Deployment

Szymon Kacprzyk

June 2025

1 Introduction to the Application

This supplementary section was made to introduce and explain the developed lightweight desktop application. That was created with a purpose of helping Nautilus OT accurately fingerprint devices, based on the thesis findings. It is a simple graphical interface that allows users to upload network datasets, train a classification model and automatically assigns categories to devices.

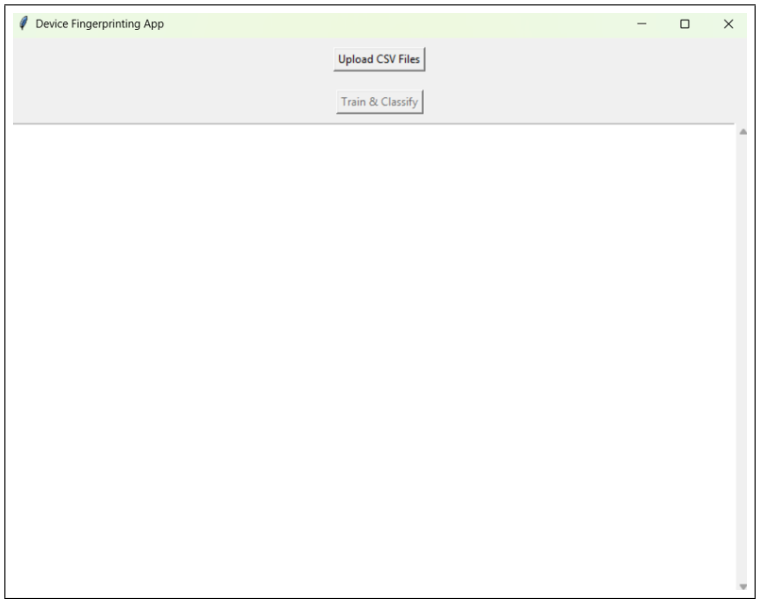

Figure 1: Main window of the application before data upload.

The internal part of the application utilizes a Random Forest Classifier from scikit-learn python's module with default parameters. It is suitable for non-technical users and quick device type modifications within the dataset.

1.1 Input Format Requirements

For the classification to work there is a required format of the datasets. The .csv file corresponding to the Asset Table must include a new column named "TRAINING". It is used to determine what devices will be used for the training. The desired goal is to manually verify and label a subset of devices in a way that each class is represented in a balanced way. Those verified devices must then receive value 1 in the "TRAINING" column, whereas those without verification that require classification with value 0.

- Devices marked with `1` will be used for training the model.
- Devices marked with `0` will be classified using the trained model.

As proven in the main thesis body a relatively small sample of devices used for training can later yield outstanding results.

After including "TRAINING" column there is one more requirement before uploading. For correct matching the datasets to internally used structures, a naming convention must be followed. The expected naming schema is:

- `***_assets.csv` → assigned to the Asset table
- `***_asset_properties.csv` → assigned to Properties
- `***_asset_relationships.csv` → assigned to Relationships
- `***_asset_relationship_services.csv` → assigned to Relationship Services
- `***_asset_relationship_service_metrics.csv` → assigned to Service Metrics
- `***_asset_types.csv` → assigned to Asset Types

Where "***" represents the name of a given network dataset. Files named differently will not be correctly parsed, causing error during training.

1.2 Example Use Case: Swiss Tools Dataset

To present how the application works, a Swiss Tools dataset modified by Nautilus OT was used. The company manually verified 150 devices. Those entries received a value 1 in the "TRAINING" column and were used in the model training. All remaining devices (value 0 in "TRAINING") will receive an assigned category label based on the trained model.

