EXPLORING BROWSER FINGERPRINTING

Srini Chelimilla (slc8kf), Parul Goswami (pg7wfm), Morgan Kinne (mck7py), Sai Singamsetty (sbs3ja)

INTRODUCTION

Browser fingerprinting is a pervasive method of user identification based on the unique characteristics of browsers and online behavior. This paper explores the classification of browsers and industry and analyzes machine learning models of fingerprinting techniques in Google Chrome and Mozilla Firefox using an open-source browser extension, FPMON, and a networking dataset obtained through browser fingerprinting [2]. We chose this topic due to our interest in browser fingerprinting and its implications for the future, and sought to classify upon industry or browser based on the fingerprinting techniques between the web pages visited. Browser fingerprinting is a relatively new privacy concern and has a serious impact on the relationship between consumers and the web pages they access and trust. This relationship is ever-changing, as webpages view their users as sources of information and therefore, profit. Despite increasing user awareness around privacy concerns, browser fingerprinting remains a prevalent issue on the majority of web pages, as it can bypass privacy tools users employ to protect themselves, such as entering "private" browsing. The primary objective of this project was to see if we would be able to classify industries of various websites dependent on various browser fingerprinting features that include network traffic metrics and JavaScript attributes.

Problem Statement

Browser fingerprinting is an accurate method of identifying users, posing privacy concerns. Our project seeks to develop machine learning models based off of collected and found browser fingerprinting technique datasets, where our classification task is to classify the industry a website is a part of. The need to address browser fingerprinting arises due to its widespread use across the web, threatening user privacy. Browser fingerprinting persists as a problem due to its accuracy and prevalence. As detailed by the team that developed FPMON [2], every browser executes embedded scripts, allowing the collection of device-specific data, and creating a digital fingerprint. This persistence highlights the urgency to address the growing threat of browser fingerprinting. Our paper will discuss our approach towards collecting browser fingerprinting data, analyzing already given browser fingerprinting data, developing machine learning models to perform our classification task, and analyzing said models for performance. We utilized FPMON to monitor JavaScript functions being tracked on web pages we visited, alongside a dataset of network traffic gathered through browser fingerprinting.

APPROACH

General Approach

At the beginning of this project, our team sought to predict the browser based on the fingerprinting patterns we would be able to analyze within our dataset. Our approach began with a search for viable datasets, and this led us to find the Wintermute dataset [5], which was compiled as part of the Wintermute Project. This was part of a research project that sought to address the complexity of network traffic and how browser fingerprinting can be used to extract that traffic without directly interfering with it. As network traffic is considered very sensitive data, the team was trying to further understand how browser fingerprinting exposed that data. This research project was done by the team (Katharina Dietz, Michael Muhlhauser, Michael Seufert, Nicholas Gray, Tobias Hoßfeld, Dominik Herrmann) studying privacy protection mechanisms in differential privacy. The dataset contains networking data captured during browser fingerprinting on popular websites and was a good first lead on our project. We then began to analyze and construct models of this networking dataset but realized that we would need a supplementary dataset to have a better understanding of browser fingerprinting raw data rather than processed data, and continued our search.

This led us to find the FPMON browser extension. The FPMON browser extension, developed and publicly released as part of a study by Julian Fietkau, Kashyap Thimmaraju, Felix Kybranz, Sebastian Neef, and Jean-Pierre Seifert, was employed to record monitored JavaScript functions during webpage visits. The extension was released publicly to empower users against browser fingerprinting threats. FPMON works by injecting scripts into the webpage to track what Javascript features are being monitored by each webpage. Then this information is grouped and returned to the user in the browser extension. The extension contains raw information on what kinds of data are being recorded by the browser for each user that visits a certain website. As we analyzed the FPMON dataset, we decided to classify the industry rather than browser because we were interested in feature engineering our datasets. In this context, industry refers to the industry associated with the popular websites that data was collected from in the Wintermute dataset. A few examples of industry in this context are e-commerce, social media, and news. The FPMON dataset was compiled by analyzing the Wintermute project dataset and identifying the top websites present in that dataset. Once the top websites had been identified and classified by industry, our team began accessing those websites and manually compiling the dataset based on the information returned by the FPMON extension.

