Introduction

Team Name: Neurotic Networkers

Team Members:

Name	Student ID	Email
Jordan O'Neill	IES24181	jo-neill@hotmail.com
Yan Renyu	IES24364	ryan022@e.ntu.edu.sg
Eloïne Vatteville-Réveillon	IES24309	eloine.vatteville-reveillon@insa-lyon.fr
Roselyn Mui Huynh	IES24362	r23huynh@uwaterloo.ca
Qin Tong Bettina Tee	IES24303	btee003@e.ntu.edu.sg
Jennifer Zhang	IES24624	j2348zha@uwaterloo.ca
Kei Saito	IES24601	s.kei_824818@keio.jp

Task Assignment:

For the closed-world experiments, the objective is to classify the 95 monitored websites.

For the open-world experiments, two goals will be achieved using binary classification and multi-class classification. In both cases, monitored website instances are treated as positive samples, and unmonitored website instances are treated as negative samples.

Project Timeline:

Thursday 17th October → Plan and construct proposal
- Proposal planning
- Methodology timeline and project overview

Monday 21st October → Finalize proposal for submission
- Incorporate feedback on proposal draft
- Proposal review and polishing

Wednesday 30th October → Start model prototyping
- Construct and test classifier architectures
- Extract and analyze important features

Wednesday 06th November → Continue model development
- Refine and Implement learning models
- Implement hyper parameters

Wednesday 13th November → Progress testing and evaluation
- Enact evaluation metrics and error analysis
- Tune hyperparameters

Wednesday 20th November → Project refinement
- Validate and iterate on model architecture including models and hyperparameters

Wednesday 27th November → Presentation preparation
- Finalize models and documentation
- Develop presentation slides

Thursday 05th December → Project presentation
- Present project

Problem definition

Website fingerprinting is the process of identifying a website based on characteristics of network traffic without knowing the user or payload. We have been provided the following data: time stamps, and direction * size. This data reflects monitored (closed-world) and unmonitored (open-world) websites.

From this metadata we will design and create a machine learning model that will accurately classify the 95 monitored websites in the closed world scenario. We also aim to achieve a binary classification and a multi-class classification for the open world scenario.

To evaluate the model on the closed world scenario, we will split the data set into training and testing portions and test its ability to accurately classify the monitored instances. For the open world scenario, the binary classifier will determine if the instance belongs to a monitored website or not. Finally we will validate that the multi-class classifier can properly categorize the websites.

Data Analysis and Pipeline

Data Overview

We are given two dataset files. The first file contains data from “monitored” websites, with 19000 instances from 95 websites, each with 10 subpages that have 20 observations each. The second file contains data from "unmonitored" websites with 10000 instances.

For each sample, the raw data we have consists of a list of timestamps at which packets were sent or received, that are positive or negative depending on reception or emission. In the given sample code this data is converted into arrays. For the data from “monitored” websites, arrays obtained are X1, X2 and y, representing the timestamps, direction (positive or negative, multiplied by 512 to have the size of the payload) and site of the instance (represented by a number from 1 to 95). For the data from “unmonitored” websites, arrays obtained are only X1 and X2, as we will put y to always be -1 (negative sample in an open world context) once we format our data.

Data Preprocessing

To prepare the data for training and testing of models, we will be preprocessing them as follows:

- Outlier detection:
  - Outliers can significantly affect the performance of our models, especially models that are sensitive to large deviations. Checking for outliers will help to ensure that our models are not influenced by anomalous data points and result in skewed results.
  - We will be using the following methods to test for outliers: Z-Score, Interquartile Range

- Normalisation, Scaling, Transformation:
  - The distribution of the data can affect the training and development of the models. Shaping our data to a more interpretable form can improve the models performance.
  - We can use the following methods to scale our data if necessary: Min-Max Scaling, Z-Score Standardisation (StandardScaler)

Feature Engineering

In order to better construct our models to accurately classify the websites, we will design a certain number of features from the raw data that can be more comprehensible.

We will be choosing features in reference to the ‘Introduction to course project’ lecture slide 15, which presents the most significant features to exploit found by a research project on the subject.

Additionally, we might explore the use of more features for the multiclass models, making use of the fact that we have samples for different web pages for each monitored website. For example, in the context of having different webpages for the same website that might behave differently, especially in terms of concentration of packets or number of packets as the content of a page might be ‘heavier’, studying more features based on standard deviation calculations could be interesting.

