WiFi Traffic Classification Report

Luca Moriondo

This document is a report extracted from the WiFi Traffic Classification project GitHub page. For more information, please refer to the project's GitHub repository at this link.

Capture Processing

I personally found it easier to convert the whole Wireshark capture file into a format that can be processed by pandas easily, in particular `.csv`. To do so, it's possible to use the tshark command-line tool that comes with Wireshark. The following command can be used to filter data and copy various information from a `.pcap` or `.pcapng` file to a `.csv`:

```
tshark -r ./private/capture.pcapng \
  -Y "wlan.addr == XX:XX:XX:XX:XX:XX && wlan.fc.type=2" \
  -T fields -E header=y -E separator=, -E quote=d \
  -e frame.time_epoch \
  -e frame.len \
  -e wlan.sa -e wlan.da -e wlan.ta -e wlan.ra \
  -e wlan.fc.type -e wlan.fc.subtype \
  -e wlan.fc.pwrmgt \
  -e radiotap.dbm_antsignal \
  -e wlan.qos.priority \
  -e wlan_radio.snr \
> ./private/traffic.csv
```

Classical Analysis

The first part of this project focuses on classical traffic analysis without machine learning.
The script src/plots.py processes the CSV exported with tshark and computes several statistical features of Wi-Fi traffic for a given station MAC address.

Specifically, the script:

- Splits the traffic into fixed-size time windows (default: 15 seconds).
- Separates uplink and downlink packets.
- Computes per-window features.
- Extracts the Power Management bit to determine when the station is in sleep or awake mode, highlighting sleep periods in the plots.
- Produces time-series plots of all the above features, saved as .png images in the data/ folder.

This step provides a feature extraction and visualization pipeline that will serve as the foundation for the next stage, where machine learning techniques will be applied to classify user activities from encrypted Wi-Fi traffic.

Machine Learning Approach

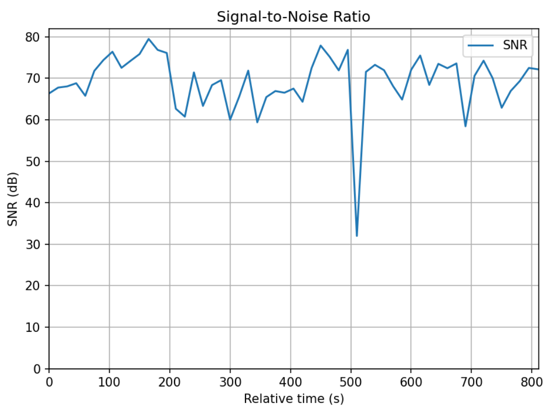
The machine learning side was developed using the scikit-learn library Random Forest. In particular, its functioning is based on the following steps:

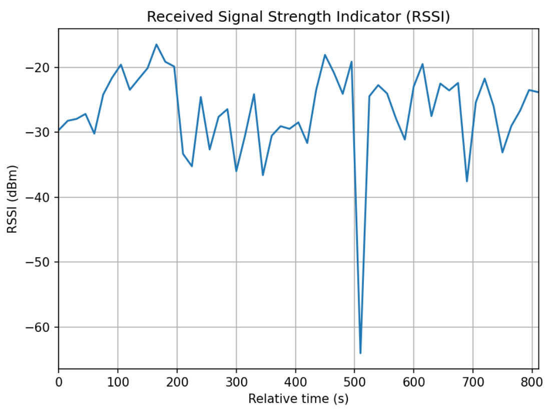
- ***Feature extraction:*** the relevant features are extracted from the raw traffic data, contained in `model/features_W15.csv`.
- ***Data labeling:*** the labels are assigned to the features based on the type of activity. This is done manually via the script `srclabeling.py`, which generates the files `model/labels_by_time_W15.csv` and `model/features_labeled_W15.csv`.
- The dataset is split into training and test sets (based on the parameter TEST SIZE), a Random Forest classifier is trained on the training portion, and its performance is evaluated on the test set through accuracy, classification report, and confusion matrix.

Results

Signal strength

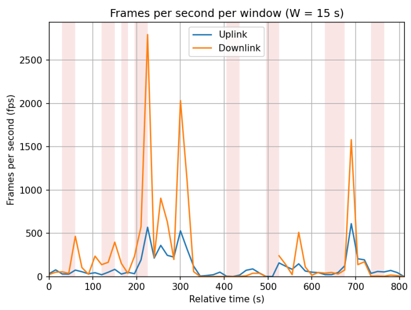
For the first part, the plots generated provide initial insight regarding the RSSI and the SNR, just to be sure that the test is mostly valid. As plotted, the RSSI and the SNR are not only in the expected range, but also strictly correlated. This highlights the absence of noise during this experiment.

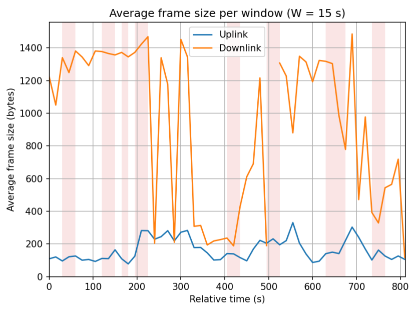


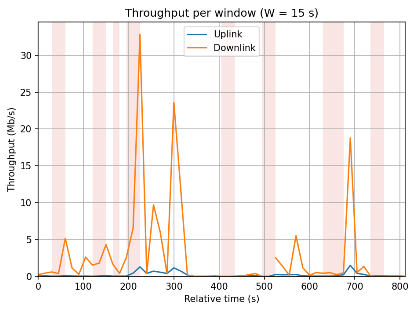


Throughput and Frames per second

The plots regarding the frames captured all highlight in red the areas where the station transceiver was off most of the time.







