applied sciences

MDPI

Article

QUIC Network Traffic Classification Using Ensemble Machine Learning Techniques

Sultan Almuhammadi [1,2,*,†], Abdullatif Alnajim [1,†] and Mohammed Ayub [1,†]

[1] College of Computing and Mathematics, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia
[2] Interdisciplinary Research Center for Intelligent Secure Systems, King Fahd University of Petroleum and Minerals, Dhahran 31261, Saudi Arabia
* Correspondence: muhamadi@kfupm.edu.sa
† These authors contributed equally to this work.

Abstract: The Quick UDP Internet Connections (QUIC) protocol provides advantages over traditional TCP, but its encryption functionality reduces the visibility for operators into network traffic. Many studies deploy machine learning and deep learning algorithms on QUIC traffic classification. However, standalone machine learning models are subject to overfitting and poor predictability in complex network traffic environments. Deep learning on the other hand requires a huge dataset and intensive parameter fine-tuning. On the contrary, ensemble techniques provide reliability, better prediction, and robustness of the trained model, thereby reducing the chance of overfitting. In this paper, we approach the QUIC network traffic classification problem by utilizing five different ensemble machine learning techniques, namely: Random Forest, Extra Trees, Gradient Boosting Tree, Extreme Gradient Boosting Tree, and Light Gradient Boosting Model. We used the publicly available dataset with five different services such as Google Drive, YouTube, Google Docs, Google Search, and Google Music. The models were trained using a different number of features on different scenarios and evaluated using several performance metrics. The results show that Extreme Gradient Boosting Tree and Light Gradient Boosting Model outperform the other models and achieve one of the highest results among the state-of-the-art models found in the literature with a simpler model and features.

Keywords: QUIC traffic; traffic classification; machine learning; ensemble learning

Citation: Almuhammadi, S.; Alnajim, A.; Ayub, M. QUIC Network Traffic Classification Using Ensemble Machine Learning Techniques. *Appl. Sci.* **2023**, *13*, 4725. https://doi.org/10.3390/app13084725

Academic Editors: Tarek Gaber, Shu-Chuan Chu and Chin-Shiuh Shieh

Received: 18 January 2023
Revised: 4 March 2023
Accepted: 5 March 2023
Published: 9 April 2023

1. Introduction

Network traffic classification plays an essential role in any network administration system. It can be used in different areas, such as quality of services (QoS), intrusion detection, and malware detection. Generally, different protocols are used for network transportation, such as TLS/TCP or UDP protocols. However, with the rapid growth of Internet users around the globe, some of these protocols may suffer from some limitations, such as latency in TLS/TCP [1]. Therefore, alternative protocols were proposed in the literature to overcome this limitation [2].

The QUIC protocol developed in 2012 by Google [3] is an alternative protocol that improves the performance of web-based applications. It makes use of UDP, which allows the connection to enhance the performance of many online applications by replacing the TCP three-way handshake with a single UDP round-trip [4]. Moreover, the level of security protection in QUIC is the same as the one provided by TLS 1.3. It is worth mentioning that QUIC creates new challenges for information security practitioners in managing and administering communication network functionalities that are very crucial for intrusion detection, anomaly detection, malware detection, QoS provisioning, pricing, etc. [5].

Nowadays, the QUIC protocol is widely used on the Internet [3]. The traffic classification for such a protocol is important since the amount of traffic going through it is increasing. This paper targeted the QUIC protocol since it is used in a lot of social media

apps that are used by the public such as Instagram and Facebook [6]. Additionally, the main objective to develop QUIC was to overcome the latency problems of other protocols [3]. This indicates that deeper analysis might be needed to test the security of that protocol. The ability to reveal the service producing some traffic may threaten one of the main security goals, which is confidentiality. For that purpose, we aim to test how confidential QUIC is against machine learning techniques to identify secure traffic. In the past, network traffic classification techniques were applied based on the TCP/UDP port numbers. However, the QUIC protocol is more complex than the traditional network protocols, which means the classical techniques may not be efficient for classifying traffic that uses this protocol. Machine learning (ML) and deep learning (DL) techniques have proven their reliability and robustness in many real-world problems. Therefore, it is efficient to utilize such techniques in the QUIC traffic classification. However, one problem with classical ML is that they are subjected to over-fitting. On the other hand, DL techniques require a large volume of data to produce better results. For that purpose, a better approach is to train models that are more robust than classical ML and do not require the same amount of data required as DL, such as ensemble machine learning techniques.

In this paper, we examine the QUIC traffic classification using five ensemble machine learning techniques, namely: Random Forest (RF), Extra Trees (ET), Gradient Boosting Tree (GBT), Extreme Gradient Boosting Tree (XGBT), and Light Gradient Boosting Model (LGBM). We use a publicly available dataset to experiment and validate our ensemble models using four evaluation metrics: precision, recall, F1-score, and accuracy. The models were trained using 15, 30, 40, 50, 60, 80, 100, and 120 packets as features. In addition to that, the models were trained and tested using two different training and testing scenarios, i.e., holdout and cross-validation. Moreover, a comparative analysis of the models is conducted. According to our exhaustive and extensive literature search and review, the study of QUIC traffic classification using machine learning is very limited. Especially, award-winning XGBT and LGBM are not explored in QUIC traffic classification. To the best of our knowledge, this is the first of its kind that studies encrypted QUIC multiclass classification using state-of-the-art award-winning machine learning techniques. In the literature, other methods may need more requirements to achieve higher results, such as a greater amount of data, more features, or more complex model architecture. These requirements may not be available at some networks. On the other hand, our results are directly comparable with the related works in the literature, achieving distinguishable results compared to [5,7,8], and closely comparable results with [9,10] with much simplistic model architecture. The highest accuracy and F1-score we achieved is 99.40%. We summarize the contribution as follows: We investigated ensemble machine learning for QUIC traffic classification using a different number of packets and fewer features. Our experimental results show that XGBT and LGBM are the best models for encrypted traffic classification even considering very few packets (15 packets). Additionally, we also investigated training and testing time and model size. The training and testing time for LGBM is 69 s and 0.017 s, respectively, which shows LGBM can be deployed in a real-time scenario. In comparison with other works in the literature, our model shows the highest performance in terms of accuracy and F1-score. Furthermore, we made our models available (https://drive.google.com/drive/folders/1NkJCWY3F-j2CWDqKAJ4D5Pi4azV8zHBh?usp=share_link accessed on 20 February 2023), and they can act benchmarks in inferencing on other data or in comparisons with other techniques. The remainder of the paper is structured as follows. Section 2 provides a short background on QUIC protocol. While Section 3 discusses the previous works related to the topic followed by methodology in Section 4, Section 5 elaborates the details about the experiments. While we discuss and analyze the experimental results in Section 6, before concluding the paper in Section 7.

2. QUIC Protocol

Network traffic can be classified based on protocols (e.g., QUIC or TCP), traffic types (e.g., voice, video, or text), applications (e.g., YouTube, Facebook, or Twitter), etc. [11].

