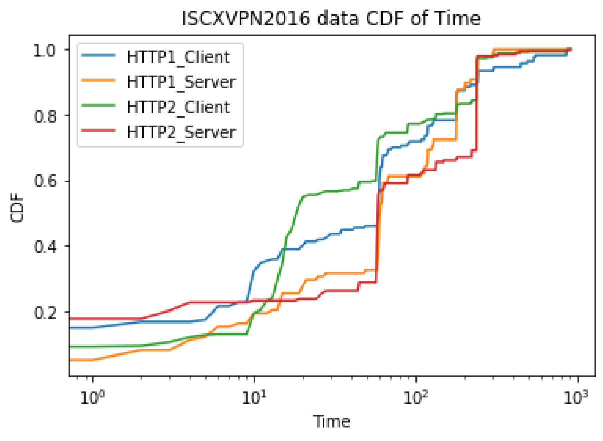

Fig. 4. CDF of flow length, ISCXVPN2016 data-set

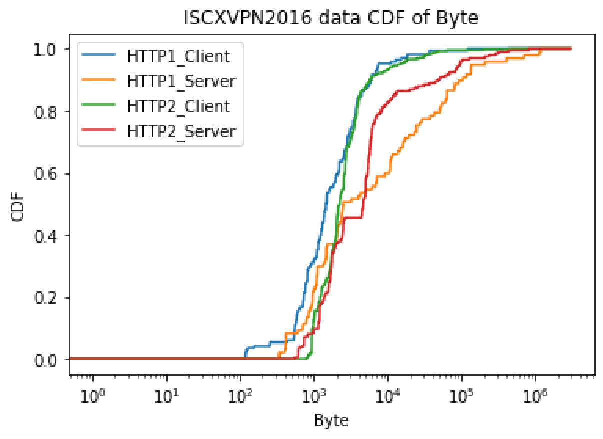

Fig. 5. CDF of bytes in flows, ISCXVPN2016 data-set

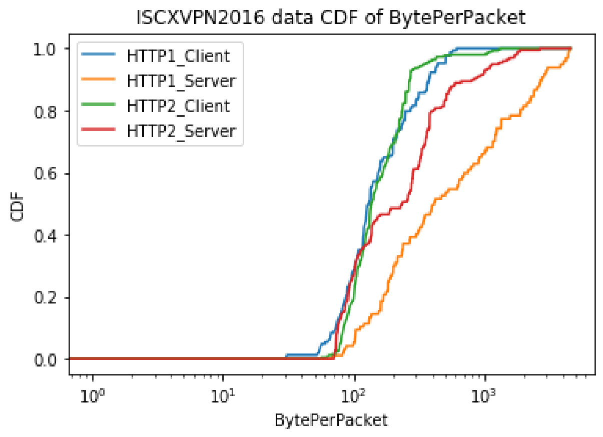

Fig. 6. CDF of bytes per packets of flows, ISCXVPN2016 data-set

According to [21], already a few in number but effective features are extracted. Still, in order to understand the variation among the samples, PCA is applied to the data-set. Since there are 4 used features, 4 PCAs obtained. They explain 45, 30, 16 and 9 per cent of the total variation respectively. As it can be seen in figure 7, a few outliers highly biased the principal components in a negative way. When the scattered data is considered, maximizing the variation does not help to separate the classes for this data-set. PCA could have been re-applied after eliminating of outliers, but there are already a few, 4 features. In addition to PCA, LDA is also applied to the data-set to be compared with PCA. Since, there are only 2 classes in the data-set, a single vector is obtained. In order to visualize it in two dimensions, an axis with random values is created. LDA mapping seems more promising than PCA. As it can be seen in figure 8, left hand side of dimension 1 is mostly HTTP1 and the right hand side has more HTTP2. Yet, they are still not very separable. As a result the models are trained without applying PCA or LDA to the samples.