Starting the analysis from frames per second, frame size, and their obvious consequence, throughput:

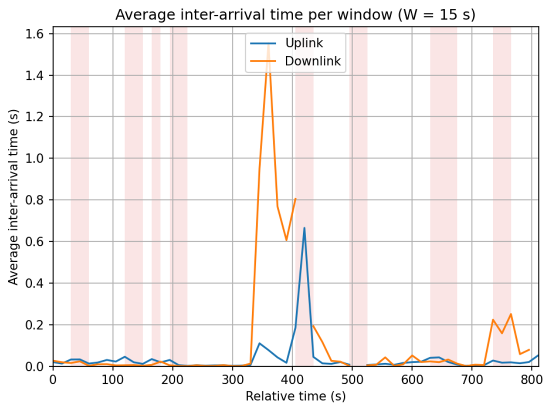
- ***Average frame size:*** Downlink frames are generally much larger than uplink ones, with consistent sizes above 1,000 bytes during high-traffic periods, while uplink traffic remains small and stable.
- ***Frames per second:*** Strong peaks appear during active downlink traffic, especially when the station is awake. During sleep intervals (red shaded areas), the number of frames drops sharply, confirming reduced channel activity.
- ***Throughput:*** Traffic is heavily dominated by downlink, with bursts up to 30 Mb/s in short intervals. During sleep phases, throughput collapses close to zero, showing the strong impact of power-saving states on data transfer.

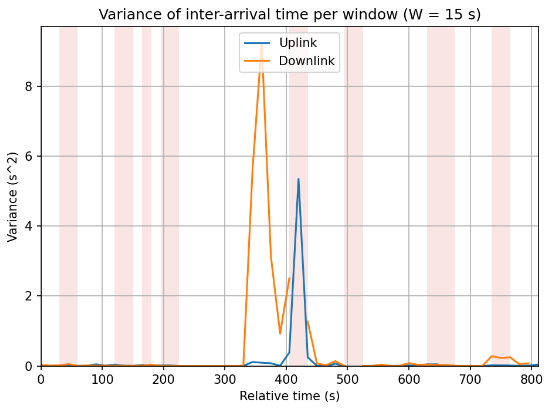
Inter arrival times

The plots above show the average and variance of inter-arrival times for uplink and downlink traffic.

- Both metrics are generally close to zero, indicating regular packet flows.
- Occasional spikes (around 350–400s) correspond to bursts of inactivity followed by traffic resumption, which cause higher inter-arrival times and larger variance.
- During predominantly sleeping periods, inter-arrival times remain low, as the station transmits very few packets.

These metrics highlight the difference between steady high-throughput traffic and sporadic or bursty transmissions.

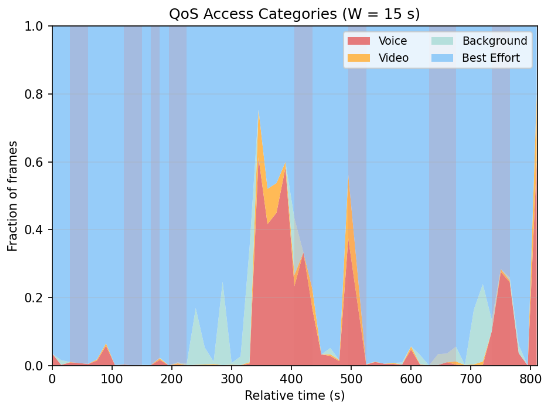




Quality of Service

The plot illustrates the distribution of QoS access categories over time.

- Best Effort (blue) dominates most of the capture, representing general web traffic.
- Short intervals of Voice (red) and Video (orange) appear during higher activity phases (e.g., streaming), highlighting latency-sensitive traffic.
- Background (green) traffic is sporadic and accounts for only small fractions.
- The station tends to enter sleep states during Video traffic, while it almost never sleeps when Best Effort dominates.



Random Forest Output

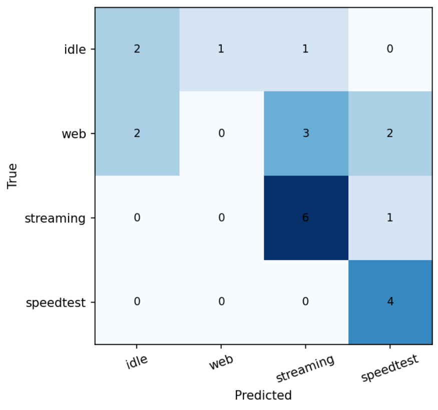
```
Train acc: 1.0
Test  acc: 0.5454545454545454

accuracy                                0.55        22
macro avg           0.42        0.59    0.48        22
weighted avg        0.39        0.55    0.45        22
```

The Random Forest achieved 100% training accuracy but only 55% accuracy on the test set. This indicates clear overfitting: the model memorized the training data but generalized poorly. The classification report also shows that some classes are rarely predicted, confirming class imbalance.
This result could be improved by collecting more balanced data across all activities.

Confusion Matrix

The confusion matrix shows that the model is able to clearly distinguish streaming and speedtest sessions, which are recognized with high accuracy.



True \ Predicted	idle	web	streaming	speedtest
idle	2	1	1	0
web	2	0	3	2
streaming	0	0	6	1
speedtest	0	0	0	4

Web browsing and idle, on the other hand, present more overlap: some idle intervals are misclassified as web or streaming, and part of the web traffic is confused with both streaming and idle. This outcome reflects the similarity of their traffic patterns, while more distinct behaviors like speedtest (high throughput bursts) and streaming (steady video flows) are easier to separate.