We then wrote a script (FPMON_Convert.py) to make the FPMON dataset follow the formatting of the cleaned Wintermute dataset so that we could combine the two and develop models of the collective data. The FPMON_dataset.csv is our initial data collection of FPMON extension data for various websites on Firefox and Chrome. The script takes this dataset and splits it into two rows per website (one row for Firefox data and another for Chrome data), keeping the basic identifying info for both rows. We then made a new dataframe that in essence one-hot encodes each of the textual features present for all the rows so that we can run our ML models. We exported this dataframe into the cleaned_FPMON_Dataset.csv. Then began our classification and final analysis.

Dataset Approach

We use four datasets for classification and analysis: 1) Wintermute 2) Wintermute feature engineered 3) FPMON 4) FPMON + Wintermute feature engineered combined. The Wintermute and FPMON datasets are two entirely different datasets that capture different sorts of data related to browser fingerprinting.

The Wintermute dataset has a dimensionality of 4495 x 146. The columns contain different networking information of browser fingerprinting data transfer such as packets, flow, burst, duration, and entropy. This was our first dataset and we classified on the browser variable. We then wanted to find out if different industries use different types of browser fingerprinting techniques. Because it was impractical to find and label the industries of each of the 4495 websites, we decided to pick 50 entries, that are distributed over the five different industries we identified. Out of the 50 websites, we grouped 10 websites for each of our five industries: e-commerce, news, tech, streaming, and social media. We then found the entries of those 50 websites in the Wintermute dataset, and each of the 50 websites had data recorded around seven to ten times, depending on the website. After performing feature engineering on the Wintermute dataset, the dimensionality of the new dataset was 475 x 147. The rows in this new dataset have all the same columns as the Wintermute dataset in addition to a new industry column.

Regarding our modeling and analysis of the FPMON dataset, 40 raw features were tracked across all websites, and the features that the website does record are included in the extension popup for each website. Some features include geolocation, app version, permissions, device memory, cookies enabled, etc. For a set of 50 websites decided upon based on the Wintermute dataset, we navigated to each of those websites, once in Chrome and once in Firefox, and gathered the raw data using the browser extension. We compiled our findings into a new dataset and wrote a Python script to properly format the data so we could run machine learning models on it.

Finally, we decided to merge the Wintermute feature-engineered dataset and the FPMON dataset to see if the network and raw features combined would yield more comprehensive findings.

Classifiers Approach

As mentioned in the previous section, we are trying to predict a company's industry based on the fingerprinting techniques its website uses. Because industry is a discrete variable with five different values, we needed to choose a multi-class classifier. We wanted to run several classifiers on our datasets because we were curious to see how our data is connected and what aspects of each model affect the accuracy of our predictions. We chose five different classifier models to run on each of our four datasets: 1) Random Forest classifier 2) K Nearest Neighbors classifier 3) Gaussian Naive Bayes classifier 4) Neural Net classifier 5) Gaussian Process classifier [1]. The justifications on why we chose these specific models are provided below:

Random Forest is well-suited for large datasets with numerous features, making it ideal for scenarios with extensive browser fingerprinting data. As it is also inherently capable of handling multiclass classification, random forest is a natural choice for categorizing websites into different industries. It also performs well with sparse datasets, due to one hot encoding, which is common in browser fingerprinting where features might not be uniformly present across all websites.

K Nearest Neighbors has potential for high accuracy, which makes it very suitable for the precise classification of websites based on patterns recognized in browser fingerprinting. Like random forest, KNN is also inherently capable of handling multiclass classification and therefore is useful for categorizing data points into different industries. KNN is also very versatile, working with linear and nonlinear patterns, and since there are some unknown patterns present in the dataset, this is a valuable capability. It also works well with noise-free data and is therefore suitable considering the cleaning work done on this data.

Naive Bayes assumes independence of features, which aligns with the relationship between browser fingerprinting features. It's also not sensitive to irrelevant features, which can be beneficial when dealing with a diverse set of fingerprinting characteristics. Naive Bayes Gaussian tends to not overfit, which provided a stable solution for the classification task within the Wintermute dataset, but because of the limited size of the FPMON dataset, we recognized a large amount of overfitting for this model. This classifier also offers faster solutions than logistic regression, which is very advantageous for a large dataset.