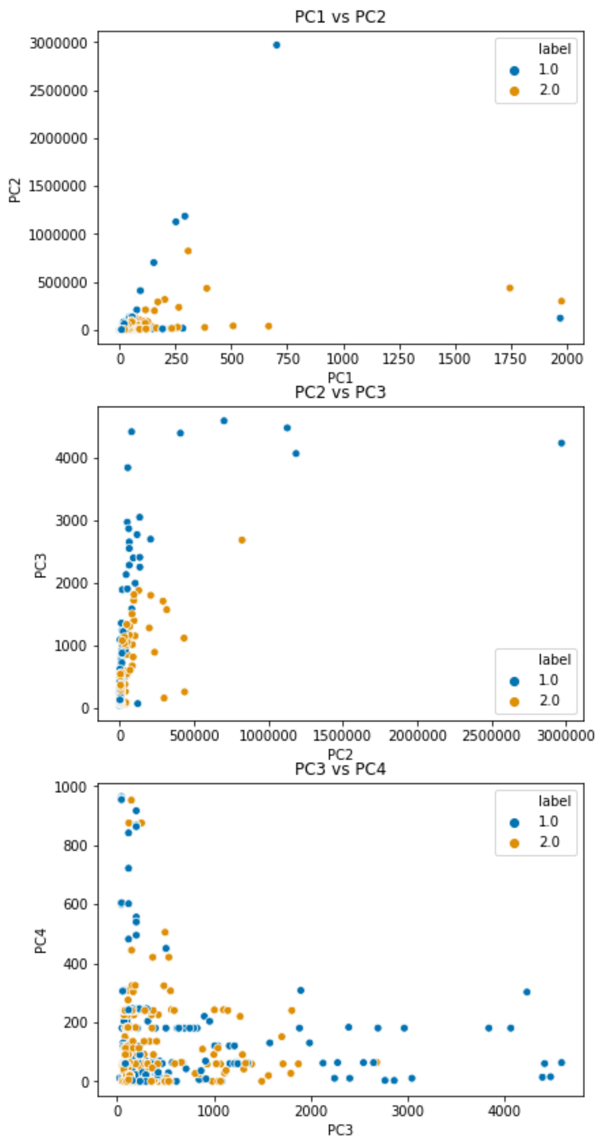

Fig. 7. Scatter plot of PCs, obtained from ISCXVPN2016 data-set

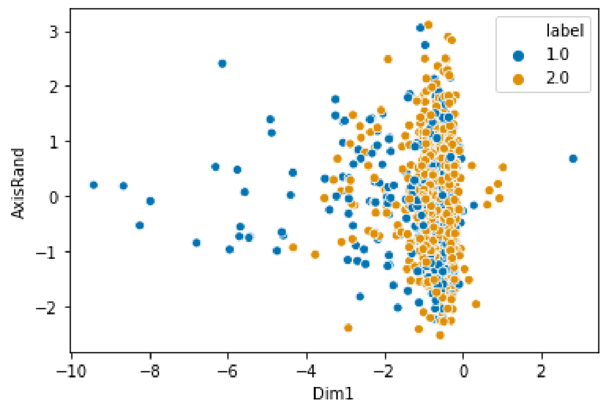

Fig. 8. Scatter plot of LDA results, obtained ISCXVPN2016 data-set

The first trained model is KNN. After training the model with different metrics, the best accuracy is obtained with 3 neighbours on *Manhattan* distance as 91%. As the number of neighbours increase, the accuracy falls. This is because some of the samples from different classes share similar characteristics, therefore including too many neighbours during the classification leads to wrong class. The comparison between different distance metrics and neighbours can be seen in figure 9.

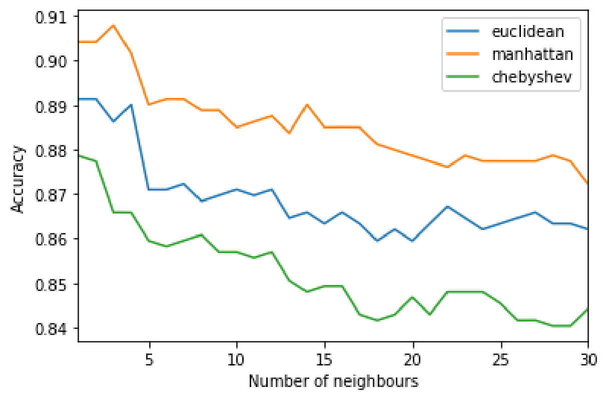

Fig. 9. KNN accuracy with different parameters

The second trained model is SVC. When the scattering of the data is considered, it could be foreseen SVC would not perform well. Still with radial basis kernel(aka squared-exponential kernel), the accuracy reaches to %80 with high C value. C value determines the margin between the hyper-planes. In other words, C value affects the lenience of the model. As C value is increased, the tolerance decreases and the model starts to over fit. Therefore, getting higher accuracy with high C value does not show a reliable performance.

