A Comparative Analysis of Machine Learning Algorithms for Website Traffic Classification from Network Packets

Matthew Berthoud, Lake Bradford, Justin Cresent, Will Katabian
William & Mary
Williamsburg, Virginia, USA

ABSTRACT

Accurately Identifying web traffic destination and origins is crucial for the efficiency of a network. This project explores the potential of machine learning in reference to web traffic classification based on the analysis of network packets. We monitored and analyzed web traffic data from ChatGPT, Blackboard, and Linkedin, with the objective of building models which will be able to predict the web traffic origin of a specific packet. The collection of data was performed using wireshark, then the data was reformatted to eliminate bias and get more accurate results. Using the collected data we then trained four models which had varying levels of accuracy, Logistic Regression (56%), K-Nearest Neighbors (77%), Random Forest (78%), and finally a neural network (80%). This project shows the importance of machine learning within the field of network traffic analysis as automaton is much more efficient and precise compared to manually examining web traffic, especially in a scale as large as the internet. One example of our project's significance is that this can be crucial data analysis for network administrators and security professionals, whom would examine the network for malicious traffic.

KEYWORDS

Network Traffic, Machine Learning, Web Traffic Classification, Network Packets, Data Analysis, Neural Networks, Random Forest, K-Nearest Neighbors, Logistic Regression

1 INTRODUCTION

The ability to monitor and analyze network traffic is crucial for the efficiency and security of a network [4]. It helps to manage the overall network performance, detect and prevent malicious activities, and ensure the network is operating as intended. The present day internet is composed of a vast variety of diverse web traffic, of which requires a more sophisticated approach to analyze and classify network traffic[2]. This project investigates a variety of machine learning algorithms to see which most accurately and effectively classify web traffic based on data from a set of captured network packets.

Traffic classification is very significant in practice, if done accurately and effectively [5]. For reasons mentioned previously, the observed capabilities of this and other projects can be crucial for network administrators and security professionals, whom would examine the network for malicious traffic.

For this project, many sets of packet data were collected from three popular websites: ChatGPT, Blackboard, and Linkedin. These sets were then merged into a single dataset, which was then split into training and testing sets. Four models were then trained and tested on this data: Logistic Regression, K-Nearest Neighbors, Random Forest, and Neural Network [3].

A thorough evaluation of the models was conducted through testing and validation sets, hyperparameter optimization employing cross-validation and grid search, and computing accuracy and Macro F1 scores. The results showed that the Neural Network model acheived the highest accuracy of 80% and Macro F1 score of BLANK. These scores display the model's ability to accurately classify the selected websites based on captured network packet data.

Overall, the significance of this project is seen in its use of Machine Learning and the extensions of these applications into the discipline of network traffic analysis. This practice has potential to be utilized in real-world applications, and many sources proving it already is [2].

2 PROPOSED METHOD

This section presents the methodology for how this project goes about capturing network packet data, analyzing the data, and classifying it based on its website of origin.

2.1 Capturing Data

2.1.1 Network analysis tool. For this project, the tool chosen and used was Wireshark [1], a widely used network protocol analyzer. Wireshark is capable of capturing and analyzing network packets, and is able to provide a detailed view of the packets exchanged between the client and server. Wireshark offers a broad variety of features that allow for users to capture, decode, and analyze network packets. It shows the packet data in a human-readable format, and provides