Neural Networks are well-established for supervised learning tasks, making them suitable for the goal of industry classification based on fingerprinting techniques. Neural Networks also make no assumptions about the probability distribution, which allows them to adapt to various data patterns without strict constraints. They're also capable of fitting nonlinear models, especially when they have hidden layers, which provides the flexibility to capture complex relationships in the data. Neural networks also don't overfit, as they incorporate regularization techniques to ensure generalization to hidden data.

With the Gaussian Process, you can shape prior beliefs using the kernel parameter allowing you to shape the fitted function in many ways [3]. One of the biggest reasons we decided to explore the classifier is because it can be used for multi-class classification. The classifier has a default approach for multi-class by using a one-versus-rest approach. This means that it takes the dataset and turns it into several binary classifications. The classifier also has internal optimization, for example, the kernel hyperparameters are optimized during fitting and a default optimizer fmin_l_bfgs_b is used [4].

DISCUSSION AND ANALYSIS

Our dataset includes information on browser fingerprinting and networking details. By modeling and analyzing this data, we aim to unveil patterns and relationships that can contribute to the classification of the industry within our collective datasets. By unveiling these trends and patterns we hope to get a better understanding of what ties differing and similar industries in terms of browser fingerprinting techniques.

As we mentioned above we ran all five classifiers for each dataset to see how the different models affected the training and validation accuracies. However, when performing our analysis in comparing the datasets we wanted to select the 'best' classifier for each dataset to use as the basis for our comparisons. We couldn't simply use validation accuracy because it was a general metric that didn't take into account how our model classified each industry. We decided to add up all the F1 scores for each classifier and compare that sum across all classifiers for each dataset (with the equation given below). A model performed well if it was close to the max possible score for that dataset.

$$\begin{aligned} CalculatedMetric \; = \; & F1Score_{RandomForest} \\ & + \; F1Score_{KNN} \\ & + \; F1Score_{NaiveBayes} \\ & + \; F1Score_{NeuralNet} \\ & + \; F1Score_{GaussianProcess} \end{aligned}$$

Here is a table containing F1 score sums for each classifier across each dataset.

	Random Forest Classifier	KNN	Naive Bayes	Neural Net	Gaussian Process	Max Possible Score
Wintermute	**1.86**	1.81	1.71	**1.88**	1.96	2*
Wintermute Feature Engineering	**4.66**	4.48	4.49	3.86	3.79	5
FPMON	3.27	2.93	2.28	**3.66**	2.73	5
Combined Dataset	**4.65**	4.2	3.17	4.38	3.18	5

**The max possible value for this dataset is 2 because we were interested in classifying browser rather than industry at this point because this is our dataset with no feature engineered features (industry is not a column in the dataset).*

When evaluating which classifier performed the best per each dataset, we maintained a threshold of +/- 0.02. We found it important to distinguish how models could evaluate slightly differently in our chosen sum metric based on variability in hyperparameter tuning. From there, we found the following results.

- Wintermute: Random Forest Classifier, Neural Net
- Wintermute Feature Engineering: Random Forest Classifier
- FPMON: Neural Net
- Combined Dataset: Random Forest Classifier

Wintermute Dataset

We found that different classifiers performed the best for different datasets. For the Wintermute dataset, although the Gaussian Process classifier had the highest score, the training accuracy is extremely high, leading us to believe that the model is overfitting. So, while the model does perform well on our test set, we believe this will not hold for general classifications outside of our dataset, which is why we decided to not choose this classifier as the “best” one. Given our +/- 0.02 threshold, we concluded that the Random Forest and Neural Net classifiers performed the best for this dataset. This makes sense as the Random Forest classifier is inherently multi-class, and does well with sparse datasets. Given that the dimensionality of this dataset is quite large (4495 x 146), it makes sense that the Neural Net classifier also performed well since this classifier works well on large datasets.