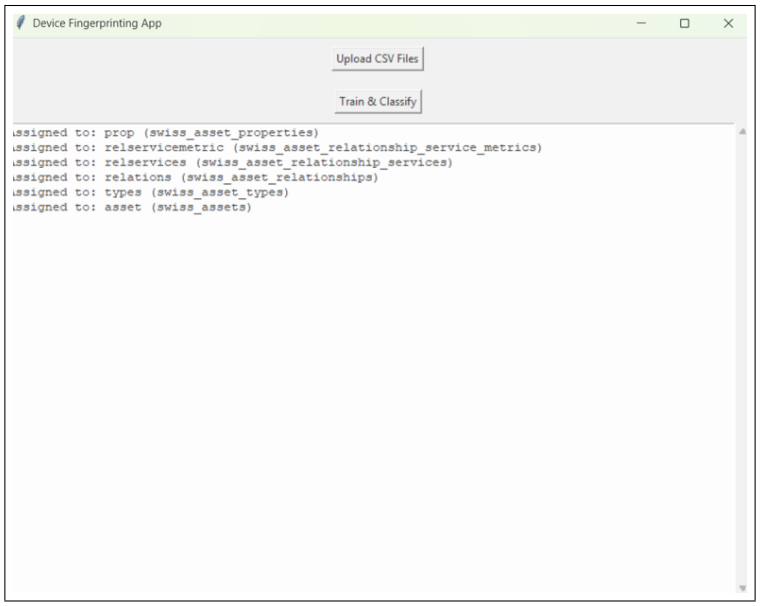

Figure 2: Main window of the application after data upload.

1.3 Outputs

Once all .csv files for a specific network are loaded, the user can press the "Train & Classify" button. The application then internally trains a model, creates a new column called "result" in a merged dataset and assigns there a predicted category for each asset. Devices used for training will receive unchanged category label in "result". The following things are returned:

- New merged .csv file with the "result" column containing fingerprinted categories
- Count of devices in each category
- Histogram demonstrating category distribution

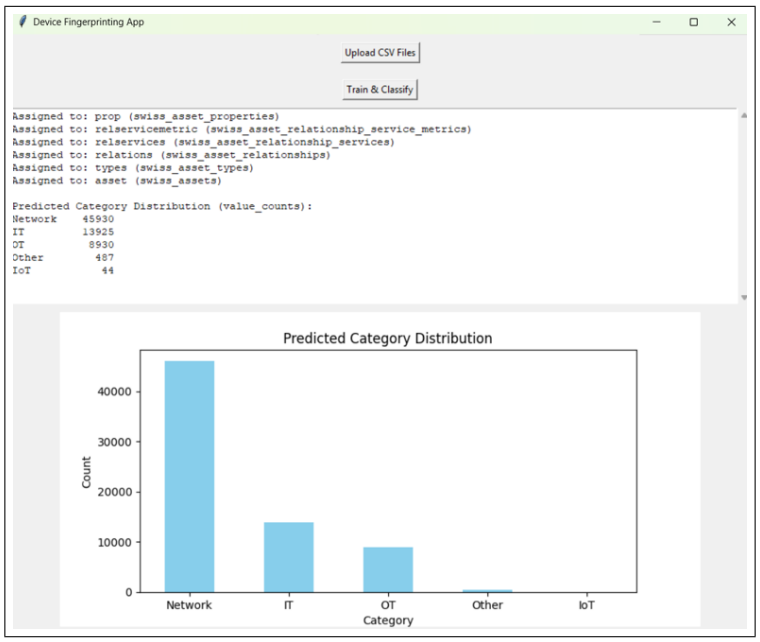

Figure 3: Main window of the application - output

1.4 Instruction and Overview

The instruction and workflow of the application can be described by the steps below:

- Launch the application
- Ensure "TRAINING" column is added and correct naming schema is followed
- Press "Upload CSV Files" and load the dataset
- Verify the text to confirm correct file assignment
- Press "Train & Classify"
- Save new .csv file and analyze histogram and class counts

This GUI provides a simple and practical foundation for further development of device fingerprinting system. It allows Nautilus OT to efficiently classify devices in any network based on a verified sample.