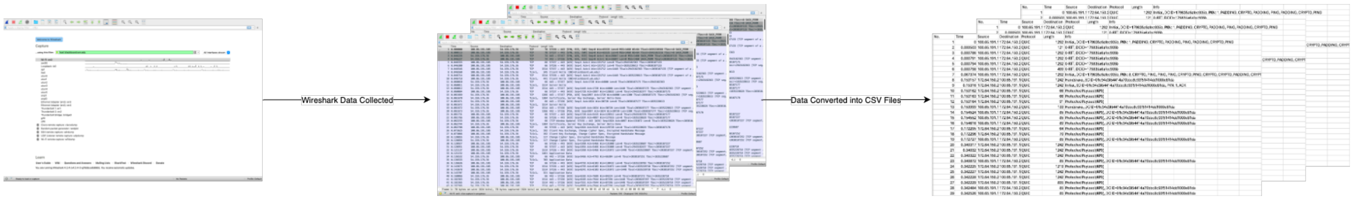
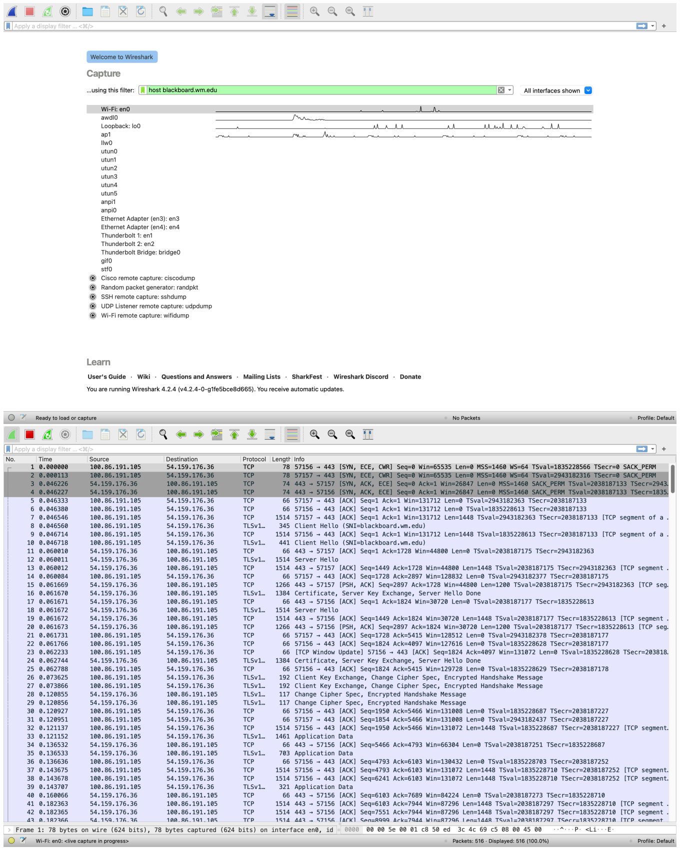


Figure 1: Wireshark Data Collection Process



Welcome to Wireshark

Capture

...using this filter: host blackboard.wm.edu

All interfaces shown

Wi-Fi: en0
awdl0
Loopback: lo0
ap1
llw0
utun0
utun1
utun2
utun3
utun4
utun5
anpi1
anpi0
Ethernet Adapter (en3): en3
Ethernet Adapter (en4): en4
Thunderbolt 1: en1
Thunderbolt 2: en2
Thunderbolt Bridge: bridge0
gif0
stf0
Cisco remote capture: ciscodump
Random packet generator: randpkt
SSH remote capture: sshdump
UDP Listener remote capture: udpdump
Wi-Fi remote capture: wifidump

Learn

User's Guide · Wiki · Questions and Answers · Mailing Lists · SharkFest · Wireshark Discord · Donate

You are running Wireshark 4.2.4 (v4.2.4-0-g1fe5bce8d665). You receive automatic updates.

