IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 25, 2021, accepted September 12, 2021, date of publication September 16, 2021,
date of current version September 28, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3113294

Feature Engineering and Machine Learning Model Comparison for Malicious Activity Detection in the DNS-Over-HTTPS Protocol

MATTHEW BEHNKE[1], NATHAN BRINER[1], DRAKE CULLEN[1], KATELYNN SCHWERDTFEGER[1], JACKSON WARREN[1], RAM BASNET[1], AND TENZIN DOLECK[2]

[1]Department of Computer Science and Engineering, Colorado Mesa University (CMU), Grand Junction, CO 81501, USA
[2]Simon Fraser University, Burnaby, BC V5A 1S6, Canada

Corresponding author: Ram Basnet (rbasnet@coloradomesa.edu)

This work was supported by the state of Colorado through funds appropriated for cybersecurity law dubbed "Cyber Coding Cryptology for State Records."

ABSTRACT The Domain Name System (DNS) is among the most ubiquitous and important protocols for network communication; however, security concerns regarding DNS have been on the rise and demand for encrypted traffic has followed suit. Using a publicly available dataset, this work compares 10 different machine learning classifiers using stratified 10-fold cross-validation. The classifiers are used to determine the most effective and efficient way of detecting malicious DNS over Hypertext Transfer Protocol Secure (HTTPS) traffic, dubbed DoH traffic. Model performance is evaluated on Non-DoH vs. DoH traffic, then tested on benign vs. malicious DoH traffic. Additionally, this paper seeks to build upon existing research by removing noise and introducing feature selection methods and feature explainability to produce a better model for real-world deployment. After eliminating five overfitting features, our findings indicate that light gradient boosting machine (LGBM) yielded the highest accuracy to training time ratio while approaching 0% error using 20 top features.

INDEX TERMS Chi-squared, DNS, DoH, decision tree, LGBM, machine learning, pearson correlation, random forest, sequential forward selection, XGBM.

I. INTRODUCTION

The Domain Name System (DNS) is a vital component of the modern internet. DNS improves user experience by translating human-readable names into Internet Protocol (IP) addresses which are necessary for accessing websites and domains. In other words, one does not need to memorize IP addresses to visit a website. Additionally, DNS provides a hierarchical system for hosts which prevents name collisions and whose traffic gives valuable information for network administrators [1].

To understand the DNS protocol, follow along with Figure 1. Say a user looks up the website www.site.com. The user's request is sent to the Local DNS Resolver (usually managed by an Internet Service Provider). The Local Resolver either responds with the corresponding IP address (if that address is cached) or it will forward the query to the Public Resolver [2]. Similarly, the Public Resolver either returns the associated address or queries the Root Server [2].

The Root Server will not reply with the corresponding address—instead, it will direct the DNS Resolver to the Top-Level Domain (TLD) Server for the.com domain (in this example). Once again, the TLD Server will not know where to find www.site.com. Therefore, it will respond with the address of the corresponding Authoritative Name Server [3]. Finally, the Authoritative Name Server returns the IP address for www.site.com [4]. The client is now able to access the requested website.

Although DNS is a critical protocol, it was not designed with security and privacy in mind. DNS is vulnerable to amplification attacks, DNS cache poisoning, botnet attacks, phishing attacks, and DNS manipulation [5]. DNS-over-Encryption protocols such as DNS-over-HTTPS (DoH) and DNS-over-TLS (DoT) were introduced to address these security concerns [2]. Internet users have rapidly adopted these encryption protocols which have become a lynchpin in protecting user's privacy.

The associate editor coordinating the review of this manuscript and approving it for publication was Jiafeng Xie.

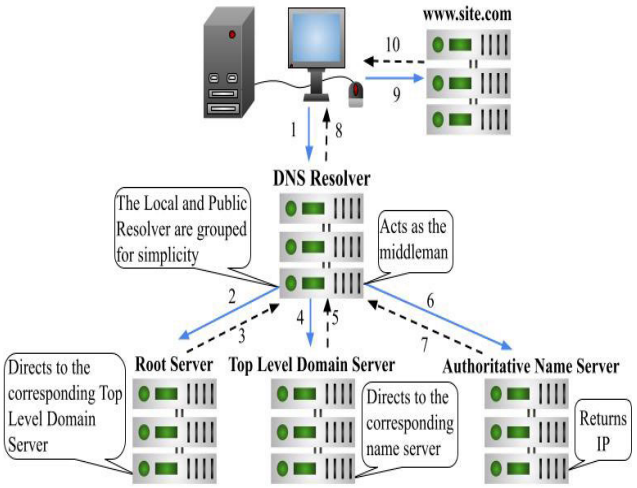

FIGURE 1. The DNS protocol.

DoT encrypts DNS traffic through the Transfer Layer Security (TLS) protocol. As seen in Figure 2, before any DNS lookups occur, a TLS session is established between the server and the client on port 853 [6]. To establish the session, the client queries the server and receives a certificate. Next, the certificate authority authenticates the certificate sent by the server. If all goes well, the client and server share encryption keys [3]. Once the TLS session is active, DNS queries and subsequent responses are sent over the encrypted channel, making it difficult for a third party to view the contents of the queries. Unlike DNS, DoT mitigates man-in-the-middle attacks [7].

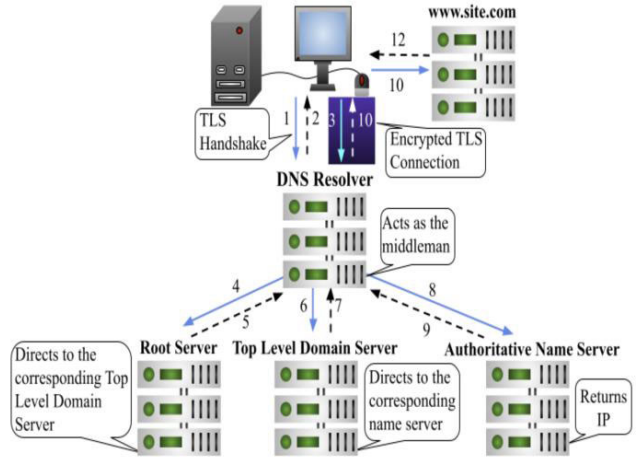

FIGURE 2. The DoT protocol.

Since port 853 is dedicated exclusively to DoT traffic, it is easy for Network Administrators to monitor and block malicious traffic. Should malicious traffic be detected, one could simply Block port 853, effectively restricting the use of DoT [2]. Developed separately in 2018, DoH serves as an alternative to DoT. Unlike DoT, DoH utilizes the HTTPS protocol on port 443. DoH queries coexist among other HTTPS connections, therefore port 443 should not be blocked [3]. Depending on the situation, the increased privacy that DoH provides can be seen as advantageous or disadvantageous because its traffic is less identifiable. For instance, a privacy concerned user would be in favor of the more anonymized traffic; however, it can be harder for network administrators to monitor and block DoH traffic as it is mixed in with HTTPS traffic. The DoH protocol can diminish network security since network administrators can no longer utilize DNS filtering or monitoring as easily as they could with DoT. These techniques provide significant data, including response IP address, originating IP address, and query type. DoH has seen more adoption than DoT. As of now, there are 17 DoH providers, notably Google and Cloudflare [7].

To establish an encrypted connection with a DoH server, the client sends a DNS request to resolve the Uniform Resource Identifier (URI) template and get the IP address of the server [8]. The client can now communicate directly with the DoH server by using HTTPS GET or POST requests [9]. At this point, the rest of the steps are the same as in Figure 1 (the DNS Model).

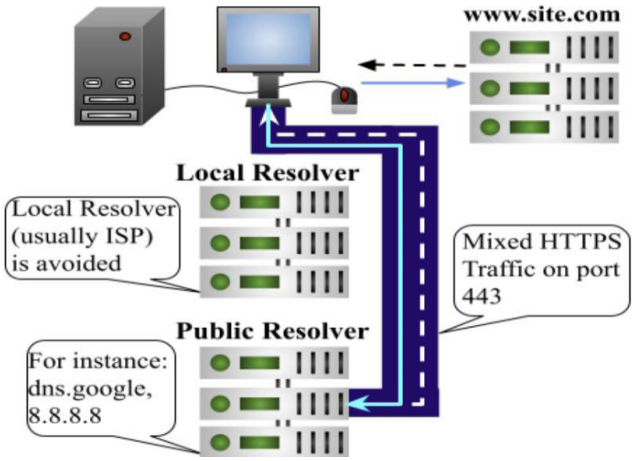

FIGURE 3. The DoH protocol.

Currently, protective DNS services filter malicious addresses/domains and provide defense against typosquatting (directing a user to a malicious website through common inadvertent typos in their URL search). Ideally, DoH connects a client to enterprise resolvers which can send queries through the same DNS protections; however, should the DoH resolvers be external, the DNS services are bypassed and many of these protections are lost [10]. There are also potential security leaks when considering external DoH. With an external DoH request, the DNS server will be forgone, and the DoH request goes straight to the external DoH resolver. This becomes problematic when users are trying to access an internal domain. The initial external request will fall over to an internal DNS resolver, where the user's information is now susceptible to attack. In addition, the user may find the site's services inaccessible due to the fact they are an external source [10].

Upstream DNS traffic becomes a security problem when using DoH. While DoH secures the connection between the client and the DoH server, it does not protect the user's DNS traffic between DoH servers and top-level DNS servers. When reaching this level, DoH is not used and traffic is not encrypted. During this time, cyber threats could passively view plaintext DNS or redirect the traffic to a malicious site [10]. Moreover, a DoH provider can store a client's data [3]. In DoH downgrade attacks, nefarious individuals work to switch a DoH connection to a DNS connection. Once a DoH connection is downgraded to DNS, man-in-the-middle attacks, cache poisoning, DNS hijacking, and many other attacks can be performed [8]. Due to these security concerns, it is advantageous to detect malicious DoH traffic.

For this paper, our research will be focused on DoH in an attempt to critique and improve Banadaki's [11] research (see Related Works section). By removing noise, introducing feature selection methods, and providing explainability to the features that characterize the dataset, we work to detect malicious DoH traffic more accurately and efficiently.