However, due to the encrypted traffic nature of QUIC, many of the proven network traffic analysis tools such as Wireshark lose visibility or functionality. Moreover, this encryption in QUIC makes classical traffic classification methods that use port numbers and payloads as features ineffective for real-world applications. As outlined in [3], the web traffic is currently increasing towards QUIC. Therefore, it is undeniable that most of such traffic is legitimate, and we cannot, therefore, outright block traffic based on inefficient analysis methods, which is likely to create more harm than benefits.

Based on the above discussion, it is important to study QUIC traffic classification using ML techniques that produce learned models using available data accurately predict unseen cases. There are many techniques, among which ensemble techniques are more robust and efficient as they make the decision based on the combination of multiple single models. As will be discussed in the related works, all the previous works found in the literature tend to utilize deep learning, except [4]. However, their work was specified for intrusion detection only, rather than traffic classification. Therefore, we are motivated to investigate QUIC network traffic classification using ensemble ML techniques. To the best of our knowledge, no work uses ensembles for QUIC traffic classification on this specific dataset. As mentioned earlier, the DL techniques require a large volume of samples for training. Since this work utilizes ensembles, we have achieved highly impressive results using relatively few data samples.

Google's QUIC protocol is becoming popular as a potential alternative to encrypted services that use TCP, TLS, and HTTP/2 [3]. Recent global internet statistics show that QUIC accounts for more than 30% in Europe, Africa, and the Middle East [6]. The QUIC traffic was not limited to google services only, but other public social media apps, such as Instagram or Facebook, are using QUIC as well [6]. This information indicates the significance of QUIC around the globe and raises the need for security analysis for such a widespread protocol. The main objective of introducing such a protocol was to upgrade the performance of previous protocols and enhance the customers' experience [3]. Additionally, QUIC was built to address other protocol problems such as handshake delay, Head-of-line blocking delay, and implementation entrenchment [3]. Experimental results showed how QUIC can minimize the latency on different types of traffic whether they were handshakes or media [3]. The QUIC protocol is a UDP-based encrypted protocol where it replaces the three-way handshakes with a one-way round trip. The connection establishment in QUIC starts with the client initiating a connection to the server. If the client connects for the first time to the server, then one round trip time (1-RTT) is established, otherwise, if the user tries to acquire access to a server with an already existing connection within a certain time period, then 0-RTT is established [4]. The 1-RTT and 0-RTT can be summarized as follows [4]:

1-RTT: First, the user (client) sends a blank hello message with a random connection ID that is used to encrypt the message. Then, the server replies with a rejection message, which includes the source address token (STK) and the server certificate (SCFG). In the next step, on receiving the rejection message, the user checks the certificate and replies back with another hello message that contains a nonce and the user's Diffie–Hellman key values. Finally, the encrypted connection is initiated, and the timeout period is specified for closing the connection.

0-RTT: The 0-RTT starts when a client establishes a connection with a server that has a previous connection with it and during the specified period of the idle_timeout. The connection initiates from the user side by sending a hello message containing all the required information. (i.e., STK, SCFG, new connection ID, and the client's Diffie–Hellman values). Next, the sent message is verified on the server side. If the verification is successful, then the encrypted connection will be established. Otherwise, the server will reject the connection, and 1-RTT is required again. Figure 1, which is adapted from [4], illustrates both connection steps.

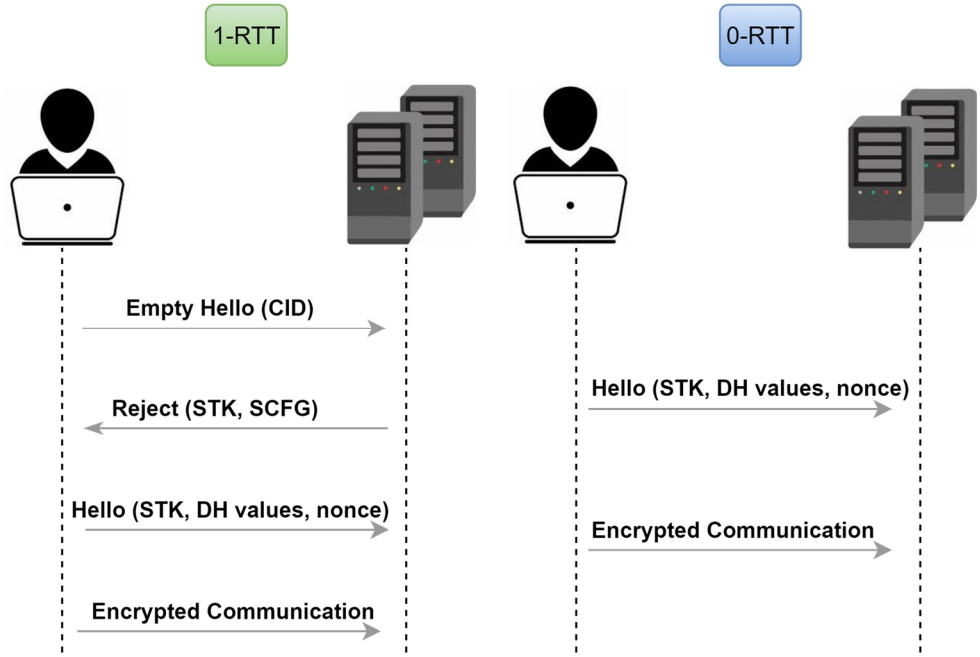

Figure 1. QUIC connection: In 1-RTT, the user sends an empty hello message and waits for the server's rejection message. Then, the user sends the actual hello message to establish the connection. In 0-RTT, the client establishes the connection by sending the actual hello directly, without the empty hello message.

QUIC traffic is currently widely used in most Google services, supported by most web browsers, and is on the way to IETF standardization [12]. According to [3], around 7% of Internet traffic is QUIC. Other advantages of QUIC traffic include its performance, security, and stream multiplexing [3]. On the other hand, QUIC may suffer from some limitations. For instance, it was found that the CPU utilization in QUIC was around 3.5 higher than in TLS/TCP [3]. This could be due to the structure of QUIC and some operations that are involved in the protocol, such as the cryptographic process, transferring UDP packets, and keeping track of the internal states of QUIC [3].

3. Related Work

In this section, the literature related to the QUIC network traffic classification is discussed.

One such research is the one conducted by V. Tong et al. [13], where a two-stages traffic classification technique based on random forests and convolutional neural network (CNN) was proposed. Experiments on different QUIC-based services showed that the proposed technique achieves accuracy as high as 99%. S. Rezaei and X. Liu [8] proposed a multi-task traffic classification technique based on CNN that predicts the bandwidth requirements, flow duration, and traffic class. The proposed framework outperformed both single-task and transfer learning approaches on ISCX and QUIC public datasets, achieving accuracy as high as 90%. The same authors proposed a semi-supervised method based on CNN [5]. The experimental results showed that the technique is promising even when it is trained on a small dataset (20 samples per class).