Ready to load or capture

No Packets

Profile: Default

```
No.   Time       Source          Destination     Protocol Length Info
    1 0.000000   100.86.191.105  54.159.176.36   TCP        78 57156 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=1835228566 TSecr=0 SACK_PERM
    2 0.000113   100.86.191.105  54.159.176.36   TCP        78 57157 → 443 [SYN, ECE, CWR] Seq=0 Win=65535 Len=0 MSS=1460 WS=64 TSval=2943182316 TSecr=0 SACK_PERM
    3 0.046226   54.159.176.36   100.86.191.105  TCP        74 443 → 57157 [SYN, ACK, ECE] Seq=0 Ack=1 Win=26847 Len=0 MSS=1460 SACK_PERM TSval=2038187133 TSecr=2943…
    4 0.046227   54.159.176.36   100.86.191.105  TCP        74 443 → 57156 [SYN, ACK, ECE] Seq=0 Ack=1 Win=26847 Len=0 MSS=1460 SACK_PERM TSval=2038187133 TSecr=1835…
    5 0.046333   100.86.191.105  54.159.176.36   TCP        66 57157 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=2943182363 TSecr=2038187133
    6 0.046380   100.86.191.105  54.159.176.36   TCP        66 57156 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=0 TSval=1835228613 TSecr=2038187133
    7 0.046546   100.86.191.105  54.159.176.36   TCP      1514 57157 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=2943182363 TSecr=2038187133 [TCP segment of a …
    8 0.046560   100.86.191.105  54.159.176.36   TLSv1…    345 Client Hello (SNI=blackboard.wm.edu)
    9 0.046714   100.86.191.105  54.159.176.36   TCP      1514 57156 → 443 [ACK] Seq=1 Ack=1 Win=131712 Len=1448 TSval=1835228613 TSecr=2038187133 [TCP segment of a …
   10 0.046718   100.86.191.105  54.159.176.36   TLSv1…    441 Client Hello (SNI=blackboard.wm.edu)
   11 0.060010   54.159.176.36   100.86.191.105  TCP        66 443 → 57157 [ACK] Seq=1 Ack=1728 Win=44800 Len=0 TSval=2038187175 TSecr=2943182363
   12 0.060011   54.159.176.36   100.86.191.105  TLSv1…   1514 Server Hello
   13 0.060012   54.159.176.36   100.86.191.105  TCP      1514 443 → 57157 [ACK] Seq=1449 Ack=1728 Win=44800 Len=1448 TSval=2038187175 TSecr=2943182363 [TCP segment …
   14 0.060084   100.86.191.105  54.159.176.36   TCP        66 57157 → 443 [ACK] Seq=1728 Ack=2897 Win=128832 Len=0 TSval=2943182377 TSecr=2038187175
   15 0.061669   54.159.176.36   100.86.191.105  TCP      1266 443 → 57157 [PSH, ACK] Seq=2897 Ack=1728 Win=44800 Len=1200 TSval=2038187175 TSecr=2943182363 [TCP seg…
   16 0.061670   54.159.176.36   100.86.191.105  TLSv1…   1384 Certificate, Server Key Exchange, Server Hello Done
   17 0.061671   54.159.176.36   100.86.191.105  TCP        66 443 → 57156 [ACK] Seq=1 Ack=1824 Win=30720 Len=0 TSval=2038187177 TSecr=1835228613
   18 0.061672   54.159.176.36   100.86.191.105  TLSv1…   1514 Server Hello
   19 0.061672   54.159.176.36   100.86.191.105  TCP      1514 443 → 57156 [ACK] Seq=1449 Ack=1824 Win=30720 Len=1448 TSval=2038187177 TSecr=1835228613 [TCP segment …
   20 0.061673   54.159.176.36   100.86.191.105  TCP      1266 443 → 57156 [PSH, ACK] Seq=2897 Ack=1824 Win=30720 Len=1200 TSval=2038187177 TSecr=1835228613 [TCP seg…
   21 0.061731   100.86.191.105  54.159.176.36   TCP        66 57157 → 443 [ACK] Seq=1728 Ack=5415 Win=128512 Len=0 TSval=2943182378 TSecr=2038187177