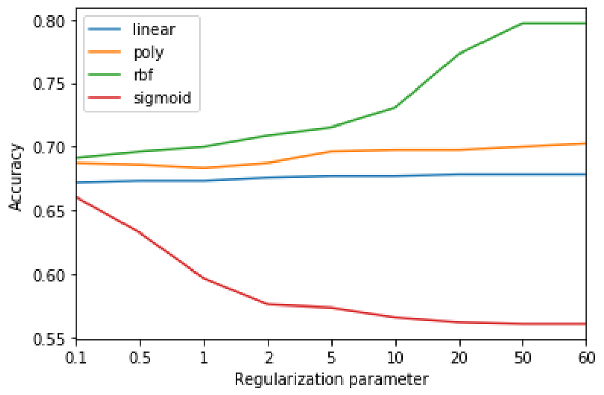

Fig. 10. SVC accuracy with different parameters

The third model is decision tree. Different values for depth are used and the best accuracy is obtained as %92 with depth 10. As the number of depth increases, the accuracy also increases. However with large values of depth, over fit tends to occur due to having many leaves. Here, over fit is the scenario when similar samples are separated into different leaves and cause unnecessary branching.

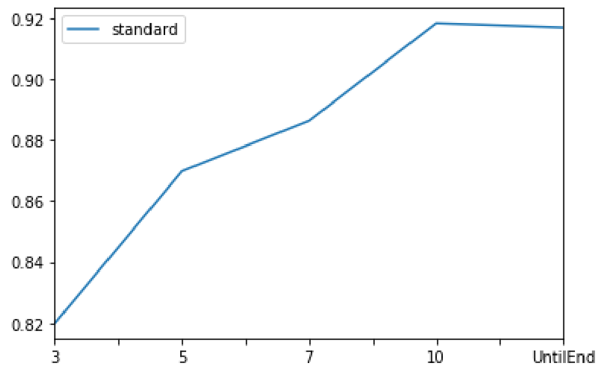

Fig. 11. Decision tree accuracy with different parameters

For the random forest model, different number of trees and depth values are tested. The best accuracy is obtained as %94 with the maximum depth and 150 trees . It may seem like random forest gave slightly better accuracy than decision tree. However when the confusion matrices are examined it can be seen, random forest over fits to HTTP2 data. Since there are more HTTP2 samples than HTTP1 in the data-set, with high tree number, the model is biased towards HTTP2. As a result, it tends to mis-classify HTTP1 traffic but seems like giving better accuracy. Also, the other reason of this problem is the sample size. In random forest, instead of relying on 1 tree, several trees are constructed and the variation among samples are aimed to be extracted better. It works better with higher samples size, otherwise information kept by new trees would be just repetition of some previous trees.

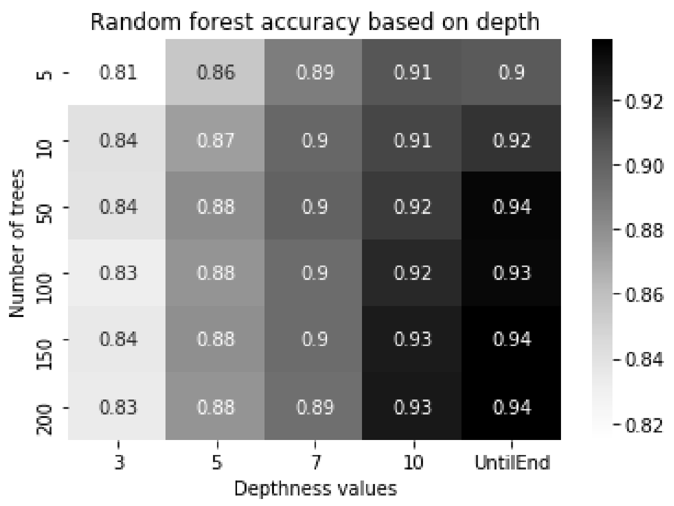

Fig. 12. Random forest accuracy with different parameters

Confusion matrices of each model are found by using the most optimum parameters. Decision tree clearly outperforms the rest. In addition to its high accuracy, the parameter tuning and model testing is faster than the others. Random forest can also be implemented however it is not necessary when the performance of a single decision tree is already well enough. Beside DT and RF, KNN also has high accuracy, however it is slower. In other words more computationally expensive to classify new samples than DT or RF. ANN is slower to train and gives %85 accuracy which is not any better than KNN or DT. It is well understood why there are not much use of ANN in the literature. Also, SVC is clearly not a good choice to classify HTTP traffic.