These are the two confusion matrixes for the random forest classifier (Fig. 1 on the next page) and the neural net classifier (Fig. 2 on the next page) displaying the proportion of correctly predicted to incorrectly predicted browsers for the Wintermute project dataset.

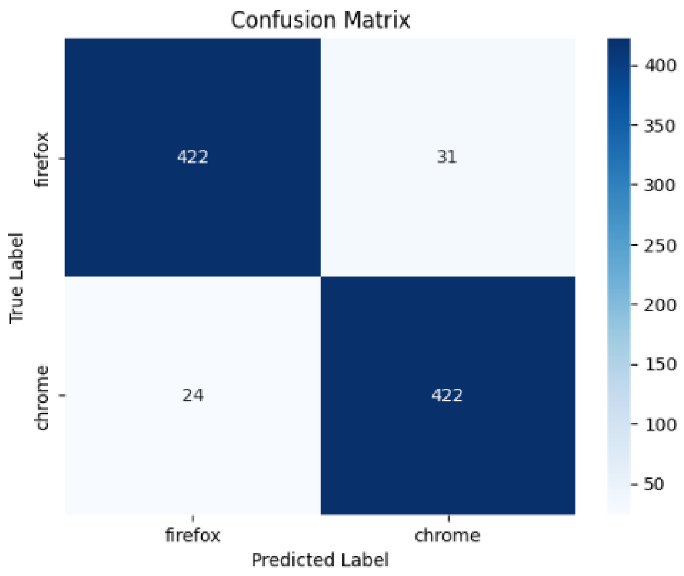

Figure 1: Random Forest Matrix

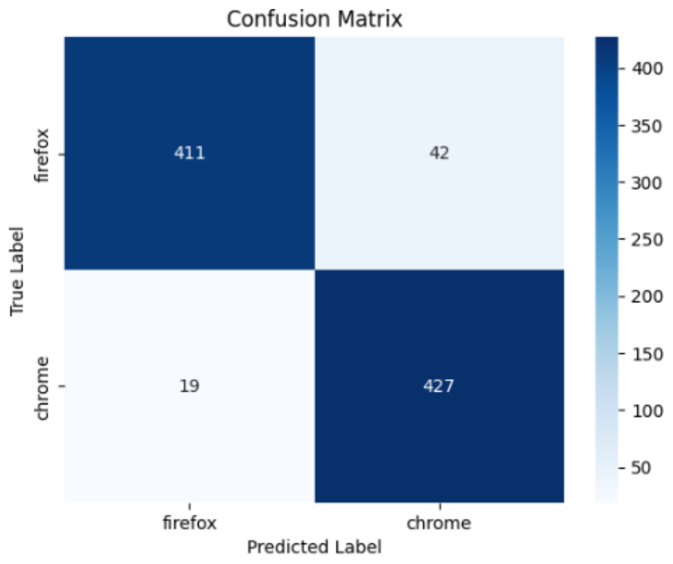

Figure 2: Neural Net Matrix

Wintermute Feature-Engineered Dataset

For Wintermute Feature Engineering, the Random Forest classifier worked the best, which makes sense as it was one of the better performers for the past dataset.

We would normally expect that the same models win across WinterMute and Wintermute FE, however, we must acknowledge that we are classifying a different attribute across the two (browser vs industry), and introduce a new feature (industry column) as well to account for slight differences we see in our expected results.

This is the confusion matrix for the random forest classifier (Fig. 3) displaying the proportion of correctly predicted to incorrectly predicted industries for the Wintermute Feature-Engineered dataset.