   22 0.061766   100.86.191.105  54.159.176.36   TCP        66 57156 → 443 [ACK] Seq=1824 Ack=4097 Win=127616 Len=0 TSval=1835228628 TSecr=2038187177
   23 0.062233   100.86.191.105  54.159.176.36   TCP        66 [TCP Window Update] 57156 → 443 [ACK] Seq=1824 Ack=4097 Win=131072 Len=0 TSval=1835228628 TSecr=203818…
   24 0.062744   54.159.176.36   100.86.191.105  TLSv1…   1384 Certificate, Server Key Exchange, Server Hello Done
   25 0.062788   100.86.191.105  54.159.176.36   TCP        66 57156 → 443 [ACK] Seq=1824 Ack=5415 Win=129728 Len=0 TSval=1835228629 TSecr=2038187178
   26 0.073625   100.86.191.105  54.159.176.36   TLSv1…    192 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
   27 0.073866   100.86.191.105  54.159.176.36   TLSv1…    192 Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
   28 0.120855   54.159.176.36   100.86.191.105  TLSv1…    117 Change Cipher Spec, Encrypted Handshake Message
   29 0.120856   54.159.176.36   100.86.191.105  TLSv1…    117 Change Cipher Spec, Encrypted Handshake Message
   30 0.120927   100.86.191.105  54.159.176.36   TCP        66 57156 → 443 [ACK] Seq=1950 Ack=5466 Win=131008 Len=0 TSval=1835228687 TSecr=2038187227
   31 0.120951   100.86.191.105  54.159.176.36   TCP        66 57157 → 443 [ACK] Seq=1854 Ack=5466 Win=131008 Len=0 TSval=2943182437 TSecr=2038187227
   32 0.121137   100.86.191.105  54.159.176.36   TCP      1514 57156 → 443 [ACK] Seq=1950 Ack=5466 Win=131072 Len=1448 TSval=1835228687 TSecr=2038187227 [TCP segment…
   33 0.121152   100.86.191.105  54.159.176.36   TLSv1…   1461 Application Data
   34 0.136532   54.159.176.36   100.86.191.105  TCP        66 443 → 57156 [ACK] Seq=5466 Ack=4793 Win=66304 Len=0 TSval=2038187251 TSecr=1835228687
   35 0.136533   54.159.176.36   100.86.191.105  TLSv1…    703 Application Data
   36 0.136636   100.86.191.105  54.159.176.36   TCP        66 57156 → 443 [ACK] Seq=4793 Ack=6103 Win=130432 Len=0 TSval=1835228703 TSecr=2038187252
   37 0.143675   100.86.191.105  54.159.176.36   TCP      1514 57156 → 443 [ACK] Seq=4793 Ack=6103 Win=131072 Len=1448 TSval=1835228710 TSecr=2038187252 [TCP segment…
   38 0.143678   100.86.191.105  54.159.176.36   TCP      1514 57156 → 443 [ACK] Seq=6241 Ack=6103 Win=131072 Len=1448 TSval=1835228710 TSecr=2038187252 [TCP segment…
   39 0.143707   100.86.191.105  54.159.176.36   TLSv1…    321 Application Data
   40 0.160066   54.159.176.36   100.86.191.105  TCP        66 443 → 57156 [ACK] Seq=6103 Ack=7689 Win=84224 Len=0 TSval=2038187273 TSecr=1835228710
   41 0.182363   54.159.176.36   100.86.191.105  TCP      1514 443 → 57156 [ACK] Seq=6103 Ack=7944 Win=87296 Len=1448 TSval=2038187297 TSecr=1835228710 [TCP segment …
   42 0.182365   54.159.176.36   100.86.191.105  TCP      1514 443 → 57156 [ACK] Seq=7551 Ack=7944 Win=87296 Len=1448 TSval=2038187297 TSecr=1835228710 [TCP segment …
   43 0.182366   54.159.176.36   100.86.191.105  TCP      1514 443 → 57156 [ACK] Seq=8999 Ack=7944 Win=87296 Len=1448 TSval=2038187297 TSecr=1835228710 [TCP segment
> Frame 1: 78 bytes on wire (624 bits), 78 bytes captured (624 bits) on interface en0, id   0000  00 00 5e 00 01 c8 50 ed  3c 4c 69 c5 08 00 45 00   ··^···P· <Li···E·
```