The rest of the manuscript is organized in the following manner. Related works are analyzed in section II, and a brief overview of the experiment and dataset are provided in part III. In section IV, feature selection is introduced along with results from the Chi-Square and Pearson Correlation Coefficient tests. Four machine learning classifiers are explained in section V, followed by experiments and results in part VI. Next, sequential forward selection results are presented in section VII. Finally, discussions and limitations are explored in section VIII, and the manuscript concludes in section IX.

TABLE 1. Acronym reference.

Acronym	Description	Acronym	Description
DNS	Domain Name System	EFB	Exclusive Feature Bundling
HTTPS	Hypertext Transfer Protocol Secure	XGBoost	eXtreme Gradient Boosting
TLS	Transport Layer Security	GBM	Gradient Boosting Machine
DoH	DNS over HTTPS	DT	Decision Tree
DoT	DNS over TLS	LDA	Linear Discriminant Analysis
LGBM	Light Gradient Boosting Machine	KNN	K-Nearest Neighbors
IP	Internet Protocol	GNB	Gaussian Naive Bayes
TLD	Top Level Domain	RF	Random Forest
URI	Uniform Resource Identifier	AB	AdaBoost Classifier
SIE	Security Information Exchange	GB	Gradient Boosting Classifier
ReLU	Rectified Linear Unit	XGB	XGBoost
CIC	Canadian Institute for Cyber Security	ET	Extra Trees
PCC	Pearson Correlation Coefficient	SFS	Sequential Forward Selection
FDG	Feature Distribution Graph	ANN	Artificial Neural Network
GOSS	Gradient One-Side Sampling	TCP/IP	Transmission Control Protocol/Internet Protocol

II. RELATED WORKS

A. DNS

The Massachusetts Institute of Technology Laboratory for Computer Science conducted research to find new solutions to increase DNS abilities, primarily to solve many DNS shortcomings using the new technology DDNS [12]. DDNS is a peer-to-peer hash table on top of the system Chord. The system sought to eliminate tedious server administration by inheriting load balancing and fault tolerance from the peer-to-peer layer. The researchers [12] discussed the benefits of using DDNS over normal methods of DNS, as well as the disadvantages of using the alternative system. Ultimately, the researchers concluded that the system's latency rates were too high to be a worthy investment to improve DNS. In addition, response times were much higher, having a median response time of 350ms compared to the 43ms for standard DNS. The team further reviewed other areas that DDNS struggled in when compared to normal DNS. Finally, the team addressed the areas of potential growth for DDNS.

Bilge *et al.* [13] presented EXPOSURE, a system that passively views DNS activity for domains that can be involved in malicious activity. Their study included 15 features for analysis that pertain to DNS queries. The research consisted of a dataset of 100 billion real-world DNS requests, which were collected over two months consisting of DNS traffic acquired from the Security Information Exchange (SIE). The system was trained with a two-layer detection method: the first layer separated benign and malicious domains, then the second layer determined and detected suspicious domains. Evaluation of the system, including detection rates, misclassified rates, and domain detection times are included. They found their classifier had a 98% detection rate using their training set. To ensure that this translated to unseen data sets, the team went further using mal-wareurls.com to present EXPOSURE to a new dataset. Their classifier achieved a 98% accuracy in this dataset. The team incorporated the implementation of the system into new scenarios through an ISP show, proving its scalability and proof of concept.

B. DoH

Huang *et al.* [8] performed DNS-over-HTTPS downgrade attacks on six different web browsers using four attack vectors. In DoH downgrade attacks, nefarious individuals work to switch a DoH connection to a DNS connection. Once a DoH connection is downgraded to DNS, all vulnerabilities associated with DNS are reintroduced. Both phases of the DoH protocol are susceptible to DoH downgrade attacks. Phase 1 occurs when the client is initially connecting to a DoH server. Phase 2 deals with the TLS connection and HTTPS GET and POST requests sent between the client and server. The four different attack vectors that the researchers used are DNS Traffic Interception, DNS Cache Poisoning, TCP Traffic Interception, and TCP Reset Injection. All four attack methods were viable on all six browsers (Google Chrome, Mozilla Firefox, Opera, Microsoft Edge, Brave and

Vivaldi) to downgrade DoH to DNS. To address the shortcomings, the team suggested that browsers should notify a user when DoH is disconnected. Furthermore, they suggested a need for a new protocol or method to address DoH security flaws. Our research attempts to address their concern by proposing a method to detect malicious DoH traffic.

Konopa *et al.* [3] used various machine learning and deep learning approaches for automated DoH detection. The team explained how DNS helps in converting domain names to an IP address by using HTTPS tunneling. The team also briefly discussed how it is difficult to maintain anonymity by using DNS without the HTTPS protocol. Many browsers e.g., Firefox and Chrome have adopted DoH to prevent this issue. They use an L3 switch called IPFOX which collected all the traffic information using the NETFLOW protocol. Various optimization processes such as Adam stochastic and Adam optimizer were used in the experiment. They implemented their model in the Keras/TensorFlow environment. The Rectified Linear Unit (ReLU) activation function was used with the first three layers and the last layer used the sigmoid function. The team produced an accuracy of approximately 80% with non-normalized data and up to 95% with cleaned and normalized data.

Nijeboer [14] uncovered a technique to filter DoH queries from other HTTPS traffic using packet size-related features and discussed why DoT is not as popular as DoH. The team used Alexa to generate the top 50 popular sites based on the average daily time spent on the website. They used a simple algorithm to identify DoH traffic. The algorithm correctly identified DoH requests to four servers (Cloudflare, NextDNS, Google, and Knot Resolver) with an accuracy of 97.87%.

Hjelm [2] proposed a system for organizations to detect and restrict encrypted DNS traffic (particularly DoH traffic). Zeek logs can be used alongside SSL and HTTP logs to determine if a client has connected to a DoH provider. Zeek can also be used to determine if a client on your internet is connected to a command-and-control server. Rita can be used alongside Zeek to look for patterns in DoH traffic and identify malicious traffic such as beaconing. Furthermore, organizations can restrict DoH traffic by blocking connections to known DoH resolvers. With appropriate analysis and controls, a company can avoid the concerns associated with DoH traffic.

C. DETECTING MALICIOUS DoH TRAFFIC

Banadaki [11] researched a two-layer approach regarding malicious DoH traffic detection. Layer one focused on differentiating Non-DoH traffic from DoH traffic, while layer two worked to distinguish Malicious-DoH traffic from Benign-DoH traffic. Banadaki utilized IBM's Auto AI for model selection, data preparation, feature engineering, and hyperparameter optimization. Auto AI cleans the data set by eliminating, scaling, and encoding features. Next, Auto AI performs an automated feature selection process. Finally, Auto AI performs hyperparameter optimization and ranks models based on how well they perform on a dataset [15]. Evaluation of the six classification algorithms was done using accuracy, precision, recall, F-Score, confusion matrices, ROC Curves, and feature importance. The findings documented that the LGBM and XGBoost algorithms were superior, reaching nearly 100% accuracy for both layers. Key features included: SourceIP, DestinationIP, SourcePort, DestinationPort, and TimeStamps. Other possible important features that could lead to high classification are introduced as well. The research conducted by Banadaki will be the focus of this paper and is discussed in greater detail in the following section.

TABLE 2. List of 34 features extracted from captured traffic.

#	Feature
1	SourceIP
2	DestinationIP
3	SourcePort
4	DestinationPort
5	TimeStamp
6	Duration
7	FlowBytesSent
8	FlowSentRate
9	FlowBytesReceived
10	FlowReceivedRate
11	PacketLengthVariance
12	PacketLengthStandardDeviation
13	PacketLengthMean
14	PacketLengthMedian
15	PacketLengthMode
16	PacketLengthSkewFromMedian
17	PacketLengthSkewFromMode
18	PacketLengthCoefficientofVariation
19	PacketTimeVariance
20	PacketTimeStandardDeviation
21	PacketTimeMean
22	PacketTimeMedian
23	PacketTimeMode
24	PacketTimeSkewFromMedian
25	PacketTimeSkewFromMode
26	PacketTimeCoefficientofVariation
27	ResponseTimeTimeVariance
28	ResponseTimeTimeStandardDeviation
29	ResponseTimeTimeMean
30	ResponseTimeTimeMedian
31	ResponseTimeTimeMode
32	ResponseTimeTimeSkewFromMedian
33	ResponseTimeTimeSkewFromMode
34	ResponseTimeTimeCoefficientofVariation

D. DEEP LEARNING FOR DNS TRAFFIC MONITORING

Deep learning has proven itself invaluable for the future of DNS traffic detection and categorization. While this paper seeks only to compare and evaluate traditional machine learning algorithms, it is worth mentioning some experiments that have utilized deep learning.

Jiang *et al.* [16] introduced an online detection scheme combining a convolutional neural network with neural language processing techniques. The system utilizes

character-level embedding to map URLs to vectors allowing for the extraction of hidden features. Their character-level convolutional neural network misclassified 40 URLs per 1000 analyzed outperforming a baseline feature-based model by a significant margin. Moreover, it's time for classification was on par with the baseline methods, proving its efficiency.

Palau *et al.* [17] trained a neural network for detecting domain generation algorithms (DGA) and tunneling DNS threats. In other related research, these two threats are generally detected through disparate models; however, through the use of a Multi-Class Convolutional Neural Network, they consolidated these models into one. Their model accurately detected normal domains 99% of the time, DGA with 97% accuracy, and tunneling with 92% accuracy.

E. COMPARISONS

Refer to Table 3 for comparisons of DoH traffic detection research.

III. EXPERIMENT OVERVIEW

A. GOAL AND MOTIVATION

In the study conducted by Banadaki [11], the primary research goal was to identify DoH traffic and distinguish malicious traffic from benign traffic using a two-layer approach. The findings documented that LGBM and XGBoost algorithms performed near flawlessly, yielding close to 100% classification accuracy on a test dataset of 4,000 samples. Banadaki [11] found that when comparing the importance of the 34 features provided by the CIRA-CIC-DoHBrw-2020 dataset, SourceIP and DestinationIP proved to be the most useful.