Another method was proposed by L. Al-bakhat and S. Almuhammadi [4] for intrusion detection based on a fingerprinting method using Random Forests, Decision Trees, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), AdaBoost, and Multi-layer Perceptron (MLP). Comparative analysis showed that KNN outperforms the others in all the evaluation metrics achieving a score of 99.56%. On the other hand, I. Akbari et al. in [9] proposed a feature engineering approach based on Long-Short term memory (LSTM) and

CNN that is generalizable to encrypted data web protocols. The proposed model was evaluated over QUIC traffic, where it achieved more than 99% accuracy.

Another work was proposed by Secchi et al. [7]. The authors developed SVM, KNN, Random Forest, and neural network models for the task of encrypted traffic classification. The proposed models were tested on QUIC traffic where they showed promising results, as they all achieved more than 97% accuracy. Tawhid and Shahriar [10] proposed a self-supervised approach for encrypted network traffic classification with few labeled data. The proposed deep learning model achieved 98% on the QUIC dataset in terms of accuracy, outperforming the baseline by 3%.

Other literature is also found in the domain, which targeted the network traffic classification but is not specific to QUIC traffic. For instance, Deep Packet [14] is a network classification framework proposed based on CNN and stacked autoencoder (SAE). The proposed model outperformed other state-of-the-art models on the "ISCX VPN-nonVPN". N. Williams et al. [15] studied the impact of feature selection in the traffic classification task and conducted a comparative study on C4.5, Bayes Network, Naïve Bayes, and Naive Bayes Trees. M. Lopez-Martin et al. [16] studied the network traffic classification on IoT devices by combining both CNN and RNN. It was interesting to observe that the model performs well even on a small subset of data (5–15 packets). Izadi et al. [17] proposed a method to distinguish between VPN and non-VPN encrypted traffic on the "ISCX VPN-non-VPN" dataset. The method combines the ant-lion meta-heuristic algorithm (ALO), the self-organizing map algorithm (SOM), and CNN for feature extraction and classification. Experimental results showed an accuracy of 98% on the test data. Similar work was proposed in [18] on the same dataset that combines data fusion methods and deep learning for traffic classification. In this work, Deep Belief Network (DBN), CNN, and MLP were employed to classify network traffic. Then, Bayesian decision fusion was used for the final results. The proposed method achieved an accuracy of 97%. Sun et al. [19] approached the network traffic classification problem by utilizing deep learning in the detection process. The features were extracted from the aggregated packets in the same direction and then transformed into images to be used in CNN. When tested over OpenVPN and ISCX-tor datasets, the proposed method achieved 97.20% and 93.31%, respectively, outperforming other state-of-the-art approaches. On the other hand, Liu et al. [20], proposed a multiclass imbalanced and concept drift network traffic classification framework based on online active learning (MicFoal). Experimental results showed that MicFoal is an efficient algorithm and performs better than several state-of-the-art approaches.

From the above review of literature, it is noteworthy that there are a very limited number of papers that focus on QUIC traffic classification using machine learning. Most of the works are on deep learning techniques and only reference [4] used ML for intrusion detection using QUIC traffic. Therefore, we can conclude that the ensemble techniques are under-studied for encrypted classification in general and QUIC traffic classification in specific. In the coming sections, more details about ensemble techniques and QUIC traffic classification are entertained.

4. Methodology

To accomplish the proposed research, the following high-level methodological steps are mechanized. The high-level steps are illustrated in Figure 2. From the raw traffic data, features are extracted for a different number of packets and are, subsequently, fed to different ensemble algorithms. It is worth mentioning that we trained all models in both holdout and cross-validation setting to come up with reliable models with impressive performance. The testing scores are analyzed and compared using four different evaluation metrics.

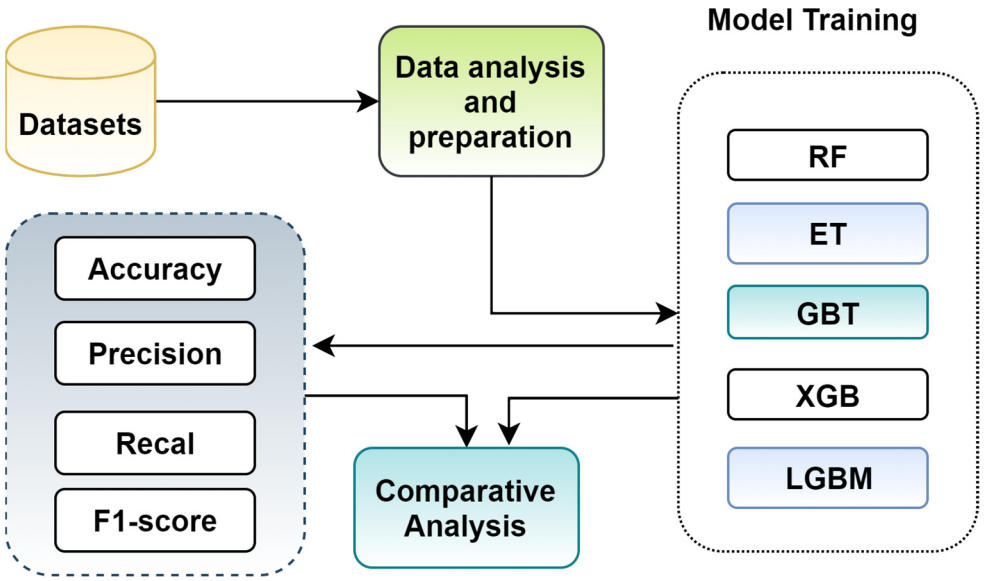

Figure 2. High level methodological steps.

4.1. Ensemble Techniques

In ensemble ML techniques, the classification or regression decision is made by a group of models rather than a single model. These ensemble models work on three methods: bagging, stacking, and boosting [21,22]. Bagging is based on random sampling with replacement, whereas boosting is based on sequential modeling. In other words, in the bagging method, a subset of training samples is selected from the original samples with replacement, and the final decision is based on majority voting [23]. In boosting technique, the model converts a weak learner into a strong learner by sequentially creating models so that the final model has the highest accuracy [24]. Stacked, or stacked generalization, is also a common ensemble method where a combined trainable learner is formed from individual learners. Individual learners are called first-level learners and the combined ones are called meta-learners. Meta-learners can be used to identify which classifiers are reliable and which are not. It is possible to use trainable combiners to identify which classifiers are most likely to succeed in which region of the feature space and combine them accordingly [22]. To visualize the three different methods, an illustration is given in Figure 3.