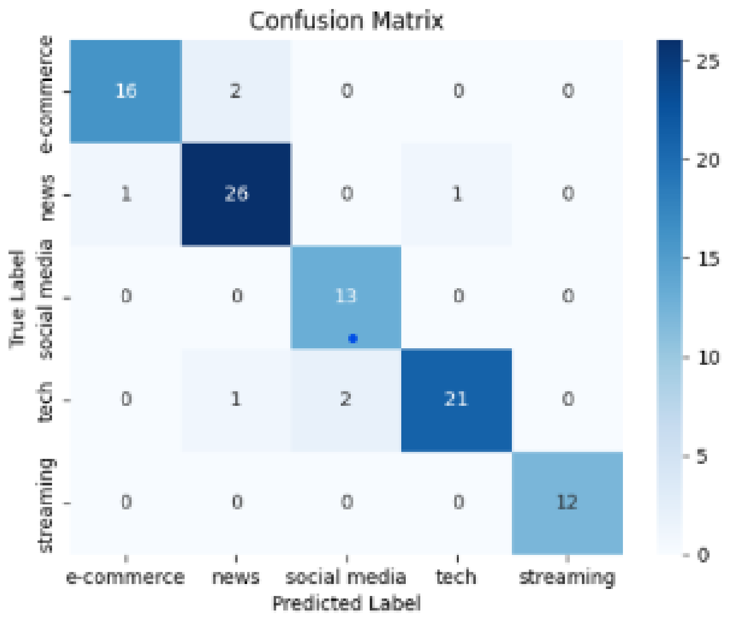

Figure 3: Random Forest Matrix

FPMON Feature-Engineered Dataset

For the FPMON dataset, we found the Neural Net classifier to work the best. Even though Neural Net classifiers tend to perform the best with large amounts of training data, it can generalize well even on smaller training data sizes. The FPMON dataset has the smallest dimensionality (100 x 45) compared to the other datasets. Therefore, whereas the other models require larger amounts of data, Neural Net adapted well to the small size, outperforming the rest.

This is the confusion matrix for the Neural Net classifier (Fig. 4) displaying the proportion of correctly predicted to incorrectly predicted industries for the FPMON Feature-Engineered dataset.

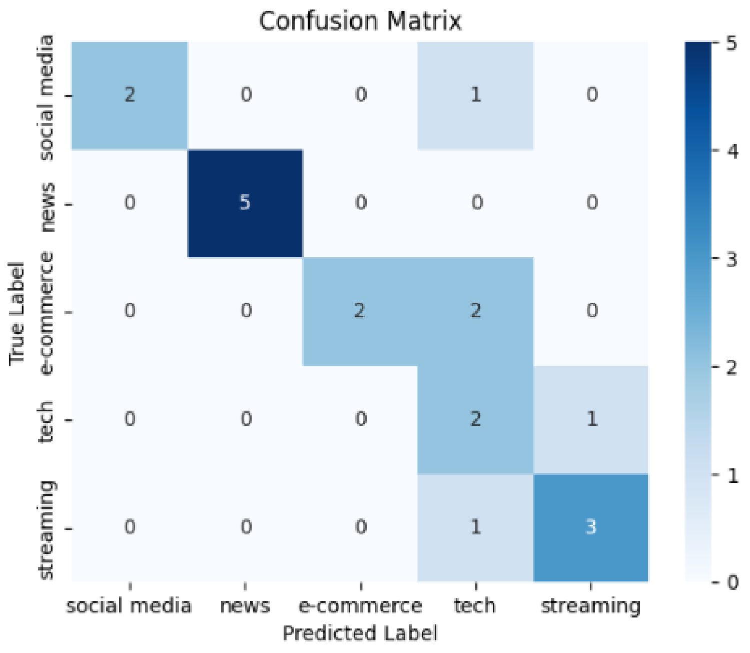

Figure 4: Neural Net Matrix

Combined Wintermute & FPMON Dataset

For the Combined dataset, Random Forest performed the best. This makes sense as this classifier is inherently multi-class, and does well with sparse datasets. This dataset contains several features that are sparse from the merging of FPMON (due to one hot encoding). Each of the forty browser fingerprinting features contained in the FPMON dataset was made into its own column. Furthermore, Random Forest generally does well with highly dimensional datasets, where the Combined Dataset holds the most features at 276 features.

This is the confusion matrix for the random forest classifier (Fig. 5) displaying the proportion of correctly predicted to incorrectly predicted industries for the combined dataset.