While the research conducted by Bandaki [11] resulted in incredibly accurate classifications, there were a few tweaks our team implemented that could improve the robustness of the model. When looking deeper into Banadaki's [11] methodology, our team found limitations. In reviewing the important features that made such a high precision rate possible (SourceIP and DestinationIP), we concluded that there was a lack of diversity in the dataset's IP addresses. In total, there were less than 30 IP addresses used for the experiment: 12 DestinationIP and 14 SourceIP. A dataset of 4,000 queries ideally would have very few duplicates to simulate a real-world environment where there are many IP addresses prone to attack at any given time. With a lack of diversity in this area, the machine learning algorithms, LGBM and XGBoost, categorized traffic based on IP addresses and ports, which could have led to the incredibly high accuracy, resulting in overfitting.

To improve on the methods used by Banadaki [11], our team re-created the experiment using the same two-layer model and a dataset containing 34 features. Layer 1's primary goal was to separate DoH from Non-DoH, while Layer 2's was to classify samples as Malicious DoH or Benign DoH. To solve the problem of overfitting features, we eliminated any features that were not significant for the purpose of the study or could create unintended bias. Data returning with NaN values were removed as well. Features including Destination IP, Source IP, Destination Ports, Source Ports, as well as timestamps were all removed from the study. Destination IP and Source IP were removed to ensure that the machine learning algorithms did not recognize the IP as malicious or benign instead of evaluating the data for being

TABLE 3. Comparisons of DNS traffic detection research.

Paper	Deep Learning	Dataset	Classifiers	Results
Bilge et al. [13]	No	DNS traffic recorded from security information exchange (SIE)	J48 decision tree (c4.5 J48)	~98% accurate (15 features)
Konopa [3]	No	Generated utilizing virtual machines and NetFlow	K-Nearest Neighbours (5-NN), C4.5 Decision Tree, Random Forest, Naive Bayes, and Ada-boosted Decision Tree	>94% accuracy
Banadaki [11]	No	CIRA-CIC-DoHBrw -2020	Decision Tree, Extremely randomized Trees (Extra Trees), Gradient Boosting, XGBoost (XGB), LGBM, and Random Forest	~99.8% accuracy (34 features)
Jiang et al. [15]	Yes	Data compiled from Online public datasets and crawled from malicious URL websites	Character-level Convolutional Neural Network	~96% accuracy
Palau et al. [17]	Yes	DNS threats dataset containing normal, DGA, and tunneling traffic	Multi-class Convolutional Neural Network	Benign: ~99% DCA: ~97% Tunneling ~92%
This Paper	No	CIRA-CIC-DoHBrw -2020	Decision Tree, Random Forest, LGBM, XGBoost, Linear Discriminant Analysis, K-Nearest Neighbors, Gaussian Naive Bayes, AdaBoost Classifier, Gradient Boosting Classifier, and Extra Trees	>99% accuracy (29 features)

malicious/benign. Eliminating these identifiers as variables increases the broader applicability of the model since the models would not have the advantage of knowing which IP is malicious or benign when being utilized in real-world situations. Both Destination Ports and Source Ports were removed with similar reasoning. Non-DoH and DoH samples for layer 1 were provided separately and presented one after the other, allowing the algorithms to utilize the difference in Timestamps to differentiate the data. In order to avoid recognition based on Timestamps, we removed this feature as well.

B. DATASET

The dataset for our research was gathered from the project CIRA-CIC-DoHBrw-2020 provided by the Canadian Institute for Cybersecurity (CIC) under the Canadian Internet Research Authority [18]. The data was generated by running chosen destination IP addresses through DoH tunnels set by source IPs for Google Chrome and Mozilla Firefox. The clients were run simultaneously on ten servers that were all connected to a single C2 server. To collect and analyze this traffic flow they developed DoHLyzer in Python using Scapy to read pcap files or sniff packets online. The resulting data was a collection of packets, flows, and DoH types for each case. Finally, using a DoHMeter developed in Python, they were able to extract the statistical and time-series features that are used to classify and group the dataset for inspection [11]. The dataset is composed of 34 features and are listed in Table 2.

IV. FEATURE RANKING

Overfitting is a problem that occurs when a model tailors itself too closely to its training data. If a model is overfit, it will function poorly when introduced to new data. For instance, the model may have high overall accuracy, but misclassify classes when used on real world data. Our dataset contains 34 features, though it is likely that not all of them are pertinent for our goal of DoH traffic detection and categorization. Unnecessary features can lead to an overfit model, rendering it unusable in real word application. To address the overfitting problem, we performed two different methods for feature selection: Chi-Square test and Pearson Correlation Coefficient (PCC) test.

A. CHI-SQUARED RESULTS

After eliminating the overfitted features, the Chi-Square test was employed to ensure that the remaining 29 features are statistically significant. This statistical test is used on categorical data to determine the likelihood that an observed difference emerged between features [19]. The first step in using the chi-squared test is to calculate the chi-square statistic which can be calculated with the following formula [20].

$$\chi^2 = \sum_{i=1}^{R} \sum_{j=1}^{K} \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \quad (1)$$

where,

- E_{ij} = an expected value
- O_{ij} = an observed value
- $R \times K$ = the total number of outcomes

This test provided us with a p-value for each feature based on its significance in the dataset. A p-value threshold of below 0.05 is commonly used to reject the null hypothesis, which implies that the feature and target variable have a relationship. Any value above the threshold of 0.05 is considered statistically insignificant. Features and their respective p-values are displayed in Table 4. P-values that are considered statistically insignificant are highlighted in red and were removed from their respective layers for training machine learning modes.

Table 4 shows chi-square test p-values for each feature in Layer 1 and Layer 2 data. Features are highlighted in red if $p >= 0.05$, denoting statistical insignificance.

B. PEARSON CORRELATION

To validate our results from the Chi-Square test, we conducted a Pearson Correlation Coefficient (PCC) test. The PCC test measures the relationship between a variable and a target. If the variable and the target have a strong positive correlation, the correlation coefficient has a value close to $+1$. Alternatively, if the variable and target are negatively correlated, the correlation coefficient is close to the value -1. The correlation coefficient is close to 0 if the correlation is weak [21]. We settled on the absolute value of all the correlation coefficients since we were looking for any correlation. As seen in Figure 4 and 5, the results from the PCC test aligned with the results from the Chi-Square test.

C. FEATURE DISTRIBUTION

Feature Distribution Graphs (FDGs) model feature variables allowing for better analysis of the data relevance. The graphs in Figure 6 and 7 show two key features from the dataset: Duration and ResponseTimeTimeSkewFromMedian. On the FDGs, we can see that there is a correlation between a longer duration and DoH classification. The first graph in Figure 6 shows the trend in shorter durations corresponding to Non-DoH data whereas the second graph shows a large increase in duration for DoH data. This visual representation of difference aligns with the results from our Chi-Squared and Pearson Correlation tests. This is also modeled with the data for ResponseTimeTimeSkewFromMedian in Figure 7 that achieved a p-value of 0. Similar to the duration graphs, the first graph in Figure 6 shows that the Non-DoH data has a greater negative skew, commonly at negative 10, with a small center skew between -2 and $+2$. The second graph shows DoH data that is more likely to fill the -2 to $+2$ range in the skew of response time from the median. The visual explanation helps us to draw more accurate conclusions and better understand the relevance of our defined features in determining the difference between DoH and Non-DoH traffic.

TABLE 4. *P*-values for layer 1 and layer 2 data.

	Layer 1		Layer 2	
#	Feature	p-value	Feature	p-value
1	Duration	0.00	PacketLengthStandardDeviation	0.00
2	ResponseTimeTimeSkewFromMedian	0.00	PacketLengthCoefficientofVariation	0.00
3	ResponseTimeTimeMode	0.00	FlowReceivedRate	$1.45e^{-244}$
4	ResponseTimeTimeMedian	0.00	PacketLengthMean	$7.98e^{-217}$
5	ResponseTimeTimeMean	0.00	Duration	$2.51e^{-216}$
6	PacketTimeSkewFromMedian	0.00	PacketTimeSkewFromMedian	$4.56e^{-188}$
7	PacketTimeMode	0.00	FlowSentRate	$1.86e^{-176}$
8	PacketTimeMedian	0.00	PacketLengthVariance	$5.35e^{-147}$
9	PacketTimeMean	0.00	PacketTimeMean	$1.86e^{-131}$
10	ResponseTimeTimeSkewFromMode	0.00	PacketTimeStandardDeviation	$3.62e^{-129}$
11	PacketTimeVariance	0.00	ResponseTimeTimeMedian	$3.36e^{-115}$
12	PacketLengthCoefficientofVariation	0.00	PacketTimeMedian	$5.14e^{-95}$
13	PacketTimeStandardDeviation	0.00	ResponseTimeTimeSkewFromMode	$9.87e^{-91}$
14	PacketLengthMode	0.00	ResponseTimeTimeMean	$1.58e^{-62}$
15	PacketLengthMedian	0.00	ResponseTimeTimeMode	$4.34e^{-61}$
16	PacketLengthMean	0.00	PacketTimeCoefficientofVariation	$6.08e^{-59}$
17	FlowBytesSent	0.00	ResponseTimeTimeSkewFromMedian	$1.38e^{-49}$
18	ResponseTimeTimeCoefficientofVariation	0.00	PacketTimeMode	$8.24e^{-34}$
19	PacketLengthStandardDeviation	$2.46e^{-133}$	FlowBytesSent	$3.98e^{-29}$
20	PacketLengthVariance	$2.84e^{-130}$	FlowBytesReceived	$4.97e^{-28}$
21	PacketTimeCoefficientofVariation	$2.44e^{-48}$	PacketLengthMode	$3.10e^{-26}$
22	FlowReceivedRate	$3.03e^{-41}$	ResponseTimeTimeCoefficientofVariation	$1.83e^{-14}$
23	ResponseTimeTimeStandardDeviation	$1.89e^{-36}$	PacketLengthSkewFromMedian	$8.14e^{-11}$
24	PacketLengthSkewFromMode	$1.36e^{-11}$	PacketTimeVariance	$8.36e^{-6}$
25	FlowBytesReceived	$2.80e^{-9}$	PacketTimeSkewFromMode	$6.50e^{-5}$
26	PacketLengthSkewFromMedian	$1.87e^{-3}$	PacketLengthMedian	$5.99e^{-5}$
27	FlowSentRate	0.51	ResponseTimeTimeStandardDeviation	0.01
28	ResponseTimeTimeVariance	0.55	ResponseTimeTimeVariance	0.03
29	PacketTimeSkewFromMode	0.64	PacketLengthSkewFromMode	0.99

V. MACHINE LEARNING CLASSIFIERS

In this section, we will be discussing various machine learning classifiers utilized for our research: Decision Tree, Random Forest, LightGBM (LGBM), and XGBoost. These algorithms are discussed more in depth since they were several of the more successful classifiers utilized; however, Linear Discriminant Analysis, K-Nearest Neighbors, Gaussian Naive Bayes, AdaBoost Classifier, Gradient Boosting Classifier, and Extra Trees, were also tested for the purposes of our research.