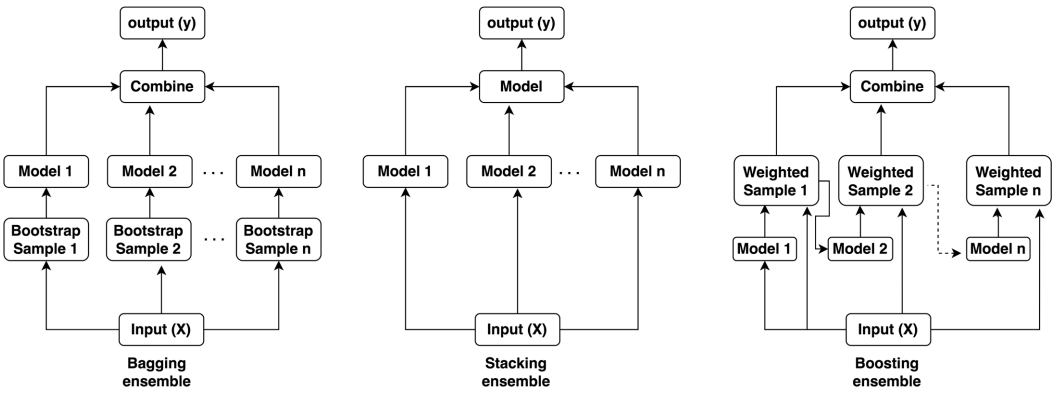

Figure 3. Different ensemble methods.

4.1.1. Random Forest

It is the combination of single tree predictors that works on bagging techniques [23]. RF takes random records from the training set, and, for each set of samples, a respective decision tree is built. In the end, the final output is taken from the results of each decision

tree by taking a majority vote for the classification problem or averaging for the regression problem. The working principle of RF is illustrated in Figure 4.

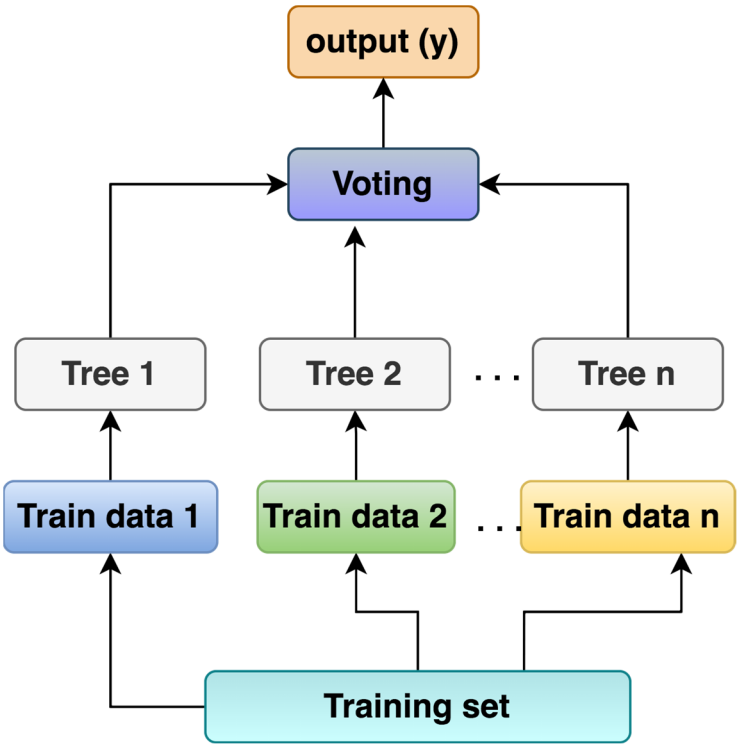

Figure 4. Random forest working principal illustration.

4.1.2. Extra Trees

It is an ensemble technique where different sub-samples of data are fit using a randomized decision tree and uses an averaging technique to calculate accuracy and avoid over-learning. In Extra Trees, the trees are built based on a random subset of features without replacement, and a split also is made based on random subsets of the feature at every node. These are the two points that make it different from the random forest that builds random trees with the replacement of features and splits based on the optimum (best) split rather than random split.

4.1.3. Gradient Boosting Tree

The GBT is also an ensemble technique that uses a boosting method to improve a decision tree (DT), with the concept of combining weak models into a single strong consensus model. In GBT, the task of each tree is to reduce the error of the previous tree rather than develop a new optimized tree. The final model aggregates the outcomes of the previous step, thereby obtaining the notion of a stronger learner [25].

4.1.4. Extreme Gradient Boosting Tree

It is also based on the gradient boosting technique and was initially proposed in [26,27]. This ensemble became popular in all machine learning research areas due to its faster training, convergence, and boosting performance. XGBT uses an elastic regularization method (L1 and L2) to avoid overfitting, which, in turn, gives better performance.

4.1.5. Light Gradient Boosting Model

It was initially proposed by Microsoft [28] and has many of XGBT's advantages. The main difference lies in the way the trees are constructed; trees are grown level-wise in XGBT, whereas, in LGBM, trees are grown leaf wise. Gradient-based One-Side Sampling

(GOSS) and Exclusive Feature Bundling (EFB) are two major reasons that LGBM has faster execution and higher accuracy [28].

4.2. Model Evaluation

The following four established metrics are used for model evaluations. Four terms are used to define the evaluation metrics, which are true positive (*TP*), true negative (*TN*), false positive (*FP*), and false negative (*FN*). TP in simple term is classifying the positive sample as positive (for example, Google Music traffic as Google Music), TN is classifying the negative sample as negative (all traffic other than Google Music), FP is falsely classifying as positive samples (any of all other traffic as Google Music traffic), and FN is falsely classifying as negative samples (Google Music traffic as any of all other traffic).

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1\ score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

5. Experiments

5.1. Environment Setup

All the experiments were executed using three different environments: (i) ASUS i7-6700HQ with a NVIDIA GEFORCE GTX 950M, (ii) Apple i5 MacBook Pro 2015, and (iii) Google Collab. As the implementation platform, the Python programming language is used.

5.2. Hyperparameters Selection

All the models were built using the default values for hyperparameters provided by the scikit-learn package in Python. We aimed to create the models as simple as possible to test if the simpler models can classify the traffic with high accuracy, and that is the purpose of using the default parameters. The list of hyperparameters for all the models is listed below, whereas the selected values for those parameters are mentioned in Table 1.

Table 1. Hyperparameters values for each model.

Model	n_Estimators	Criterion	Max Depth	Min Samples Split	Min Samples Leaf
RF	100	Gini	None	2	1
ET	100	Gini	None	2	1
GBT	100	friedman_mse	3	2	1
XGBT	100	friedman_mse	3	2	1
LGBM	100	friedman_mse	3	2	1

- n_estimators: Number of trees in the model.
- criterion: Quality measurement of the split.
- max_depth: Maximum depth of the tree.
- min_samples_split: Minimum number of samples required to split.
- min_samples_leaf: Minimum number of samples required for a leaf.

5.3. Data Preparation

We used the data collected at the University of California at Davis [5] (https://drive.google.com/drive/folders/1Pvev0hJ82usPh6dWDlz7Lv8L6h3JpWhE accessed on 16 April 2022) in [8], where five different types of QUIC traffic types were classified using machine learning techniques in addition to bandwidth and duration. The encrypted traffic flows were based on four Google services and one YouTube service, the details are shown in Table 2. The data for each service was organized in different folders in a text file format, where each text file data represents a single flow. In each text file again, there are multiple two-way communication data consisting of a time-stamp, relative time (from the first packet), and packet length. The data was preprocessed by applying the same method used in [8]. We used the inter-arrival time, packet length, and direction of each packet as features for the multi-class classification task. In more detail, for each packet, two features were used; inter-arrival time and packet length combined with direction. Both packet length and inter-arrival times were normalized using 1434 bytes and 1 s, respectively, as in [8]. Header information, statistical features such as bandwidth and duration of flow, and payload can be used as features. However, consideration of these as features required the observing of the whole duration of flow, which is undesirable for online encrypted traffic classification. Moreover, accuracy tends to be lower [11] when header information is used as features, and, therefore, we opted for using the said two features in this study as a further exploration.