Wi-Fi: en0: <live capture in progress>

Packets: 516 · Displayed: 516 (100.0%)

Profile: Default

Figure 2: Wireshark interface and example of captured data

a detailed view of the packet's contents. This data contains information on time taken for packet to send and arrive, protocols used, size of each packet, and other fields such as more information on the packet and the source IP address. This tool also supports use of different forms of link communication and can be specified to capture packets from a specific network interface (ie. Ethernet, Wifi, etc.). The many applications and ease of use of Wireshark make it a great tool for this project.

The process for this project (shown in Figure 1) involved setting a hostname filter for each desired website, capturing packets from the host, and then exporting the data to a CSV file.

2.1.2 Data Collection. To commence the collection of our data, we open Wireshark and select a network connection. In this project we established a connection the College of William and Mary's network interface (en0). In order to isolate the web traffic in order to specifically collect packets that we are analyzing, the capture filter functionality is utilized. The website of interest is then visited in the web browser which causes Wireshark to begin collecting the packets being exchanged. After collecting sufficient packets, the data is then converted into a Comma Separated Values (CSV). This process of data collection is performed for Blackboard, LinkedIn, and ChatGPT resulting in the creation of three distinct files.

2.2 Data Preparation and Processing

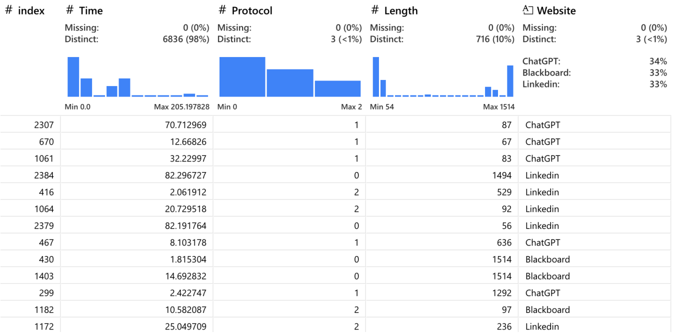
In order to ensure the integrity and reliability of our models, the collected data must be processed prior to model training.

2.2.1 Feature Selection. The data collection process using Wireshark yields a dataset containing seven distinct features shown in Figure 2.

- **Packet Number**: An integer representing the order in which the packets were captured during the Wireshark session. Provides insight on the order of the packets on the network interface.
- **Time**: A floating point value representing the relative timestamp for each packet since the start of the Wireshark session. Similar to packet number, the time feature provides a better understanding of the packet order yet is more precise.
- **Source**: A string value containing the internet protocol address of the device that the packet originated from.
- **Destination**: A string value containing the internet protocol address of the device that is receiving the packet.
- **Protocol**: A string value which contains the network protocol employed for the transmission of the packet. Three distinct protocols were identified within the datasets used in this project (QUIC, TCP and TLSv1.2).
- **Length**: An integer representing the size of a packet in bits
- **Info**: A string value containing more information pertaining to a packet. This value can vary heavily between packets.

As the objective of this project is to predict the origin of web traffic, both source and destination are excluded from the dataset used to train the models. Similairly, the "info" feature is also removed from our train data as further testing revealed that the feature led to extraneous noise in our dataset. Consequently, the features selected for training and testing the models are Time, Length, and Protocol. The Protocol feature is converted to an integer for compatibility with the models using a dictionary to map protocol names to a corresponding numerical representation (e.g., 'TCP': 0, 'QUIC': 1, 'TLSv1.2': 2) as shown in Figure 3. Additionally, a categorical variable titled "Website" is created which displays which website the packet originated (e.g., LinkedIn, ChatGPT, Blackboard). The "Website" feature acts as the target variable and allows the models to detect relationships between the packets and their corresponding web traffic origin.

2.2.2 Data Formatting. To maintain normalized representation in all the collected datasets, a process of data harmonization was applied after collecting the necessary data. The dataset with the minimum number of observations is identified and accounted for when truncating the remaining two datasets. Subsequently, the three datasets are combined and rearranged to minimize bias while training each of our models. This process ensures a balanced yet unbiased dataset that maximizes the effectiveness of our models. Figure 3 pesents the refined data following the data preparation.