A. DECISION TREE

Decision tree is a supervised machine learning model that takes a divide-and-conquer approach to classification – which can both provide insightful features and extract patterns from large pools of data [22]. Decision trees use various algorithms which split a node into two or more predecessor nodes. For every split, the algorithm calculates the information gain and entropy of every unused attribute and then selects the highest. Then, it splits again and repeats the process with the unused features. This process continues until all attributes are utilized.

B. RANDOM FOREST

Random forest is an ensemble method where various tree predictors are built independently with each tree underlaid by a bootstrapped sample from the training data. The predictions of the trees are aggregated – this reduces the overall bias of an individual tree – and helps augment the robustness of the prediction [23].

C. LightGBM

LightGBM is a type of gradient boosted decision tree algorithm that was implemented by Microsoft. Most decision trees 'learn' by finding which splits receive the highest information gain or the change in entropy before and after the

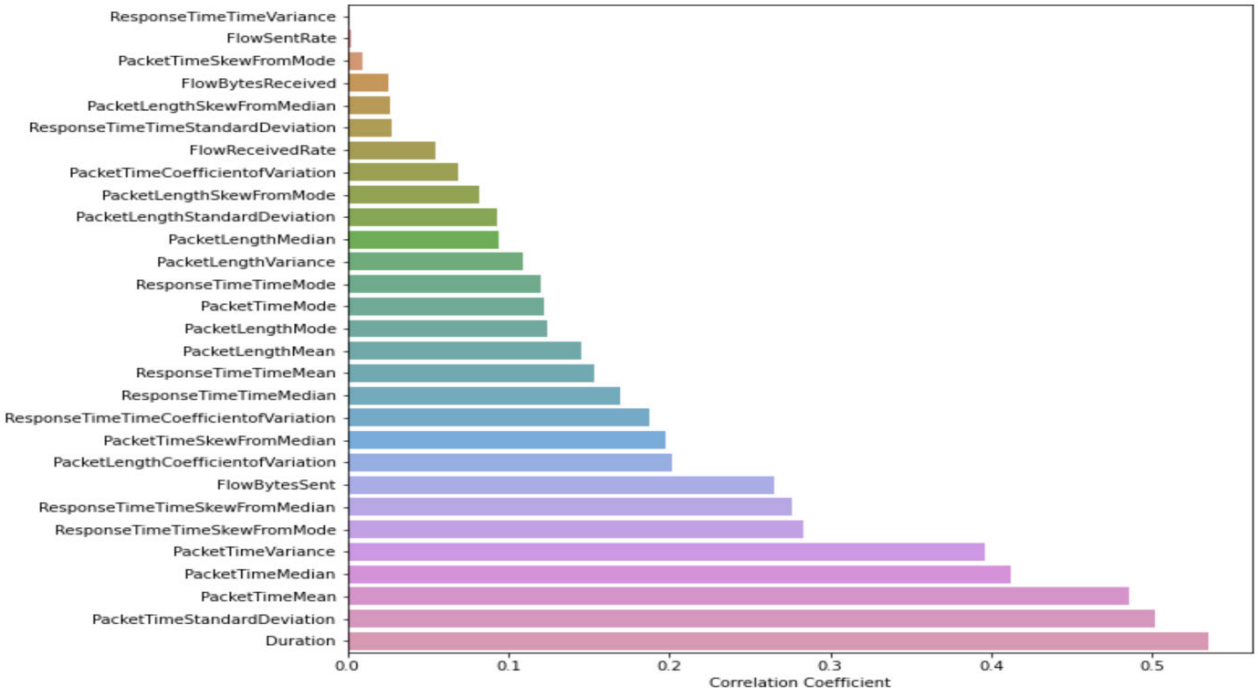

FIGURE 4. The relationship between Pearson correlation coefficients and DoH traffic features. Correlation coefficients are displayed from lower to higher statistical significance.

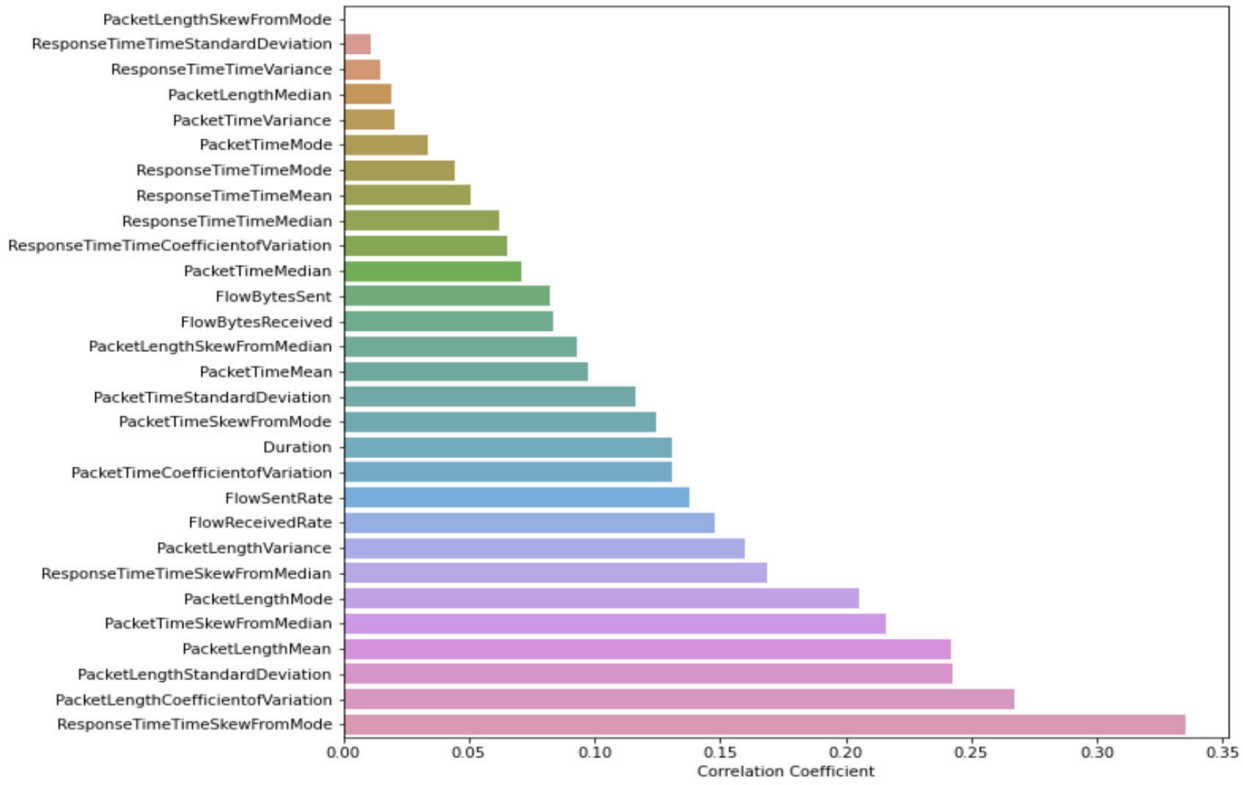

FIGURE 5. The relationship between Pearson correlation coefficients and malicious traffic features. Correlation coefficients are displayed from lower to higher statistical significance.

split. The best split is calculated through either a pre-sorted or histogram-based algorithm; however, it is a time-consuming process, especially when data sets become larger. This is the issue that LightGBM seeks to mitigate. LightGBM utilizes two techniques: Gradient One-Side Sampling (GOSS) and Exclusive Feature Bundling (EFB). GOSS notices that

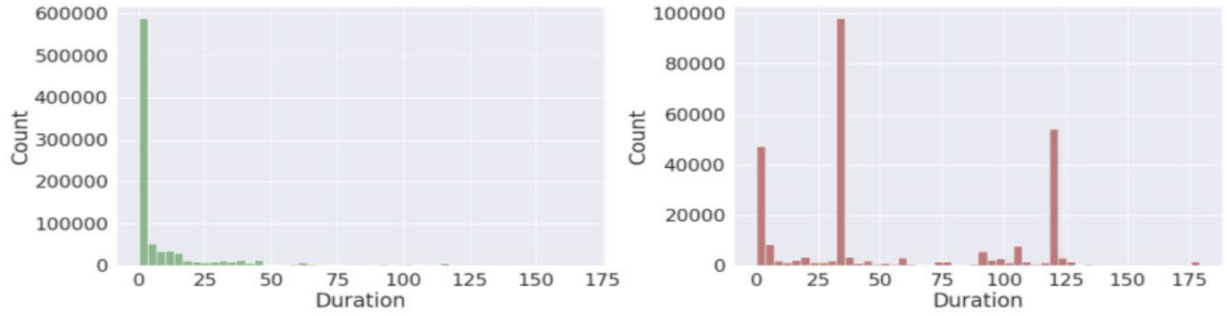

FIGURE 6. The graphs display the NonDoH data in green, the left column, and the DoH data in red, the right column. These are the Duration graphs for layer 1.