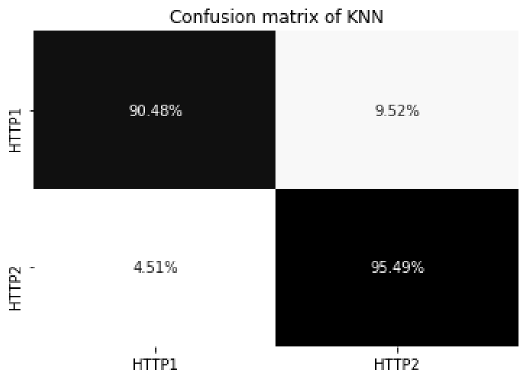

Fig. 13. Confusion matrix of KNN

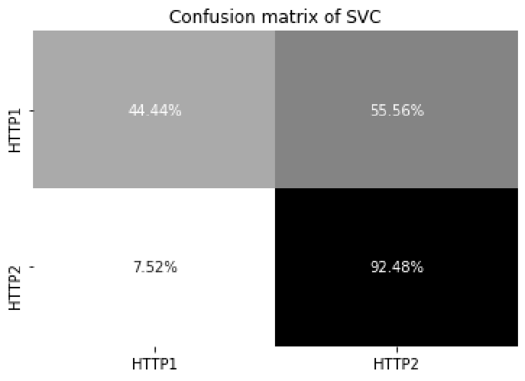

Fig. 14. Confusion matrix of SVC

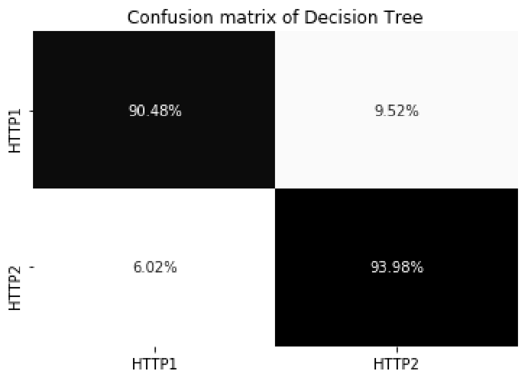

Fig. 15. Confusion matrix of decision tree

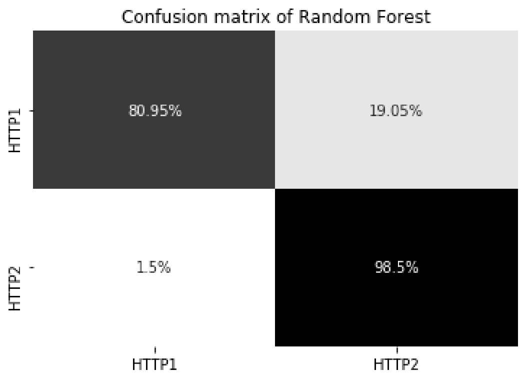

Fig. 16. Confusion matrix of random forest

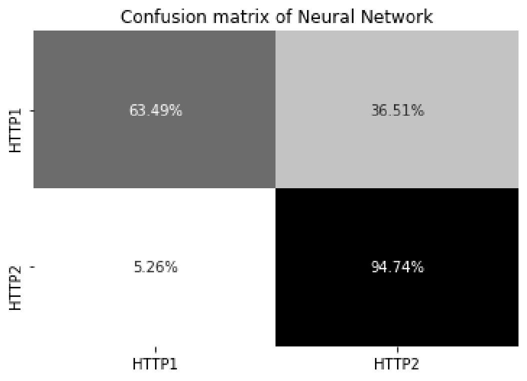

Fig. 17. Confusion matrix of ANN

V. Conclusion

A comparison between flow level HTTP1 and HTTP2 traffic is presented. Due to increase of encryption over the networks and inefficient payload examinations, features which could be extracted without a deep packet investigation are used. Then, considering the differences between the protocols, different classifier models are trained and their performances in terms of accuracy are found. When accuracy is considered as the optimization constraint, decision trees are shown to be the most suitable option among the other used classifiers in this work. In addition to their accuracy, requiring less computational power makes decision trees more appealing.

In this work, a publicly available static data-set is used. As a future work, dynamic data can be used to detect live traffic. Also, the adoption of HTTP2 continues and the characteristics of the web continuously change. Therefore, it is important to train models regularly to keep them up to date. In this work, such aspects are omitted. Also running and training time of the classifiers could be measured in a more detailed way. Because in the case of monitoring a live traffic, used models ought to make fast predictions to cache flow speed, specially over high-band networks.

All in all, this work gives an introductory step to understand traffic classification and provides a methodology to classify HTTP2 and HTTP1 traffic based on flow level characteristics. The implemented codes can be seen in [30].