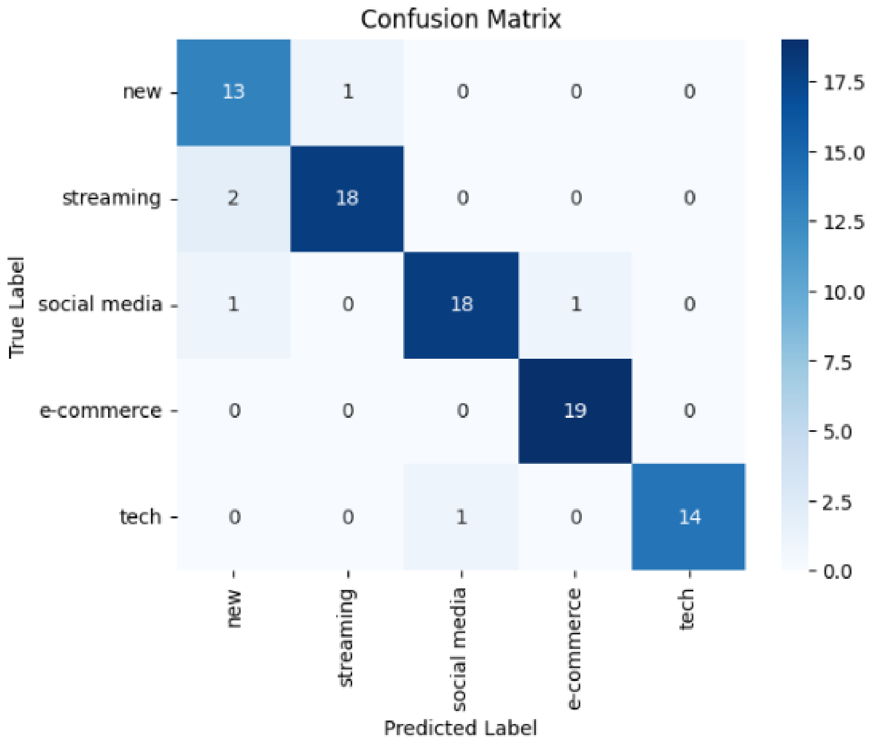

Figure 5: Random Forest Matrix

CHALLENGES

We faced several challenges during this project, each requiring careful consideration and different approaches.

One of the initial challenges we encountered was related to the datasets we needed to use. After looking through the Wintermute dataset, it became evident that additional raw data would be needed to perform a comprehensive analysis. Upon a recommendation from Prof. Sun, we attempted to use the Princeton Web Census dataset. However, we encountered several issues with using this dataset in Google Colab, our collaborative working platform, due to its substantial size. There are several limitations on the volume of data that could be stored and processed within Google Colab as well as on our own personal computers, which led us to reevaluate our dataset choices and discover FPMON. The code to retrieve and read the Princeton dataset is still included in our Google Colab submission to showcase our attempts at retrieving the data.

We also faced some difficulty in gathering the FPMON data initially and storing it a manner that would be appropriate for running our ML models. Reformatting our manual collection dataset was challenging as we had to understand our data and the best way to display it for modeling and create a script that did that. Since we gathered the initial data manually, there was of course user error in the consistency of certain formatting, therefore we ran into some problems in using our FPMON_convert.py script, but after carefully looking through the manual dataset and fixing some formatting errors, we were able to run the script successfully.

During the modeling and analysis phase of our project, we recognized an issue of overfitting in our data, which occurs when a model captures noise in the training data rather than the patterns. While we did some tuning to help minimize overfitting, we recognized that if given more time we could do more comprehensive hyperparameter tuning.

Referencing our shift from prediction to classification, we recognized a challenge when looking at our variables and discovering that our variable of interest, the labels for our model, was not a continuous variable. We originally did not realize the continuous variable constraint for regression tasks, so we pivoted to classification. Our past attempts at regression models are included in the Google Colab to showcase this challenge and our past work.

We also encountered an issue in determining the most suitable model for our classification task. We wanted to evaluate each model's performance in predicting the distinct characteristics of websites with the five industries under consideration. We incorporated F1 scores, which take into account precision and recall and consider false positives and negatives, which allows for a better understanding of our findings. By prioritizing the highest summation F1 scores across the industries, we were able to gauge the accuracy and precision of each model. This produced a good measure of performance and reinforced the relevance of our findings.