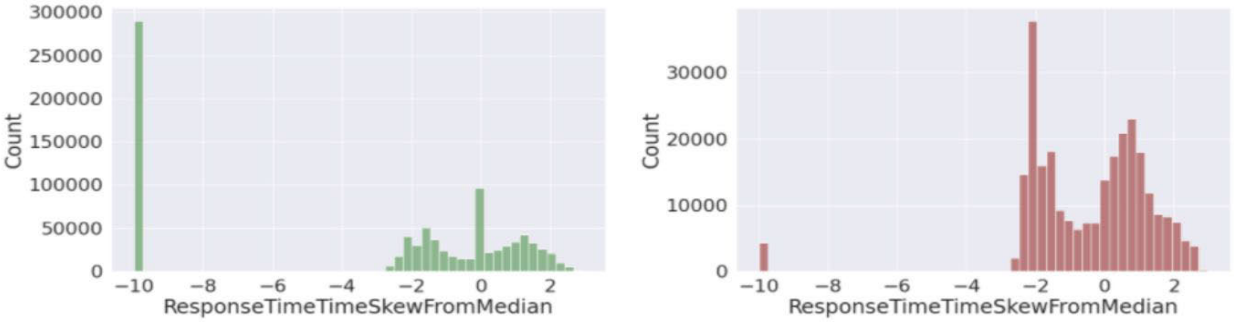

FIGURE 7. The graphs display the Non DoH data in green, the left column, and the DoH data in red, the right column. These are the ResponseTimeTimeSkewFromMedian graphs for layer 1.

gradients play a large role in information gain – specifically – larger gradients correlated with larger information gain. Thus, small gradients are randomly dropped. This method is combined with EFB, which identifies features that are mutually exclusive and reduces them into one feature to reduce time complexity [24].

D. XGBoost

XGBoost (eXtreme Gradient Boosting) is a gradient boosted decision trees algorithm. XGBoost utilizes the Gradient Boosting Machine's (GBM) framework but makes optimizations to it. XGBoost provides many system optimizations that include parallelized decision tree construction, depth first tree pruning, and out of core computing. XGBoost also provides algorithmic enhancements to the GBM framework. These enhancements include LASSO and Ridge regularization to prevent overfitting, sparsity awareness to handle missing data, Weighted Quantile sketch algorithm to find optimal decision tree splits, and built-in cross validation [25].

VI. EXPERIMENTS AND RESULTS

The first step in deciding which machine learning models are effective in predicting Non-Doh vs DoH and Malicious vs Benign DoH data was to deploy an array of different models on all features of both layers of the data set –given that the features are statistically significant (i.e., $p > 0.05$). Decision Tree (DT), Linear Discriminant Analysis (LDA), K-Nearest Neighbors (KNN), Gaussian Naive Bayes (GNB), Random Forest (RF), AdaBoost Classifier (AB), Gradient Boosting Classifier (GB), XGBoost (XGB), Extra Trees (ET), and LGBM were trained on both layers of data. The models were trained using stratified 10-fold cross-validation. This splits the data into 10 folds, where the original ratio of samples is preserved. Each fold will act once as the test split, while the remaining data acts as the training data. This allows the models to retain reliable estimates on unseen data. Accuracy, Precision, Recall, F1-Score, total training time, and prediction time are collected for each fold. Accuracy and total training time are reported in Figure 8, Figure 9, Figure 10, and Figure 11 and provides the mean score for all 10 folds.

A. CLASSIFICATION MEASURES

The classification measures used in subsequent sections of the paper include Accuracy, Recall, Precision, F1-Score, and AUROC (Area Under Receiver Operating Characteristics Curve). These measures are modeled with the following equations:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} * 100 \tag{2}$$

$$\text{Recall} = \frac{TP}{TP + FN} * 100 \tag{3}$$

$$\text{Precision} = \frac{TP}{TP + FP} * 100 \tag{4}$$

$$\text{F1-Score} = \frac{2TP}{2TP + FP + FN} * 100 \tag{5}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{6}$$

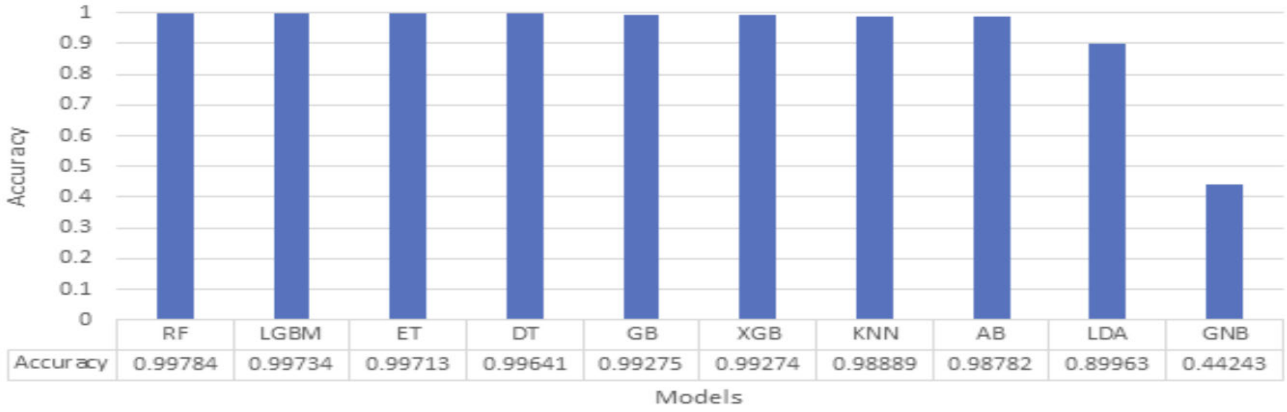

FIGURE 8. Layer 1 accuracies for various machine learning models with all features that satisfy statistical significance. Six models, RF, LGBM, ET, DT, GB, XBG achieved an accuracy over 99%.

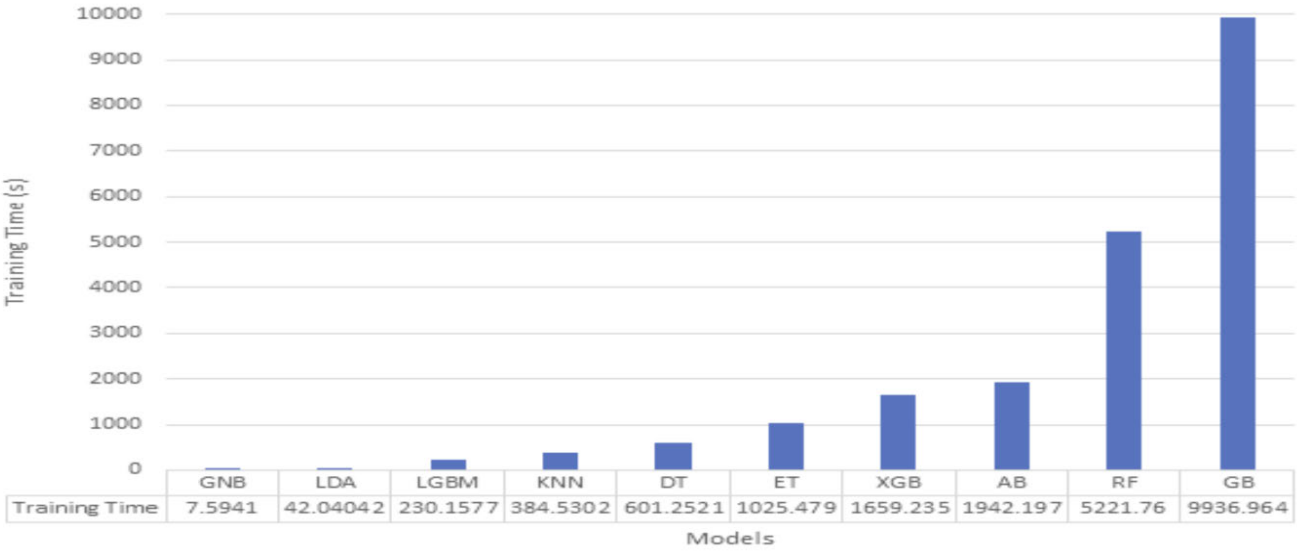

FIGURE 9. Layer 1 training time for various machine learning models with all features that satisfy statistical significance. GNB, LDA, and LGBM had the lowest training times out of all 10 machine learning models.

where, TP is a correct prediction of either DoH traffic in Layer 1 or Malicious DoH activity in Layer 2; conversely, FP is an incorrect prediction of either DoH traffic in Layer 1 or Malicious DoH activity in Layer 2. TN is a correct prediction of either Non-DoH traffic in Layer 1 or Benign DoH traffic in Layer 2 and FN is an incorrect prediction of Non-Doh traffic in Layer 1 or Benign DoH traffic in Layer 2. The ROC curve is the true positive rate (sensitivity/recall) measured against the false positive rate (specificity). The area measured under this curve is the AUROC (sometimes just labeled AUC). ROC (and AUROC) is a useful metric because it displays classification effectiveness regardless of class imbalance in testing data as well as making it easy to see how well a model can distinguish between two classes.

B. INITIAL MACHINE LEARNING MODEL RANKING

The top three models were then selected for the Sequential Forward Selection (SFS) algorithm. For layer 1 data, the random forest model was selected as it had the highest accuracy. Decision Tree and LGBM models were selected due to their similarly high accuracy and low training times. For layer 2 data, Random Forest was selected due to very high accuracy. XGBoost and LGBM were selected due to a combination of high accuracy and low training times.