References

[1] M. F. Arlitt, C. L. Williamson "Internet web servers: Workload characterization and performance implications", IEEE/ACM Transactions on networking, 1997, vol. 5, p. 631-645

[2] T. TT. Nguyen, G. Armitage, "A survey of techniques for internet traffic classification using machine learning", IEEE communications surveys tutorials, Oct. 01 2008, vol. 10, p.56-76

[3] A. Callado, C. Kamienski, G. Szabó, B. P. Gero, J. Kelner, S. Fernandes, D. Sadok, "A survey on internet traffic identification", IEEE communications surveys tutorials, 2009, vol. 11, p. 37-52

[4] T. Mangla, E. Halepovic, M. Ammar, E. Zegura, "eMIMIC: Estimating HTTP-based Video QoE Metrics from Encrypted Network Traffic", 2018, 2018 Network Traffic Measurement and Analysis Conference (TMA), IEEE, p. 1

[5] B. Yeganeh, R. Rejaie, W. Willinger, "A view from the edge: A stub-AS perspective of traffic localization and its implications", Jun 21 2017, 2017 Network Traffic Measurement and Analysis Conference (TMA), IEEE, p. 1

[6] P. Casas, P. Fiadino, A. D'Alconzo, "Machine-Learning Based Approaches for Anomaly Detection and Classification in Cellular Networks", April 07 2016, 8th Traffic Monitoring and Analysis (TMA), Louvain La Neuve, Belgium

[7] M. Belshe, R. Peon, M. Thomson, BitGo, Google, "Hypertext Transfer Protocol Version 2 (HTTP/2) " May 30 2015, Retrieved from https://tools.ietf.org/html/rfc7540

[8] R. Boutaba, M. A. Salahuddin, N. Limam, A. Ayoubi, N. Shahriar, F. Estrada-Solano, O. M. Caicedo "A comprehensive survey on machine learning for networking: evolution, applications and research opportunities", Jun. 21 2018, Journal of Internet Services and Applications vol. 9 p.16

[9] X. Tianming, R. Song , "A Method of P2P Traffic Identification on Internet Based on the Deep Flow Inspection", March 2009, International Communication on Software and Networks, IEEE

[10] J. Erman, M. Arlitt, A. Mahanti, "Traffic classification using clustering algorithms", Sep. 11 2006, Proceedings of the 2006 SIGCOMM workshop on Mining network data, p. 281

[11] J. Erman, A. Mahanti, M. Arlitt, "Internet traffic identification using machine learning", Oct. 27 2006, Global Telecommunications Conference, IEEEm p. 1

[12] M. Belshe, R. Peon, M. Thomson, BitGo, Google, May 30 2015, "Hypertext Transfer Protocol Version 2 (HTTP/2)" Retrieved from https://http2.github.io/http2-spec/TLSUsage

[13] M. Gebski,A. Penev, R.K. Wong, "Protocol Identification of Encrypted Network Traffic", 2006, International Conference on Web Intelligence, IEEE

[14] Electronic Communications Privacy Act of 1986 (ECPA)

[15] A. Moore, J. Hall, C. Kreibich, E. Harris, I. Pratt, "Architecture of a network monitor", April 06 2003, Passive Active Measurement Workshop

[16] ISCXVPN2016 data is available at: https://www.unb.ca/cic/datasets/vpn.html

[17] G. Draper-Gil, A. H. Lashkari, M. S. I. Mamun, A. A. Ghorbani, "Characterization of encrypted and vpn traffic using time-related", 2016, Proceedings of the 2nd international conference on information systems security and privacy (ICISSP)

[18] A. McGregor, M. Hall, P. Lorier, J. Brunskill ,2004, "Flow clustering using machine learning techniques in Passive and Active Network Measurement", Lecture Notes in Computer Science, vol. 3015, p.205–214.

[19] F. Palmieri, U. Fiore, "A nonlinear, recurrence-based approach to traffic classification",2009, Computer Networks: The International Journal of Computer and Telecommunications Networking, vol. 53(6) p.761–773.

[20] M. Kim, Y. J. Won, H. Lee, J. W. Hong, R. Boutaba, "Flow-based characteristic analysis of Internet application traffic ", 2004 ,Workshop Chair

[21] J. Manzoor, I. Drago, R. Sadre, "How HTTP/2 is changing Web traffic and how to detect it", June 21 2017, Network Traffic Measurement and Analysis Conference (TMA), IEEE , p. 1-9

[22] https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html

[23] https://scikit-learn.org/0.16/modules/generated/sklearn.lda.LDA.html

[24] https://scikit-learn.org/stable/modules/neighbors.html

[25] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[26] https://scikit-learn.org/stable/modules/tree.html

[27] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassif

[28] https://keras.io/models/sequential/

[29] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

[30] https://github.com/YektaDemirci/ECE656-Project.git