Table 2. No. of instances used for different traffic types.

Traffic	Label	Training Size	Testing Size	Sum
Google Drive	1	1434	100	1534
YouTube	2	877	100	977
Google Doc	3	1021	100	1121
Google Search	4	1715	100	1815
Google Music	5	392	100	492
Total		**5439**	**500**	**5939**

5.4. QUIC Traffic Classification

For the classification of five different QUIC encrypted traffic, we formulated the problem as multi-class classification, with labels from 1 through 5, as shown in Table 2. For each traffic type, 100 instances are kept for testing and the remaining are used for training. For cross-validation, the test set is part of the training test.

To classify these five encrypted traffics, we deployed five ensemble techniques, namely: RF, ET, GBT, XGB, and LGBM. For each technique, the default parameters were used based on the findings of trial experiments on RF using a different number of estimators and other important parameters. It is worth noting that the classification accuracy also increased when the number of estimators was increased to 100. Therefore, for all models, we kept the default parameters. Two different types of experiment scenarios (strategies) were performed: the holdout method and the 10-fold cross-validation. In the holdout method, the data was separated into train and test sets. In the 10-fold cross-validation method, the data is partitioned into 10 subsets, and, in each fold, one subset is used for testing, and the remaining 9 are used for training. For all machine learning algorithms, the sci-kit learn library was used. QUIC traffic classification is an online application and delivering a quick decision on traffic type is critical for many applications. As such, consideration of a large number of packets requires a longer time to make traffic decisions, which is undesirable in a real-time scenario. Therefore, we investigated the QUIC traffic classification problem using 15 packets, and up to 120 packets, as inputs. More specifically, for all five techniques we used a different number of packets; 15, 30, 40, 50, 60, 80, 100, and 120, for the reason discussed in the preceding sentences and analyzed the classification results for better insights.

If we scan the scores given in Table 3, it can be observed that the highest accuracy achieved for 30 feature-inputs (i.e., for 15 packets used as input features) was 92.31% when the cross-validation was used. For the case of the holdout method, the highest accuracy was 91.60%. The highest accuracy for both scenarios was achieved by LGBM. Moreover, for both scenarios, the lowest scores were achieved by ET in terms of all metrics considered. It is also noteworthy that the performance scores of RF and ET, XGBT, and LGBM were close to each other in all evaluation metrics considered. The performance of GBT laid in-between these two groups. The accuracy scores of the cross-validation scenario are higher than those of the holdout scenario.

Table 3. Results on the test set with 15 packets as input.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	87.61	85.80	85.59	85.80	89.47	9.63
ET	86.67	85.40	85.24	85.40	86.43	3.59
GBT	88.30	87.40	87.37	87.40	89.96	58.01
XGBT	91.52	91.00	91.02	91.00	91.67	50.16
LGBM	91.94	91.60	91.61	**91.60**	**92.31**	22.7

From Table 4, it can easily be observed that the highest accuracy of 94.04% is achieved by LGBM when 30 packets, i.e., 60 features, are used as input. The same performance trend as in the previous discussion for all techniques is observed. It is worth mentioning that there is a significant improvement in accuracy for the cross-validation method in the case of XGBT and LGBM.

Table 4. Results on the test set with 30 packets as input.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	87.42	85.60	85.30	85.60	89.72	12.52
ET	85.39	84.20	84.00	84.20	86.45	6.53
GBT	88.70	87.80	87.76	87.80	90.22	117.76
XGBT	91.56	90.80	90.75	90.80	93.29	77.58
LGBM	92.15	91.40	91.34	**91.40**	**94.04**	29.70

The highest accuracy when using 40 packets as input is 93.80% for LGBM, as shown in Table 5. This result was achieved when the model is trained using 10-fold stratified cross-validation. The highest F1 score of 92.80 also is achieved by LGBM in the hold-out scenario. As in previous experiments, the lowest accuracy is achieved by ET when trained using the hold-out scenario.

Table 5. Results on the test set with 40 packets as input.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	86.92	85.20	84.95	85.20	89.70	13.94
ET	85.66	84.40	84.12	84.40	85.95	4.48
GBT	90.23	89.40	89.31	89.40	90.99	155.32
XGBT	93.07	92.20	92.17	92.20	93.20	105.12
LGBM	93.31	92.80	92.77	**92.80**	**93.80**	35.94

With 100 features (i.e., 50 packets are used as input), Table 6 shows that the highest scores of accuracy and F1 in the holdout scenario are 92.40 and 92.36, respectively, achieved by LGBM. For the cross-validation training scenario, the highest accuracy is 94.04%, and the lowest accuracy is 87.42% achieved, respectively, by LGBM and ET.

Table 6. Results on the test set with 50 packets as input.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	88.00	86.40	86.24	86.40	90.51	15.44
ET	85.52	84.60	84.36	84.60	87.42	5.09
GBT	89.02	88.20	88.12	88.20	91.21	199.23
XGBT	92.26	91.60	91.58	91.60	93.14	119.82
LGBM	92.79	92.40	92.36	**92.40**	**94.04**	41.13

The scores of all metrics for both training scenarios when using 60 packets' features as input are shown in Table 7. From the table, the highest and the lowest precision scores are 94.03 and 84.55 for LGBM and ET, respectively. The highest accuracy is 94.80% for the cross-validation scenario, which, again, is achieved by LGBM. It is emphasized that the performance scores of ET in the holdout scenario decreased in this experiment, though the cross-validation accuracy increased compared to previous experiments.

Table 7. Results on the test set with 60 packets as input.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	89.83	89.60	89.54	89.60	92.37	16.86
ET	84.55	84.40	84.22	84.40	88.86	7.06
GBT	90.92	90.60	90.55	90.60	92.41	239.78
XGBT	93.83	93.60	93.58	93.60	94.30	149.52
LGBM	94.03	93.80	93.77	**93.80**	**94.80**	49.70

The performance scores of all five techniques trained on both scenarios of holdout and cross-validation using 80 and 100 packets are shown in Tables 8 and 9, respectively. Significant improvement in performance in both RF and ET is observed, compared to previous experiments. It is also interesting to mention that the performance of XGBT and LGBM has almost no significant improvement in these sets of experiments.