SFS is then applied for all three machine learning models for both layers of data. SFS starts model training with 1 feature, where the first feature is the most statistically significant which is measured by the lowest p-value (features are added in the sorted order based on Table 4). SFS then adds the next most statistically significant feature and the models

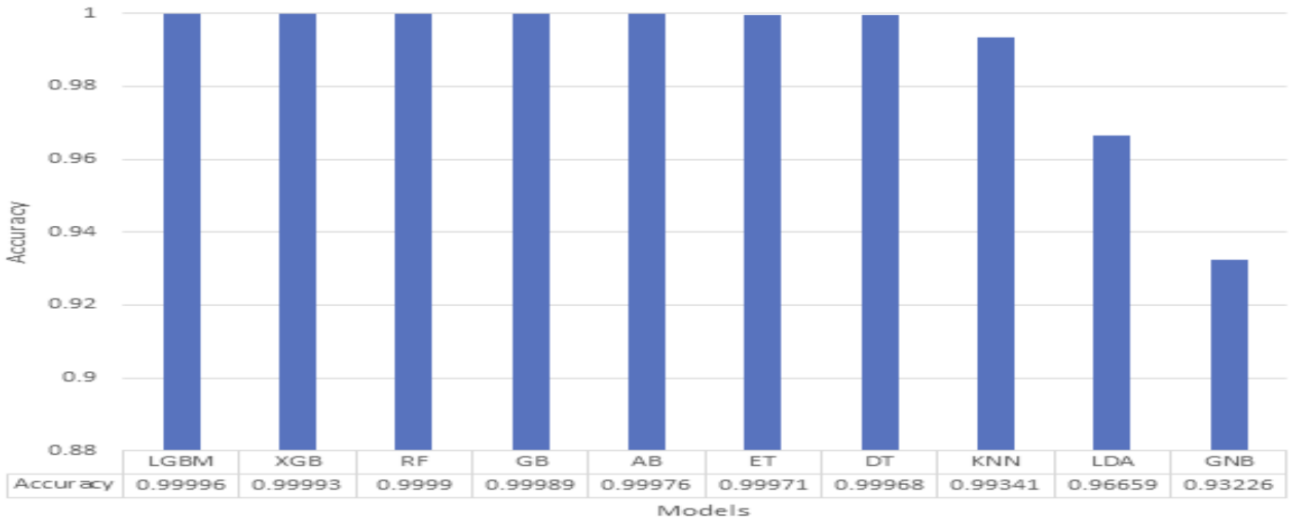

FIGURE 10. Layer 2 accuracies for various machine learning models with all features that satisfy statistical significance. Six models achieved accuracies over 99.9% (LGBM, XGB, RF, GB, AB, ER, DT).

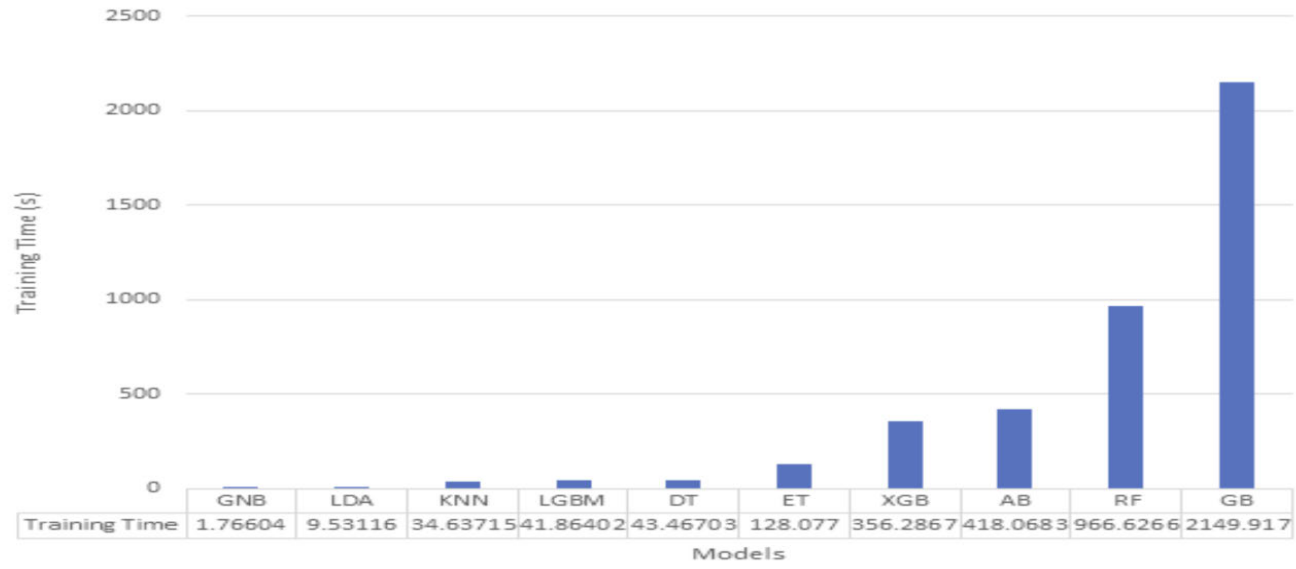

FIGURE 11. Layer 1 training time for various machine learning models with all features that satisfy statistical significance. GNB, LDA, and KNN had the lowest training times out of all ten machine learning models.

train on the new subset of features. This process repeats until all statistically significant features have been added and trained on.

VII. SEQUENTIAL FORWARD SELECTION TRAINING RESULTS

Both layers of data go through Sequential Forward Selection (SFS) training and metrics of accuracy, recall, precision, F1-Score, training time, prediction time, and AUROC. The results for Layer 1 are displayed in Figure 12 and Figure 13. Figure 12 shows that accuracy, recall, precision, and F1-score scores for all models become exceedingly close to no error as the number of features reaches 20. The total training time of these precise models differ vastly. Random Forest took the longest to train for any number of features, where many of the trials resulted in a training time taking over an hour. Both decision tree and LGBM models took considerably less time to train with LGBM taking the shortest training time. Prediction time, approximately two seconds on average, for Random Forest was the longest out of the three models to predict one stratified test fold. Both Decision Tree and LGBM had exceptionally fast prediction times with less than 0.5 seconds, Decision Tree predicting DoH vs Non-DoH data the fastest. The results for Layer 2 data can be seen in Figure 14 and Figure 15.

From Figure 14, it can be observed that all three models approach 0% error at approximately 20 features. Random Forest, similar to Layer 1's results, has the longest training time. In Layer 2 data, Random Forest has a longer training time of approximately 1,100 seconds. XGBoost and LGBM

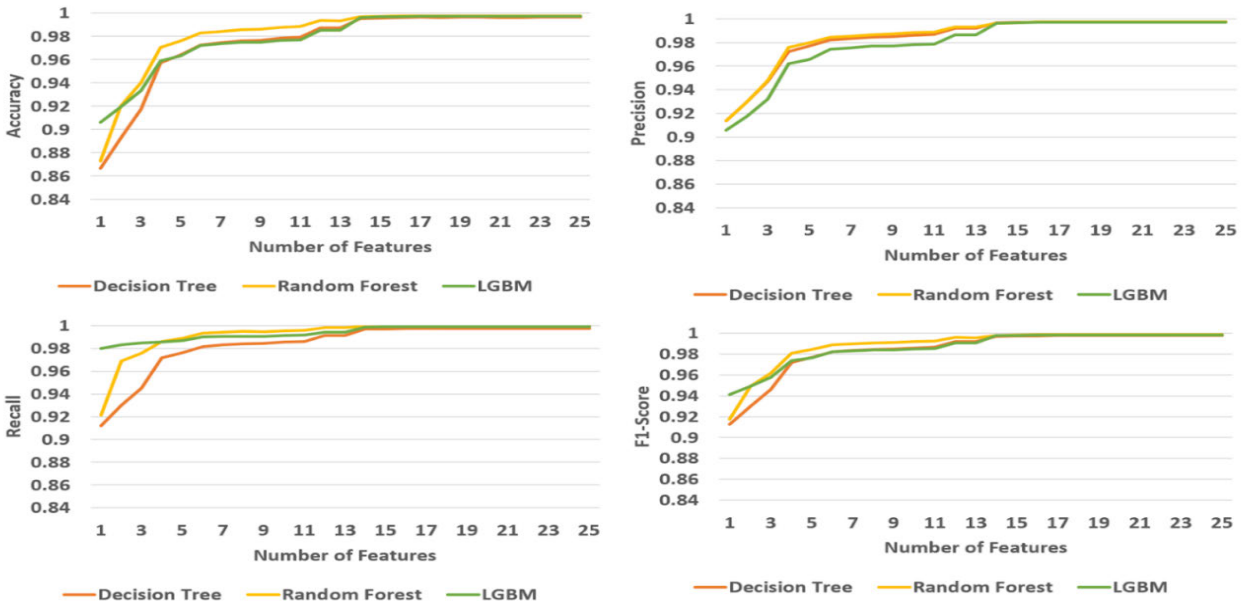

FIGURE 12. Shows accuracy, precision, recall and F1-score metrics for Non-DoH vs DoH data for SFS using Decision Tree, Random Forest and LGBM models.

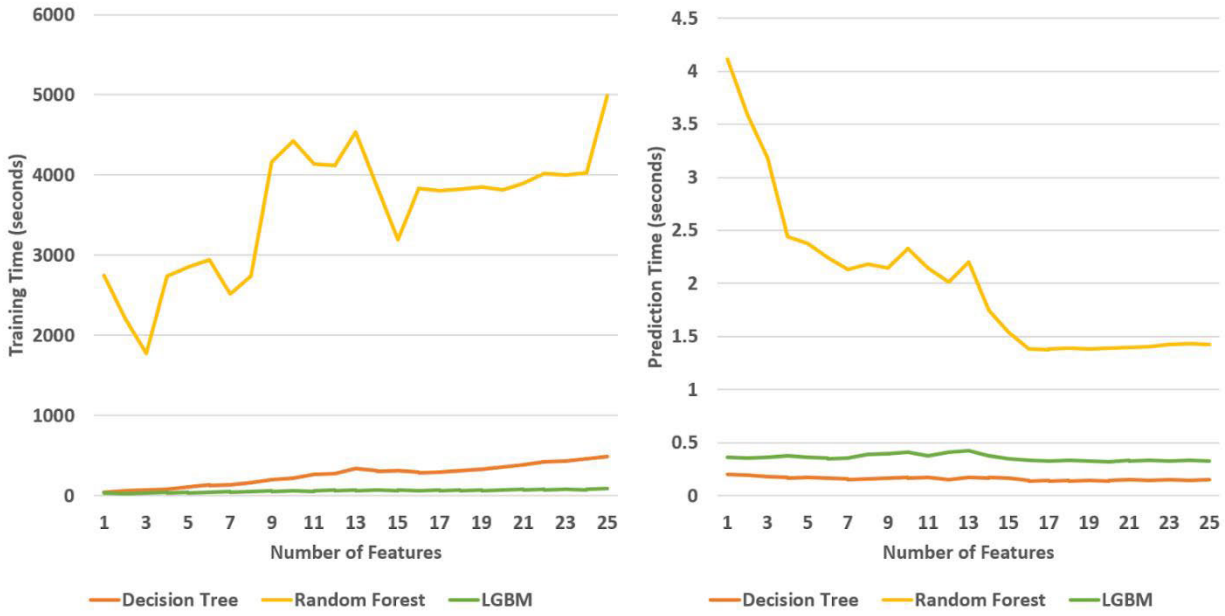

FIGURE 13. Average training time and prediction time for Non-DoH vs DoH data for SFS using Decision Tree, Random Forest and LGBM models during 10-fold cross validation.

both have lower training times, with LGBM having the lowest training time. For prediction time, Random Forest once again took the longest with approximately 0.25 seconds to predict 1 stratified fold. In this Layer, XGBoost outperforms LGBM in prediction for most features, until the number of features reaches 21, where they become approximately the same. As seen in figure 16, the layer 1 AUROCs are comparably the same. In layer 2, XGBoost lags behind LGBM and Random Forest until 21 features where all three models approach 1.

