A Classifier Based On Linear Discriminant Analysis For Packet Status Prediction In A Wireless Star Network Topology

Mariana Ávalos Arce
Universidad Panamericana
Faculty of Computer Science
Spring 2022

Guadalajara, Mexico

A Classifier Based On Linear Discriminant Analysis For Packet Status Prediction In A Wireless Star Network Topology

Mariana Ávalos Arce
Universidad Panamericana
Faculty of Computer Science
Spring 2022

Abstract

The purpose of this document is to present the complete process of data extraction, collection, visualization, analysis for a packet status predictor in a wireless IEEE 802.15.4 protocol network that implements a star topology. Packet loss is one of the two most researched problems for controlled network systems since it increased delay time in said systems. The model for the prediction in the present study is built using Linear Discriminant Analysis and further performance metrics are discussed.

---

1 Introduction

1.1 The problem

Nowadays, Wireless Networks are used in industrial automation, vehicles, remote surgery, robots, mobile sensor networks vehicles, and Internet-based applications. A **Wireless Network** is then a connection of two or more nodes where information is sent or received (or both) without the use of a **physical cable connector** between the nodes.

A **packet** is a small segment of a larger message. Data sent over **a network** is divided into packets, and the destination node recombines the packets that form a single data piece when the packets are received. Theoretically, it could be possible to send data over a network without chopping them down into small packets of information. However, such an approach becomes impractical when more than two nodes are involved in the network: whenever any long line of bits gets passed over the network involving two nodes, the other nodes would need to **wait** for this communication to finish so that they can make use of that channel [5]. In other words, packet division for data makes a network able to exchange billions of files instead of just a few. It also means that **packets can take different network paths to the same destination**, as long as they all arrive at their destination.

This management of information as independent pieces that travel in *any order* involves multiple problems when in practice: packets may collide and get lost in the process of arriving to its destination, and this is called **packet loss**. Intermittent data **packet losses** and **network-induced time delay** are known to be two of the main causes for performance deterioration or even instability of any controlled networked system.

1.2 Hypothesis

Due to the nature of **shared paths** of communication in wireless networks described above, the study of packet loss has attracted considerable research interests. With the purpose of further investigating this problem and with the aim of providing new insights of what are the conditions in which packet loss is presented, the hypothesis of this study is:

There exists a condition or a set of conditions in a network's environment that, when present, cause packet loss.

In the following pages a detailed description of the installation of a network testing environment, its data storage and data analysis will be presented, alongside a prediction model for the Packet Status (OK or ERROR) that results from the data collected.

2 Data Collection

2.1 Testing Hardware and Architecture

For the study of packet loss, it is intuitive that a **network** needs to be installed and tested, but also **observed**. The network installation was done using Texas Instruments' Evaluation Boards of models CC13x and CC26x as nodes. These pieces of hardware were chosen since they are both boards with wireless protocols support and a built-in temperature sensor. The network had two types of nodes: the **collector** and **sensor** node, so as to implement the simplest form of master-worker architecture that most networks nowadays require, which at the same time implements **a star network topology**, symbolized by the collector and sensors as a star. Specifically, the collector node was a CC26x Evaluation Board (1) and the sensor nodes where a mixture of CC26x and CC13x Evaluation Boards (8). The boards mentioned support different wireless network protocols: Zigbee, Bluetooth and IEEE 802.15.4.

One of the basic examples provided by Texas Instruments guide to IEEE 802.15.4 was a **collector-sensor** model that involves N nodes sending temperature data from the built-in sensor to the collector every certain amount of time. The study involved the loading of such sample code to each of the 9 evaluation boards. For the **observation** required to analyze the network packets, a *sniffer board* was acquired. A **sniffer** refers to a program that performs the monitoring of a network's traffic in real time. More specifically, the study used Texas Instruments' CC2531 USB Dongle, which only supports IEEE 802.15.4 wireless protocol. The sniffer board comes with an open source specialized software, which became for the present study, TI SmartRF Packet Sniffer 2.18.1 Software, which is a UI for visualizing and storing the data the sniffer captures in real time.