Table 8. Results on the test set with 80 packets as input.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	91.06	90.60	90.57	90.60	93.11	18.35
ET	89.92	89.60	89.56	89.60	91.06	5.88
GBT	91.69	91.00	90.98	91.00	93.25	317.11
XGBT	93.32	92.80	92.79	92.80	94.80	174.21
LGBM	93.50	93.20	93.16	**93.20**	**95.05**	58.53

Table 9. Results on the test set with 100 packets as input.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	92.91	92.40	92.39	92.40	93.82	20.30
ET	92.00	91.60	91.54	91.60	91.17	7.00
GBT	92.19	91.20	91.23	91.20	93.53	393.74
XGBT	93.84	93.20	93.16	**93.20**	95.07	202.63
LGBM	93.66	93.20	**93.17**	**93.20**	**95.79**	70.22

Table 10 tabulates the performance scores of all five ensemble techniques when using 120 packets as input features. It is interesting to observe the significant improvement in all evaluation metrics for all techniques. This time, the highest accuracy of 99.40% is achieved by LGBM in the holdout scenario. It should also be emphasized that the performance scores of GBT, XGBT, and LGBM are the same for the holdout scenario, with an insignificant difference in the case of cross-validation. These experiments' lowest and highest F1-scores are 96.78 and 99.40, respectively. For overall visualization of accuracy scores for all experiments, please refer to Figure 5.

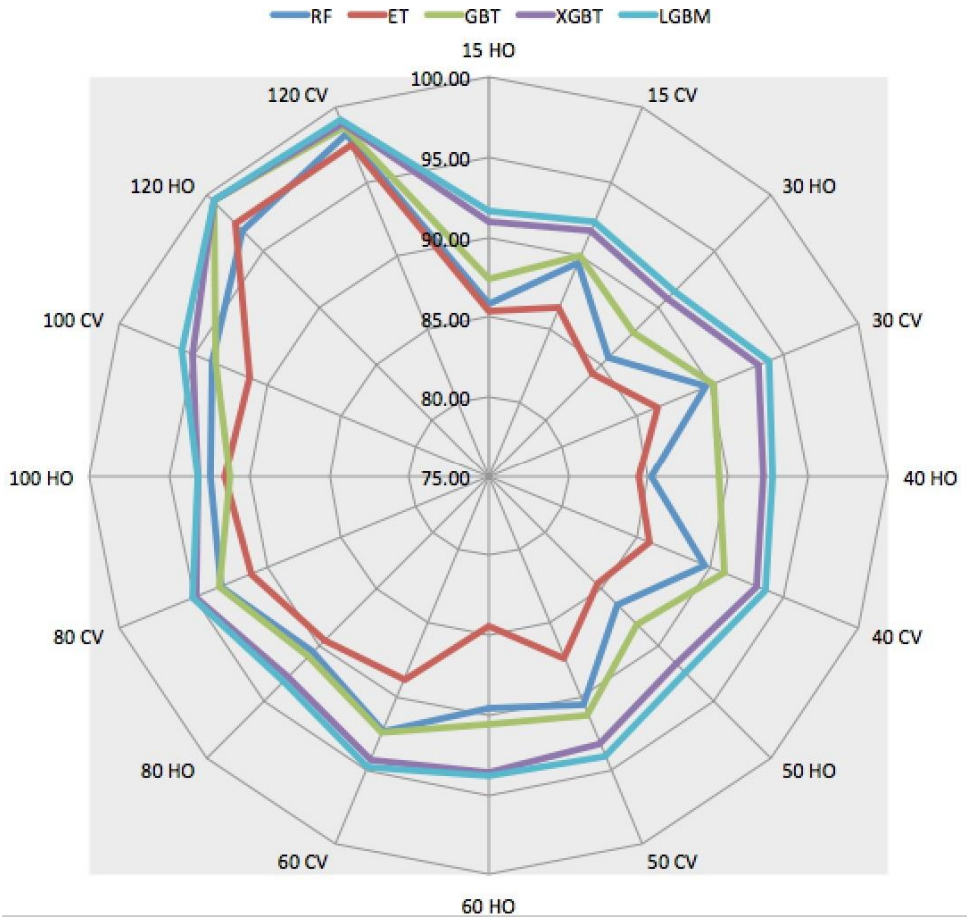

Figure 5. Performance sensitivity analysis of all models in both strategies.

Table 10. Results on test set 120 packets.

Model	Precision	Recall	F1-Score	Accuracy	10-Fold Cross-Validation Acc.	Training Time (s)
RF	96.93	96.80	96.78	96.80	98.24	14.67
ET	97.48	97.40	97.40	97.40	97.43	8.17
GBT	99.40	99.40	99.40	99.40	98.66	523.32
XGBT	99.40	99.40	**99.40**	**99.40**	98.92	156.47
LGBM	99.41	99.40	**99.40**	**99.40**	**99.12**	69.46

5.5. Time Cost and Model Size Comparison

The training cost of all ensemble models is captured and analyzed in addition to the trained model size. Please note that the training time recorded in Tables 3–10 is captured using a MacBook Pro 2015 model with 8 GB RAM on a Core i5. As a general trend, it is observed that increasing the number of packets lengthens the training process. Among the five ensemble models, ET takes the shortest raining time, whereas GBT takes the longest. It is interesting to note that LGM has the third shortest training duration after ET and RF with the highest performance. For the case of model size, GBT has the trained model with the smallest size of 582 KB and ET has the largest trained model size of 15.9 MB. The LGBM has a trained model size of 1.6 MB and XGBT has only 636 KB as its trained model size. Furthermore, the prediction is monitored, and it is discovered that the testing time is insignificant, with the RF model taking the shortest time of 0.0065 s and the ET ensemble model taking the longest time of 0.025. LGBM takes 0.017 s to test 100 input samples. We emphasize that training and testing times are affected by a variety of factors, including hardware configuration, number of input instances, and features. Nonetheless, this analysis provides an intuitive understanding of the training and testing time, as well as the size of ensemble models.

5.6. Comparison with Other Works

QUIC traffic classification results of our GBT, XGBT, and LGBM are compared with the other works available in the literature. For the comparison, to be more realistic and fairer, we chose the work that closely matched our objective of QUIC traffic classification and used the same dataset as we used. Additionally, we considered the papers that used accuracy as an evaluation metric to maintain the uniformity of the measurement. It should be emphasized that we did not re-run (replicate) their experiments gain, but the results were taken from each of their best-performing models. More details about their evaluation methods can be read in the respective papers shown in Table 11. From Table 11, it is observed that the classification accuracy of our work is either significant or closely comparable with that of other related works in the literature. In particular, from the confusion matrix of XGBT and LGBM, it is interesting to note that only three samples are misclassified from among five classes overall, where each class has 100 samples. In detail, out of each of the 100 testing samples from Google Drive and Google Music traffic, LGBM misclassified only 2 samples and one sample, respectively. In contrast, for all the other three traffics, classification accuracy is 100%. The same trend is observed for the case of XGBT, but with misclassification traffic in Google Drive and YouTube. The overall highest accuracy and F1-score are 99.40% each. One advantage of this approach is its simplicity since it only requires the packets captured from the traffic as input and does not require any additional feature extraction. The next section is rendered for performance sensitivity analysis of different ensemble techniques in light of the number of packets used and the training scenario used.

Table 11. Comparison between the proposed work and others in the literature.