VIII. DISCUSSION, LIMITATIONS, AND FUTURE WORK

From the accuracy metrics in Figure 12 and 14, it can be observed that all the considered models, LGBM, Random Forest, Decision Tree, and XGBM approach 0% misclassification error for Non-DoH vs DoH traffic and benign vs malicious DoH traffic. Despite all models demonstrating exceptionally high accuracy at 21 features, there were noticeable differences in the training and prediction times for the models. LGBM was the fastest model, with a total training time of approximately 87 seconds for Layer 1 data

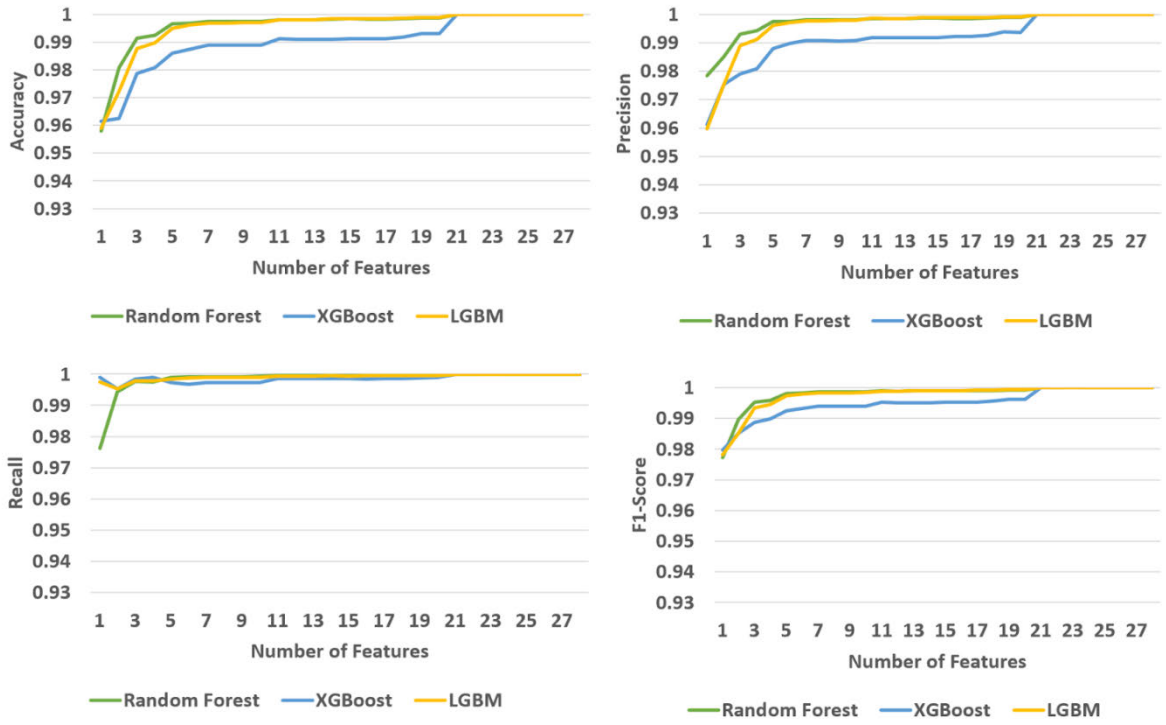

FIGURE 14. Accuracy, precision, recall and F1-score metrics for benign vs malicious DoH data for SFS using Decision Tree, Random Forest and LGBM models.

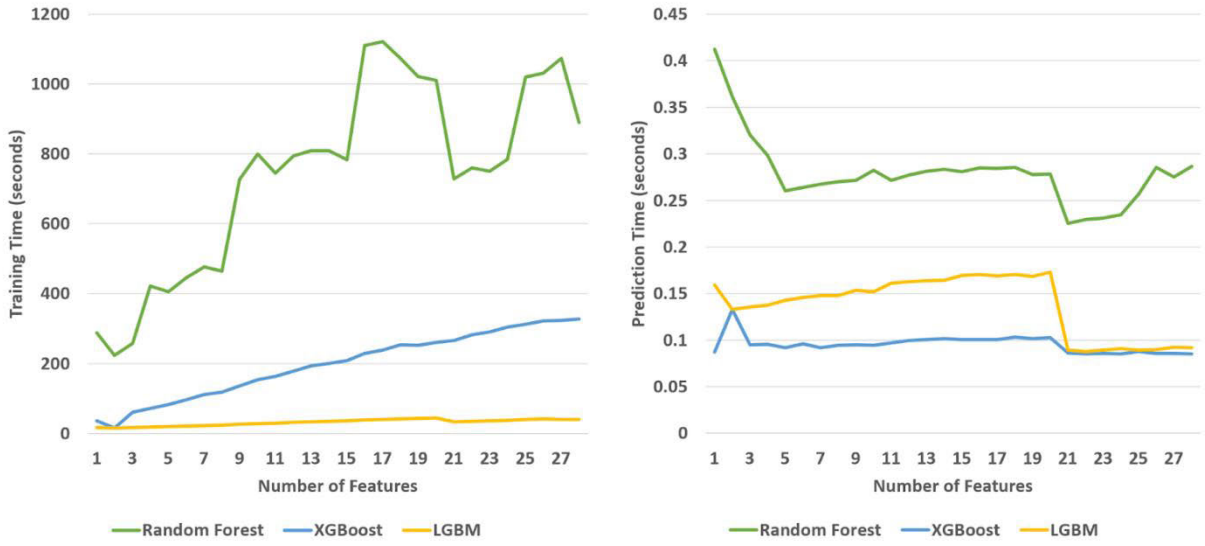

FIGURE 15. Average Training time (left) and prediction time (right) for benign vs malicious DoH data for SFS using Decision Tree (blue), Random Forest (green), and LGBM (yellow) models during 10-fold cross validation.

with 25 features and approximately 40 seconds for Layer 2 data with 27 features. This is in stark contrast to the Random Forest model, which had comparably egregious training times of 4,991 seconds (1.39 hours) for Layer 1 data with 25 features and approximately 890 seconds (14.83 minutes) for 27 features in Layer 2 data. This implies that LGBM would be a preferred model if the model is continually trained with new data instances. If the deployed model does not forgo any further online training, Random Forest may be a preferred model for both layers of data. All models demonstrated relatively low prediction times. From Fig. 13 and 15, all models had an average precision time in 10-fold cross-validation under one second for both layers of data at the optimal 21 features. This is a very fast prediction time and implies that any model could be deployed and perform well.

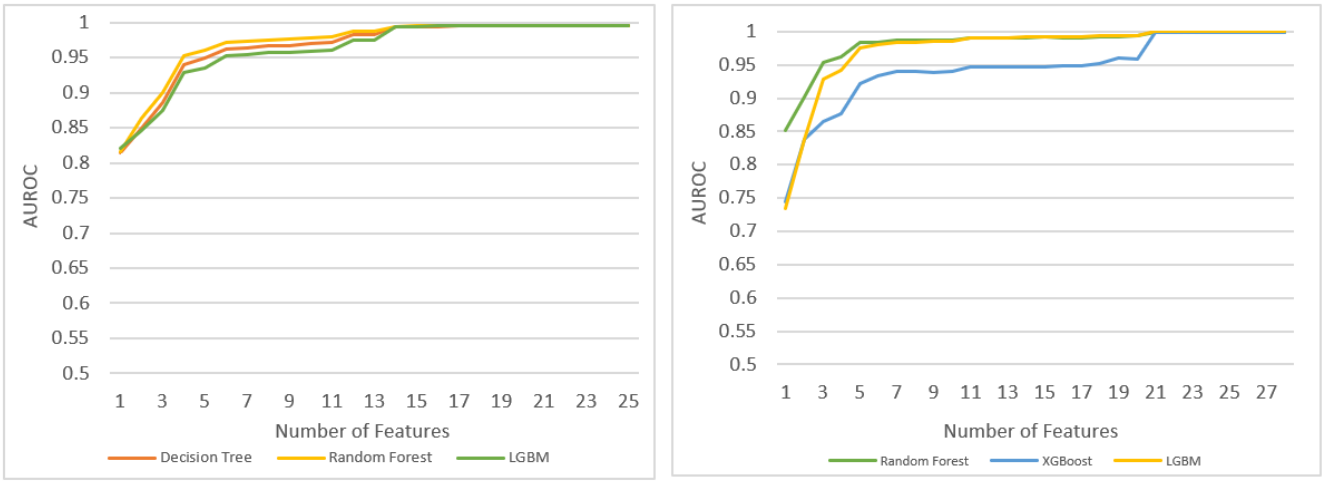

FIGURE 16. Graph comparing AUROC for Random Forest, XGBoost, and LGBM for Non-DoH vs DoH data (left) and benign vs malicious DoH data (right).

The present study has some limitations which also provide avenues for future research. The models that were evaluated (Random Forest, LGBM, XGBoost, Decision Tree, Linear Discriminant Analysis, K-Nearest Neighbors, Gaussian Naive Bayes, AdaBoost, Gradient Boosting, and Extra Trees) on the data were default models in scikit-learn. This means there is an opportunity for future work to evaluate the impact of hyperparameter tuning on each model. By performing hyperparameter tuning, the accuracy of the machine learning models may perform at optimal levels with a fewer number of features, thereby reducing the amount of training time needed. Another popular machine learning method, Artificial Neural Networks (ANN), was not considered in this study. Future work can be conducted where different ANN architectures can be compared to already high performing machine learning models like Random Forest, Decision Tree, LGBM, and XGBoost.