Training and Testing Data

We will try to make sure that the training and testing sets maintain the same distribution of class labels as the original dataset, especially in the context of multi-class.

We plan on starting with using a 70:30 ratio for the splitting of both the monitored and unmonitored website samples, as the dataset is pretty big. This might evolve as we evaluate the performance of the model and see if there is a need to prevent overfitting or underfitting.

Data Pipeline

Data Preprocessing (on raw data) → Feature engineering → Data Preprocessing (on features) → Feature Selection → Model Training → Model Tuning → Model Evaluation

Candidate features

We categorize the top ten most important features from the ‘Introduction to course project’ lecture slide 15 into 4 distinctive groups below.

- Traffic Volume Feature (Packet count)
  - Number of incoming packets
  - Number of outgoing packets
  - Total number of packets

- Traffic Ratio Feature (Packet count)
  - Number of outgoing packets as a fraction of the total number of packets
  - Number of incoming packets as a fraction of the total number of packets

- Traffic Order and Timing Feature (Relative timestamps)
  - Standard deviation of the outgoing packets ordering list
  - Average of the outgoing packet ordering list

- Concentration and Burst Feature (Relative timestamps)
  - Sum of all items in the alternative concentration feature list

Within each of the 4 groups, the features are expected to be highly correlated. Hence, we will be selecting 1 or 2 features from each of the 4 groups to reduce replication of features.

The first group is data that is related to packet traffic volume and the second group focuses on the ratio of incoming/outgoing packets. We hypothesize that these features will be the most important since packet traffic data will be key in website classification and specifically, incoming packet data will likely be a key feature since it is indicative of a website's output. Within these 2 groups, multiple pairs of features are heavily correlated and we expect to not require all 5 features to have a good assessment on the packet traffic of the websites. Therefore, we expect to select 2 features from the combination of group 1 and 2 based on the results of our feature importance analysis.

The third and fourth groups are derived from relative timestamps. The third group highlights the distribution and range of outgoing packets in the traffic. Due to the use of multiple relays, we expect the order of outgoing packets to be more irregular. Hence, the distribution feature could have significant importance. On the other hand, the fourth group focuses on the sequence of bursts. Tor traffic often has bursts of packets due to relay-induced delay, where accumulated data is sent at once. This may cause bursty traffic that will impact the value retrieved from this data, however, it is still a key feature that is indicative of the concentration of user interaction with the website.

Model Introduction

For this project, we will use Random Forests for all three classification tasks: closed-world multi-class classification of monitored websites, open-world binary classification between monitored and unmonitored websites, and open-world multi-class classification of monitored websites. Random Forests are chosen for their ability to handle large datasets with complex, non-linear relationships while reducing overfitting. Using the same model for all tasks allows for clearer comparisons and consistent learning outcomes.

In the closed-world multi-class classification of 95 monitored websites, Random Forests can model complex relationships effectively. The ensemble of decision trees in this model ensures robustness against overfitting while maintaining high accuracy, making it a great fit for large datasets.

For open-world binary classification, Random Forests excel at distinguishing between monitored and unmonitored websites. The ensemble method provides strong performance with minimal tuning and efficiently processes the large dataset, making it ideal for this binary task.

Similarly, for the open-world multi-class classification, Random Forests handle the complexity of classifying monitored websites while identifying unmonitored websites as a separate class. Its ensemble approach allows the model to generalize well across multiple classes while reliably recognizing unmonitored websites as negative samples.

Other models covered in class, such as SVM and k-Nearest Neighbours, are less efficient for larger datasets due to high computational costs. Logistic regression and Naïve Bayes struggle with non-linear relationships and make invalid assumptions about feature independence, while Clustering is not suited for labeled data. Decision trees alone are prone to overfitting, and Neural Networks introduce unnecessary complexity. Overall, Random Forests offer the best balance of scalability, accuracy, and generalization for all three tasks.

Methodology

For the training and selection process, we will first split our data into training and testing sets. Once we have trained our model using the training set, we plan on proceeding with hyperparameter tuning to optimise performance. This involves adjusting parameters like learning rate, regularization strength, and other model-specific settings to find the best combination that maximizes predictive accuracy while keeping overfitting low. After optimizing the model, we will use cross validation to ensure that our model is not overfitted on any specific subset of the data. Our evaluation criteria will include metrics such as prediction accuracy, precision and recall. We will then repeat the process for each scenario to obtain three differently tuned models. Finally, we will compare the performance between the models for each experimental scenario.