Ref.	Year	Dataset	Algorithms	Main Objective	Score
[9]	2022	Orange'20, UC Davis QUIC [5]	CNN and Stacked LSTM	Encrypted web traffic classification	99.37% (Acc.)
[10]	2022	Orange'20, UC Davis QUIC [5], ISCX	ResNet34	Encrypted network traffic classification	99.24% (Acc.)
[8]	2020	UC Davis QUIC [5]	CNN	Network traffic classification	90.00% (Acc.)
Proposed work		UC Davis QUIC [5]	**RF, ET GBT, XGBT, LGBM**	Encrypted traffic classification	**99.40** (Acc.)

6. Discussion

In this section, the performance sensitivity of different techniques is analyzed based on the number of packets used for training each model. As such, we consider both the holdout training method and the cross-validation method. Please note that the sensitivity analysis is based on only the accuracy metric. The details are visualized in Figures 5–7.

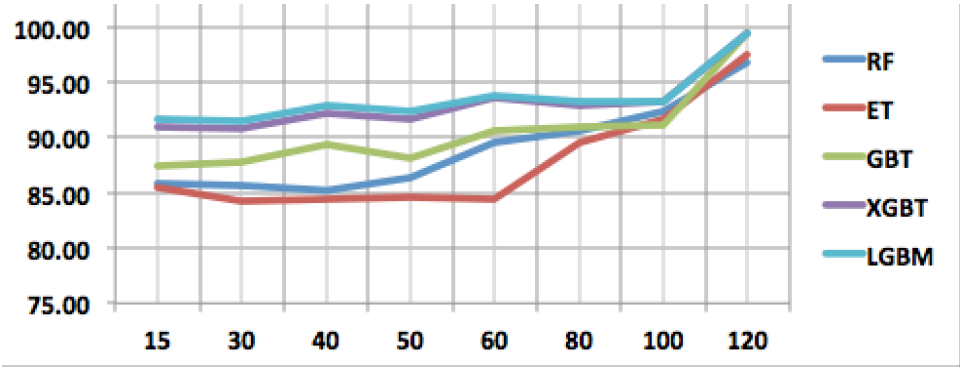

Figure 6. Performance sensitivity analysis of all models in holdout strategy.

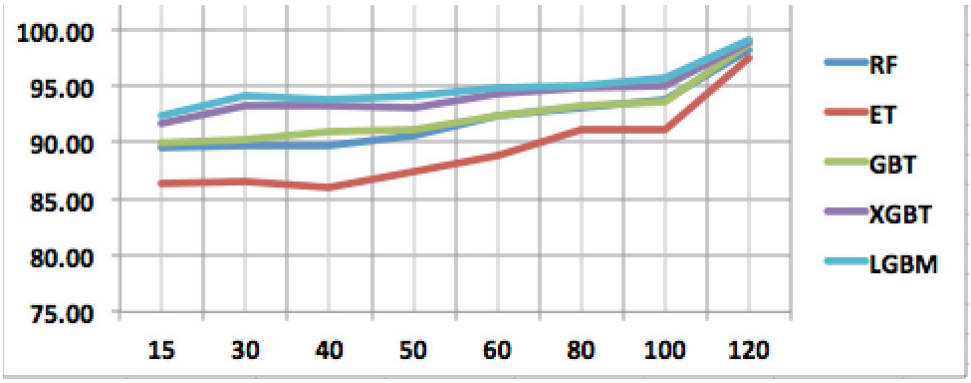

Figure 7. Performance sensitivity analysis of all models in cross-validation strategy.

If we first analyze the performance sensitivity of the RF algorithm in light of the number of packets used, it is observed that RF can classify QUIC traffic almost with the same accuracy for input packets 15, 30, and 40. This observation is about the holdout strategy and the same observation is also true for cross-validation scenarios. These trends can be visualized in Figures 6 and 7. When more than 50 packets are used as input features, the classification accuracy remarkably increases. In the case of ET, the classification trend is slightly different; for the holdout strategy, the classification accuracy almost remains stable

till 60 packets, and, after that significant increase is observed, and in the cross-validation scenario, classification accuracy increases with 40 packets afterward. GBT can classify QUIC traffic with varying accuracy depending on the number of packets. This behavior is seen up to 60 packets and, after that, classification accuracy increases as seen in Figure 6. However, when cross-validation is used for training, GBT shows trending classification accuracy as seen in Figure 7. Additionally, it should be noted that RF and GBT are going shoulder-to-shoulder in the cross-validation strategy, which shows the robustness of the cross-validation training strategy in QUIC traffic classification.

For XGBT and LGBM, in both holdout and cross-validation scenarios, the scores of accuracies are very close and show similar trending as seen in Figures 6 and 7. As seen in these two figures, these techniques are less sensitive to input packets (features) compared to other techniques for both training strategies. More specifically, the performance sensitivity is less, concerning the number of input features for XGBT and LGBM in the case of cross-validation scenarios.

One of the reasons that XGBT and LGBM outperform the others is the fact that they are built as a boosting technique. While bagging and stacking try to enhance the predictions by fitting multiple models at the same time without a difference between the models, the boosting technique involves fitting the models sequentially such that the next model will be more enhanced than the previous model. Therefore, the final output is more robust since it was tuned by multiple previous models. For that reason, we can see that GBT, which is a boosting technique as well, outperforms both RF and ET in all the experiments. The second reason is that both XGBT and LGBM are designed in a more regularized way to control over-fitting. In more specific terms, XGBT uses the elastic regularization technique, i.e., it uses both L1 and L2 regularization, whereas LGBM uses Gradient-Based One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB) for performance boosting. For that purpose, we can see that both perform better than GBT. These reasons also are contributing points to the situation that the performance remains robust and comparatively higher than other competing algorithms, even if a relatively less number of packets (input features) are used. This can be observed in Figure 5.

As an overall impression, among five ensemble techniques experimented with for QUIC traffic classification in this work, LGBM and XGBT show promising performance in terms of all metrics. In extreme feature requirement situations, classification accuracy lies above 99%, and, in less-feature requirement situations, accuracy is above an acceptable range such as 91%. Moreover, they are less sensitive to feature requirements and faster than all other algorithms. Therefore, it is safe to mention that LGBM and XGBT are the right choices for deploying a real-world online application for multi-class QUIC traffic classification because they can be trained using a small number of features as 15 packets.

7. Conclusions

In this paper, five different ensemble machine learning models were investigated for the QUIC traffic classification, namely, Random Forest, Extra Trees, Gradient Boosting Tree, Extreme Gradient Boosting Tree, and Light Gradient Boosting Model. The models were trained using 15, 30, 40, 50, 60, 80, 100, and 120 packets as features on two different training and testing scenarios, i.e., holdout, and 10-fold cross-validation. From the experimental results and analysis, it was found that the LGBM outperforms the others achieving more than 99% in terms of accuracy, precision, recall, and F-1 score. In addition to that, it was found that LGBM and XGBT can be trained using a small number of features such as 15 packets and still achieve a performance score of about 92%. The models were compared to other state-of-the-art models found in the literature, and achieved the highest results, using only the traffic packets as a feature. These results indicate that ensemble learning techniques, especially LGBM and XGBT, can reveal the service from generated traffic and be a possible choice for deploying in real-world online application.