IX. CONCLUSION

Our experiments critiqued and improved Banadaki's [11] research. Specifically, we removed extra noise, introduced feature selection methods and provided explainability to the features that characterize the dataset, which showed detection of malicious DoH traffic more accurately and efficiently. These modifications, specifically, removing overfitting features and creating an applicable model from a generic feature set make our research methods more suitable to real-world applications. In addition, by eliminating multiple insignificant features identified using the Chi-Squared test and PCC test, we see a large improvement in training and testing the models in our experiments. With fewer features to assess, the model will train and predict faster without a loss in accuracy. In summary, our study removes overfitting features and insignificant data creating a faster, more accurate study on DoH detection. Ultimately, we recommend the LGBM model to distinguish between Non-DoH and DoH data and malicious and benign DoH traffic because of its exceptional classification accuracy and low comparative training time compared to other machine learning classifiers such as Random Forest, Decision Tree and XGBoost. These promising results offer avenues for future work on DoH classification research including how deep learning would compare to LGBM in terms of accuracy and time.

ACKNOWLEDGMENT

Any opinions, findings and conclusions, or recommendations expressed in this article are those of the authors and do not necessarily reflect the views of the funding sources. The authors would like to thank Clayton P. Johnson and Bishal Khadka for their early help with getting this research project started.

REFERENCES

[1] P. Mockapetris and K. J. Dunlap, "Development of the domain name system," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 18, no. 4, pp. 123–133, Aug. 1988, doi: 10.1145/52325.52338.

[2] D. Hjelm. (2019). *A New Needle and Haystack: Detecting DNS Over HTTPS Usage*. [Online]. Available: https://www.sans.org/reading-room/whitepapers/dns/needle-haystack-detecting-dns-https-usage-39160

[3] M. Konopa, J. Fesl, J. Jelínek, M. Feslová, J. Cehák, J. Janeček, and F. Drdák, "Using machine learning for DNS over HTTPS detection," in *Proc. 19th Eur. Conf. Cyber Warfare*, 2020, p. 205, doi: 10.34190/ews.20.001.

[4] Y. Zhao, N. Hu, C. Zhang, and X. Cheng, "DCG: A client-side protection method for DNS cache," *J. Internet Services Inf. Secur.*, vol. 10, no. 2, pp. 103–121, 2020, doi: 10.22667/JISIS.2020.05.31.103.

[5] A. Khormali, J. Park, H. Alasmary, A. Anwar, M. Saad, and D. Mohaisen, "Domain name system security and privacy: A contemporary survey," *Comput. Netw.*, vol. 185, Feb. 2021, Art. no. 107699, doi: 10.1016/j.comnet.2020.107699.

[6] S. Siby, M. Juarez, C. Diaz, N. Vallina-Rodriguez, and C. Troncoso, "Encrypted DNS—Privacy? A traffic analysis perspective," in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–8, doi: 10.14722/ndss.2020.24301.

[7] C. Lu, B. Liu, Z. Li, S. Hao, H. Duan, M. Zhang, C. Leng, Y. Liu, Z. Zhang, and J. Wu, "An end-to-end, large-scale measurement of DNS-over-encryption: How far have we come?" in *Proc. Internet Meas. Conf. (IMC)*. New York, NY, USA: Association for Computing Machinery, Oct. 2019, pp. 22–35, doi: 10.1145/3355369.3355580.

[8] Q. Huang, D. Chang, and Z. Li, "A comprehensive study of DNS-over-HTTPS downgrade attack," in *Proc. 10th USENIX Workshop Free Open Commun. Internet*. Berkeley, CA, USA: USENIX Association, 2020, pp. 1–8. [Online]. Available: https://www.usenix.org/conference/foci20/presentation/huang

[9] E. Blidborg and C. Gunnarsson. (2020). *Cache Poisoning in DNS Over HTTPS Clients*. [Online]. Available: http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-278059
[10] National Security Agency Cybersecurity Information. (2021). *Adopting Encrypted DNS in Enterprise Environments*. [Online]. Available: https://media.defense.gov/2021/Jan/14/2002564889/-1/-1/0/CSI_ADOPTING_ENCRYPTED_DNS_U_OO_102904_21.PDF
[11] Y. M. Banadaki, "Detecting malicious DNS over HTTPS traffic in domain name system using machine learning classifiers," *J. Comput. Sci. Appl.*, vol. 8, no. 2, pp. 46–55, Aug. 2020, doi: 10.12691/jcsa-8-2-2.
[12] R. Cox, A. Muthitacharoen, and R. Morris, "Serving DNS using a peer-to-peer lookup service," in *Peer-to-Peer Systems*. Berlin, Germany: Springer, 2002, doi: 10.1007/3-540-45748-8_15.
[13] L. Bilge, S. Sen, D. Balzarotti, E. Kirda, and C. Kruegel, "Exposure: A passive DNS analysis service to detect and report malicious domains," *ACM Trans. Inf. Syst. Secur.*, vol. 16, no. 4, pp. 1–28, Apr. 2014, doi: 10.1145/2584679.
[14] F. Nijeboer. (2020). *Detection of https Encrypted DNS Traffic*. [Online]. Available: http://essay.utwente.nl/82085/1/Nijeboer_BA_EEMCS.pdf
[15] G. Regkas. (2020). *Empowering Citizen Data Scientists With Watson AutoAI*. [Online]. Available: https://towardsdatascience.com/empowering-citizen-data-scientists-with-watson-autoai-49a087df99e5
[16] J. Jiang, J. Chen, K.-K. R. Choo, C. Liu, K. Liu, M. Yu, and Y. Wang, "A deep learning based online malicious URL and DNS detection scheme," in *Proc. Int. Conf. Secur. Privacy Commun. Syst.*, in Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 2018, pp. 438–448, doi: 10.1007/978-3-319-78813-5_22.
[17] F. Palau, C. Catania, J. L. Guerra, S. García, and M. Rigaki, "Detecting DNS threats: A deep learning model to rule them all," in *Proc. XX Simposio Argentino de Inteligencia Artif. (ASAI JAIIO)*, vol. 48, 2019, pp. 1–12, doi: 10.13140/RG.2.2.14296.03849.
[18] M. MontazeriShatoori, L. Davidson, G. Kaur, and A. H. Lashkari, "Detection of DoH tunnels using time-series classification of encrypted traffic," in *Proc. 5th IEEE Cyber Sci. Technol. Congr.*, Calgary, AB, Canada, Aug. 2020, pp. 63–70. [Online]. Available: https://www.nb.ca/cic/datasets/dohbrw-2020.html
[19] R. L. Plackett, "Karl Pearson and the chi-squared test," *Int. Stat. Rev./Revue Int. de Statistique*, vol. 51, no. 1, p. 59, Apr. 1983, doi: 10.2307/1402731.
[20] E. Chitsaz, M. Taheri, S. Katebi, and M. Jahromi, "An improved fuzzy feature clustering and selection based on chi-squared-test," in *Proc. Int. Multiconf. Eng. Comput. Sci.*, 2009, pp. 1–6. [Online]. Available: https://www.researchgate.net/publication/44259514_An_Improved_Fuzzy_Feature_Clustering_and_Selection_based_on_Chi-Squared-Test
[21] J. Adler and I. Parmryd, "Quantifying colocalization by correlation: The Pearson correlation coefficient is superior to the Mander's overlap coefficient," *Cytometry A*, vol. 77A, no. 8, pp. 733–742, Mar. 2010, doi: 10.1002/cyto.a.20896.
[22] A. J. Myles, R. N. Feudale, Y. Liu, N. A. Woody, and S. D. Brown, "An introduction to decision tree modeling," *J. Chemometrics*, vol. 18, no. 6, pp. 275–285, Jun. 2004, doi: 10.1002/cem.873.
[23] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, pp. 261–277, Dec. 2001, doi: 10.1023/a:1017934522171.
[24] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A highly efficient gradient boosting decision tree," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 3146–3154. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf
[25] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

MATTHEW BEHNKE is currently pursuing the bachelor's degree in computer science minor in mathematics and a professional certificate in cyber security with Colorado Mesa University (CMU). He will be pursuing the Ph.D. degree in computer science with the University of Central Florida, in Fall 2021. He is also a Treasurer of Upsilon Pi Epsilon (the International Honor Society for Computing and Informatics) and the Vice President of Kappa Mu Epsilon (the National Mathematics Honor Society) at CMU.

NATHAN BRINER is currently pursuing the bachelor's degree in computer science and a minor in mathematics with Colorado Mesa University. He is also a Proud Member of the Cybersecurity Club and looks forward to doing more research involving machine learning.

DRAKE CULLEN is currently pursuing the bachelor's degree in computer science, statistics, applied mathematics, and a minor in cybersecurity with Colorado Mesa University (CMU). He is also the President of the Cybersecurity Club and a Research Fellow with the Cybersecurity Center, CMU.

KATELYNN SCHWERDTFEGER is currently pursuing the bachelor's degree in computer science with minors in mathematics and Spanish with Colorado Mesa University (CMU). She is also a Research Assistant with the Cybersecurity Center, CMU. Her research interests include the areas of machine learning and data science.

JACKSON WARREN is currently pursuing the bachelor's degree in computer science with Colorado Mesa University. He is currently researching DDoS attacks and neural networks, with an interest in machine learning and theory of algorithms.

RAM BASNET received the B.S. degree in computer science from Colorado Mesa University (CMU), in 2004, and the M.S. and Ph.D. degrees in computer science from New Mexico Tech, in 2008 and 2012, respectively. He is currently an Associate Professor of computer science and cybersecurity with CMU. His research interests include the areas of information assurance, machine learning, and computer science pedagogy.

TENZIN DOLECK received the Ph.D. degree from McGill University, in 2017. He is currently working as Canada Research Chair and an Assistant Professor with Simon Fraser University.