CONCLUSIONS

The primary objective of this project was to see if we would be able to classify industries of various websites dependent on various browser fingerprinting features that include network traffic metrics and JavaScript attributes. In order to achieve this objective we gathered data, cleaned it, ran various machine learning classification models on the datasets, and analyzed the results. From looking at the best models featured in our analysis above, we can look at those models' validation accuracies as a means to understand how well we can classify various browser fingerprinting features by industry. Specifically, we believe that the network traffic features (from the Wintermute dataset) are better predictors of industry compared to the javascript features we gathered using FPMON. This is because the validation accuracy for the Wintermute feature-engineered dataset was 92%, on the other hand, the validation accuracy for the FPMON feature-engineered dataset was 73%, while both train accuracies were relatively high. The combined dataset had a validation accuracy of 93%. Therefore, we determined that the network traffic contributed more to the accurate classification of industries since it had a much higher accuracy.

Being able to see if fingerprinting features can classify industries helps us understand that there could be a correlation between what techniques (features) are used by different industries. While we didn't explore specific features, we can see the general idea that network traffic features are the better predictor. This sets up the notion that users can understand a general idea of what fingerprinting data is being assessed based on the industry of the website and this sets up the foundation for future work in digging deeper into specific features.

Future Works

As mentioned above, we believe that this project sets the foundation that based on the industry you can get a general idea of what fingerprinting features are being assessed, since fingerprinting features overall can be used to classify industries of websites. Therefore future work on this topic would be to take a deeper look into the specific features that contribute to industry classification instead of the multi-class classification we did for this project. More work could also be done on this front by looking at new and emerging browser fingerprinting techniques, especially in the context of industries that make their main profit from data collection and whether or not they use browser fingerprinting. As the modern internet changes, future work could focus on developing privacy tools that adapt to emerging fingerprinting patterns. This could involve regular updates to privacy tools to properly handle changes in web technologies and new fingerprinting techniques.

This research could be expanded to include user-centric privacy assessments, which refer to evaluating privacy concerns and protections from the perspective of individual users. This work could investigate the impact of browser fingerprinting on individual users, rather than in the context of this work. This could provide a more comprehensive approach to understanding the implications of browser fingerprinting for all users. Another way to improve our understanding of this classification process would be to incorporate additional industry categories. This could involve identifying and classifying websites in emerging sectors and those with unique fingerprinting characteristics not covered in the initial analysis. Another facet would be to further develop the FPMON extension to contribute better to privacy protection by including a feature that allows users to manually block the monitoring of their browser features. This project could also be expanded to work with privacy advocacy groups, which could involve sharing insights and working toward practical solutions for users. By addressing these future avenues, this project could evolve beyond the current scope of a school project and have a real impact on the ongoing discussions around browser fingerprinting and user privacy.

REFERENCES

In general, sci-kit-learn documentation is not cited unless quoted.

[1] Brownlee, Jason. “Gaussian Processes for Classification with Python.” MachineLearningMastery.Com, 2 Aug. 2020, machinelearningmastery.com/gaussian-processes-for-classification-with-python/.

[2] “A Fingerprinting Monitor For Chrome.” FPMON, fpmon.github.io/fingerprinting-monitor/. Accessed 13 Dec. 2023.

[3] Knagg, Oscar. “An Intuitive Guide to Gaussian Processes.” Medium, Towards Data Science, 15 Jan. 2019, towardsdatascience.com/an-intuitive-guide-to-gaussian-processes-ec2f0b45c71d#:~:text=Gaussian%20processes%20are%20a%20powerful,estimate%20of%20their%20own%20uncertainty.

[4] scikit-learn-developers. “Sklearn.Gaussian_process.Gaussianprocessclassifier.” Scikit, scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.GaussianProcessClassifier.html. Accessed 13 Dec. 2023.

[5] “Wintermute Project.” GitHub, github.com/wintermute-project/private-browserfingerprinting/tree/main/data. Accessed 13 Dec. 2023.

[6] Fingerprinting the Fingerprinters: Learning to Detect ..., arxiv.org/abs/2008.04480. Accessed 14 Dec. 2023.