# index	# Time	# Protocol	# Length	Website
	Missing: 0 (0%) Distinct: 6836 (98%)	Missing: 0 (0%) Distinct: 3 (<1%)	Missing: 0 (0%) Distinct: 716 (10%)	Missing: 0 (0%) Distinct: 3 (<1%)
	Min 0.0 Max 205.197828	Min 0 Max 2	Min 54 Max 1514	ChatGPT: 34% Blackboard: 33% Linkedin: 33%
2307	70.712969	1	87	ChatGPT
670	12.66826	1	67	ChatGPT
1061	32.22997	1	83	ChatGPT
2384	82.296727	0	1494	Linkedin
416	2.061912	2	529	Linkedin
1064	20.729518	2	92	Linkedin
2379	82.191764	0	56	Linkedin
467	8.103178	1	636	ChatGPT
430	1.815304	0	1514	Blackboard
1403	14.692832	0	1514	Blackboard
299	2.422747	1	1292	ChatGPT
1182	10.582087	2	97	Blackboard
1172	25.049709	2	236	Linkedin

Figure 3: Processed Web Traffic Data

2.3 Employed models

For this project, we required a classification model that would be able to accurately predict the website of origin for a given packet. We chose to employ three different models: K-Nearest Neighbors, Random Forest, and Neural Network. The reason we had selected more than one model was to be able to effectively compare the performance of each and determine which was the most effective for our solution. These models were chosen for their abilities to classify data and their potential to accurately predict the website of origin for a given packet. We also used a baseline model, Logistic Regression, to also compare the performance of the other models to a more simple, linear model.

2.3.1 Logistic Regression (Baseline). As a starting point, we employed the Logistic Regression model as a baseline for our data analysis. Logistic regression is a statistical method used for binary classification tasks, where the goal is to predict the probability of an event happening or not happening. It was understoof initiallly that this model may not be an ideal fit for our data, due to the regressions binary nature having more than two classifications is not ideal. However, for our purposes this was an excellent baseline. It was important for us to first visualize what features define network traffic so that we could further model around them, and the logistic regression was the most efficient choice for that. Our model returned us an accuracy score of .56, or 56%, which is less than ideal but not entirely unexpected. To analyze the data further, we put the results from our model into a confusion matrix. With this we were able to see that the model predicted

Blackboard quite well, atleast in comparison to ChatGPT and Linkedin which were far poorer.

2.3.2 K-Nearest Neighbors. K-Nearest Neighbors (KNN) is a supervised learning model which classifies data points based on their proximity to a specified number (k) of existing data points. The implementation process is as followed:

(1) **Optimization and Training**: Prior to training the model, the optimal number of neighbors (k) for the model is identified using K-Fold cross validation. This technique involves dividing the training data into multiple folds and utilizing one fold for testing while the remaining ones are used for training. For every fold, a KNN model is trained with a specified k value and the accuracy (external validation) is calculated. Multiple different k values are tested using this process and the k value that yields the highest average external validation is used in the final model. After the optimal k value is calculated, the model achieved an accuracy of 97% on the testing data.

(2) **Evaluating Model**: In order to assess the effectiveness of the model, an independent dataset containing 2000 packets was collected and evaluated. Having the model evaluate a seperate unseen dataset reveals a more realistic measurement of the accuracy of our model. After evaluating this dataset, the KNN model achieves an accuracy of 77%.

2.3.3 Random Forest. A Random Forest is a form of an ensemble model that is composed of multiple decision trees. Each tree is ran and gives its predicted classification, and the final classification is determined by a majority vote of all the trees (seen in figure 3).

Advantages of Random Forest Models:

(1) **Ability to handle complex features**: Random forests have the ability to handle complex features, such as time and protocols in the case of network packets, and the relationships between them, which can be very important in classifying network traffic.

(2) **High accuracy**: Random forests diminishes the problem of overfitting that can occur in decision trees, and can achieve higher accuracy, helping to more accurately determine website origins of a packet.

(3) **Feature importance**: Random forests can provide insight into which features are most important in the classification process, such as protocol vs time in classifying websites for this project.

(4) **Scalability**: Random forests can be scaled to handle much larger datasets, which can be very important for network traffic which can receive a high volume of packet data extremely quick.