For future work, a replication study might be conducted on a larger dataset with more QUIC services. In addition, other models could be tested and compared to the proposed

models in this research. Finally, a prediction tool based on the proposed models could be produced for the industry. Moreover, automatic hyperparameter optimization can be applied to further investigate the performance of ensemble models.

Author Contributions: Conceptualization, S.A., M.A. and A.A.; methodology, M.A. and A.A.; software, M.A. and A.A.; validation, S.A., A.A. and M.A.; formal analysis, S.A., M.A. and A.A.; investigation, S.A.; resources, S.A., M.A. and A.A.; data curation, M.A. and A.A.; writing, S.A, M.A. and A.A.; writing, review and editing, S.A., A.A. and M.A.; visualization, S.A., M.A. and A.A.; supervision, S.A.; project administration, S.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data used in this paper is available at [5].

Acknowledgments: The authors would like to acknowledge King Fahd University for supporting this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kumar, P.; Dezfouli, B. Implementation and analysis of QUIC FOR MQTT. *Comput. Netw.* **2019**, *150*, 28–45. [CrossRef]
2. Erman, J.; Gopalakrishnan, V.; Jana, R.; Ramakrishnan, K.K. Towards a spdy'ier mobile web? *IEEE/ACM Trans. Netw.* **2015**, *23*, 2010–2023. [CrossRef]
3. Langley, A.; Riddoch, A.; Wilk, A.; Vicente, A.; Krasic, C.; Zhang, D.; Yang, F.; Kouranov, F.; Swett, I.; Iyengar, J.; 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, Los Angeles, CA, USA, 21–25 August 2017; pp. 183–196.
4. Al-Bakhat, L.; Almuhammadi, S. Intrusion detection on Quic Traffic: A machine learning approach. In Proceedings of the 2022 7th International Conference on Data Science and Machine Learning Applications (CDMA), Riyadh, Saudi Arabia, 1–3 March 2022. [CrossRef]
5. Rezaei, S.; Liu, X. How to achieve high classification accuracy with just a few labels: A semi-supervised approach using sampled packets. *arXiv* **2020**, arXiv:1812.09761v2.
6. Sandvine. Global Internet Phenomena Report. 2022. Available online: https://www.sandvine.com/global-internet-phenomena-report-2022 (accessed on 20 February 2023).
7. Secchi, R.; Cassara, P.; Gotta, A. Exploring machine learning for classification of QUIC flows over satellite. In Proceedings of the ICC 2022-IEEE International Conference on Communications, Seoul, Republic of Korea, 16–20 May 2022. [CrossRef]
8. Rezaei, S.; Liu, X. Multitask Learning for Network Traffic Classification. In Proceedings of the 2020 29th International Conference on Computer Communications and Networks (ICCCN), Honolulu, HI, USA, 3–6 August 2020. [CrossRef]
9. Akbari, I.; Salahuddin, M.A.; Ven, L.; Limam, N.; Boutaba, R.; Mathieu, B.; Moteau, S.; Tuffin, S. Traffic classification in an increasingly encrypted web. *Commun. ACM* **2022**, *65*, 75–83. [CrossRef]
10. Towhid, M.S.; Shahriar, N. Encrypted network traffic classification using self-supervised learning. In Proceedings of the 2022 IEEE 8th International Conference on Network Softwarization (NetSoft), Milan, Italy, 27 June–1 July 2022. [CrossRef]
11. Rezaei, S.; Liu, X. Deep learning for encrypted traffic classification: An overview. *IEEE Commun. Mag.* **2019**, *57*, 76–81. [CrossRef]
12. Iyengar, J.; Thomson, M. QUIC: A UDP-Based Multiplexed and Secure Transport. In *RFC 9000*. 2021. Available online: https://datatracker.ietf.org/doc/rfc9000/ (accessed on 20 February 2023).
13. Tong, V.; Tran, H.A.; Souihi, S.; Mellouk, A. A novel quic traffic classifier based on convolutional neural networks. In Proceedings of the 2018 IEEE Global Communications Conference (GLOBECOM), Abu Dhabi, United Arab Emirates, 9–13 December 2018. [CrossRef]
14. Lotfollahi, M.; Jafari Siavoshani, M.; Shirali Hossein Zade, R.; Saberian, M. Deep packet: A novel approach for encrypted traffic 516classification using Deep Learning. *Soft Comput.* **2019**, *24*, 1999–2012. [CrossRef]
15. Williams, N.; Zander, S.; Armitage, G. A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification. *ACM SIGCOMM Comput. Commun. Rev.* **2006**, *36*, 5–16. [CrossRef]
16. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A.; Lloret, J. Network traffic classifier with convolutional and recurrent neural networks for internet of things. *IEEE Access* **2017**, *5*, 18042–18050. [CrossRef]
17. Izadi, S.; Ahmadi, M.; Nikbazm, R. Network traffic classification using convolutional neural network and ant-lion optimization. *Comput. Electr. Eng.* **2022**, *101*, 108024. [CrossRef]
18. Izadi, S.; Ahmadi, M.; Rajabzadeh, A. Network traffic classification using Deep Learning Networks and bayesian data fusion. *J. Netw. Syst. Manag.* **2022**, *30*, 25. [CrossRef]
19. Sun, W.; Zhang, Y.; Li, J.; Sun, C.; Zhang, S. A deep learning-based encrypted VPN traffic classification method using packet block image. *Electronics* **2022**, *12*, 115. [CrossRef]

20. Liu, W.; Zhu, C.; Ding, Z.; Zhang, H.; Liu, Q. Multiclass imbalanced and Concept Drift Network traffic classification framework based on online active learning. *Eng. Appl. Artif. Intell.* **2022**, *117*, 105607. [CrossRef]
21. Bühlmann, P. Bagging, boosting and ensemble methods. In *Handbook of Computational Statistics*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 985–1022.
22. Zhang, C.; Ma, Y. *Ensemble Machine Learning: Methods and Applications*; Springer: Berlin/Heidelberg, Germany, 2012.
23. Breiman, L. Random forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
24. Schapire, R.E. The boosting approach to machine learning: An overview. In *Nonlinear Estimation and Classification*; Springer: New York, NY, USA, 2003; pp. 149–171.
25. Chen, J.; Huang, H.; Cohn, A.G.; Zhang, D.; Zhou, M. Machine learning-based classification of rock discontinuity trace: SMOTE oversampling integrated with GBT ensemble learning. *Int. J. Min. Sci. Technol.* **2022**, *32*, 309–322. [CrossRef]
26. Chen, T.; He, T. Higgs boson discovery with boosted trees. In Proceedings of the NIPS 2014 Workshop on High-Energy Physics and Machine Learning, Montreal, QC, Canada, 13 December 2014; pp. 69–80.
27. Chen, T.; Guestrin, C. XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'16), San Francisco, CA, USA, 13–17 August 2016; Volume 785, p. 794.
28. Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; Liu, T.Y. Lightgbm: A highly efficient gradient boosting decision tree. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 3149–3157.