The library used for the Random Forest model was scikit-learn [3]. Sci-kit learn's implementation of Random Forests utilize aggregation to take a soft vote of the soft classifications (probability of the packet being from each class). Within soft voting, a weighted average of the probabilities from the classification are taken, and the class with the highest probability is chosen as the final classification.

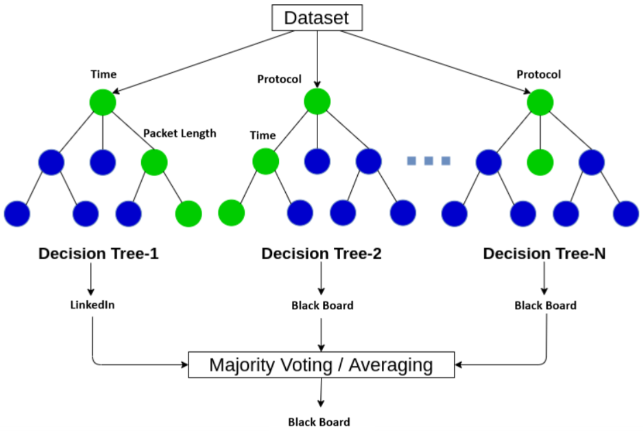


Figure 4: Random Forest model structure example

2.3.4 Neural Network.

3 EVALUATION

This section comprehensively evaluates our models and their performance on the dataset and validating sets. Later, goes on to determine the model which best classified our selected websites based on network packet data.

3.1 Dataset

3.2 Evaluation Metrics

Accuracy and Macro F1 score were the primary metrics used to evaluate each model. These metrics are applicable to all models and allow for a direct comparison for each of their performances. Two other metrics: precision and recall, help to provide a more detailed understanding of the model's performance on a class-to-class basis.

The equation for **Accuracy**:

$$Accuracy = \frac{TP + TN}{TS}$$

The equation for **Macro F1**:

$$MacroF1 = \frac{1}{n}\sum_{1}^{n}\frac{2 * Precision_i * Recall_i}{Precision_i + Recall_i}$$

The equation for **Recall**:

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

The equation for **Precision**:

$$Accuracy = \frac{TP}{TP + FP}$$

Accuracy is the score that rates the effectiveness of a model at classification overall (ie. how many it predicted correctly out of the total set).

Macro F1 is the harmonic average of the of precision and recall for every class, and then taking the average of these calculated scores across all classes.

Recall is the calculation of how many classifications the model got correct (ie. how many truly belong in that class in the entire dataset). Precision is the calculation of how many classifications truly belong to each class (ie. can be thought of as class accuracy).

3.3 Model Comparison and Evaluation Results

4 DISCUSSION & FUTURE WORK

4.1 Analysis of Results

4.2 Future Work

5 CONCLUSION

REFERENCES

[1] Wireshark Foundation. 2024. Wireshark.. https://www.wireshark.org/.

[2] Kay Boldt, Kenneth B. Kent, and Rainer Herpers. 2020. Investigation of encrypted and obfuscated network traffic utilizing machine learning. In *Proceedings of the 30th Annual International Conference on Computer Science and Software Engineering* (Toronto, Ontario, Canada) *(CASCON '20)*. IBM Corp., USA, 43–52.

[3] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[4] Hiroshi Tsunoda and Glenn Mansfield Keeni. 2012. Security by simple network traffic monitoring. In *Proceedings of the Fifth International Conference on Security of Information and Networks* (Jaipur, India) *(SIN '12)*. Association for Computing Machinery, New York, NY, USA, 201–204. https://doi.org/10.1145/2388576.2388608

[5] Jun Zhang, Xiao Chen, Yang Xiang, Wanlei Zhou, and Jie Wu. 2015. Robust network traffic classification. *IEEE/ACM Trans. Netw.* 23, 4 (aug 2015), 1257–1270. https://doi.org/10.1109/TNET.2014.2320577

[4] [2] [5] [3] [1]