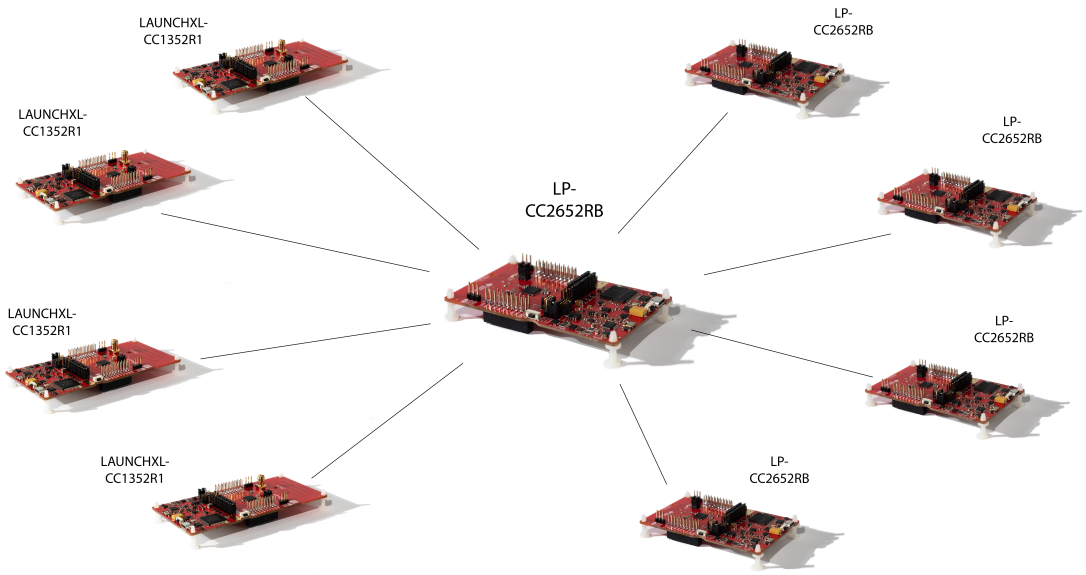

Figure 1: Testing architecture implementing a star topology.

2.2 Experimentation

Once the nodes had their respective code loaded, the nodes were placed in a closed environment consisting of a 10 mts x 30 mts rectangle room, resembling the living room and kitchen of a household, so that the study reflects the usual conditions where networks are used nowadays by billions of users around the world in their homes.

As a total, the study involved **7 Controlled-environment Tests**, where each of them exposed the installed network to different conditions over **1 hour of duration**, that is, the following tests were each performed in a 60 minute time period where the sniffer and network are working simultaneously. The tests are:

- Control Test 1: this is the *control* or baseline network data for the experiments. In this test there were **no interference or barrier conditions** at all to stress the network, so that it represented the **ideal conditions** of the network.
- Control Test 2: this is another instance of the previous test, captured right after to later be averaged with the first control test and form a single Control Test.
- Double Network Test: this involved another star network nearby the analyzed one, becoming a test for IEEE 802.15.4 network interference.
- Radio-frequencies Test: this involved five common-use machines that produce radio-frequencies of different kinds inside a home: a microwave, an air fryer, a blender, a toaster and a sandwich maker where working *at the same time and place* that the network in question was operating in. The said equipment can be seen in Figure 2, except for the microwave.
- No restriction Test: this involved the people living in the said house living normally without restrictions, meaning this test would represent the network and interference traffic in an average household.
- Wind Test: this involved two fans blowing air across the space where the collector node was installed, at a speed of 20 km/h.
- Wireless Test: this involved different machines that used specifically **WiFi network protocol**, which were 1 SmartTV, 3 Smartphones, 1 game console and 2 personal computers, all downloading content from the Internet in the same room and at the same time the network was operating.

Figure 2: Radio-frequency machines in a household.

It is important to note that, while each of the tests was being performed, the sniffer was also capturing in real time the network packets of the network in question, and after each individual test was finished,

the captured packets over that test were stored in a file format named Packet Sniffer Data (PSD) using the Packet Sniffer Software mentioned. Therefore, at the end of the experimentation, the outcome were 7 files with `.psd` extension that contained the packets' information the sniffer captured.

2.3 Building The Dataset

The PSD file format can be read using TI Packet Sniffer Software, but what was needed now was to *translate* that information into a tabular file format, such as a CSV file. To do so, we developed a program that read each of the `.psd` files byte-wise, following the specifications provided by Texas Instruments' manual *SmartRF™ Packet Sniffer User's Manual*[3] and IEEE 802.15.4 signals standard in the manual *Generation of IEEE 802.15.4 Signals*[6]. The first source specified that the PSD file format stores each packet with the structure seen in Figure 3.

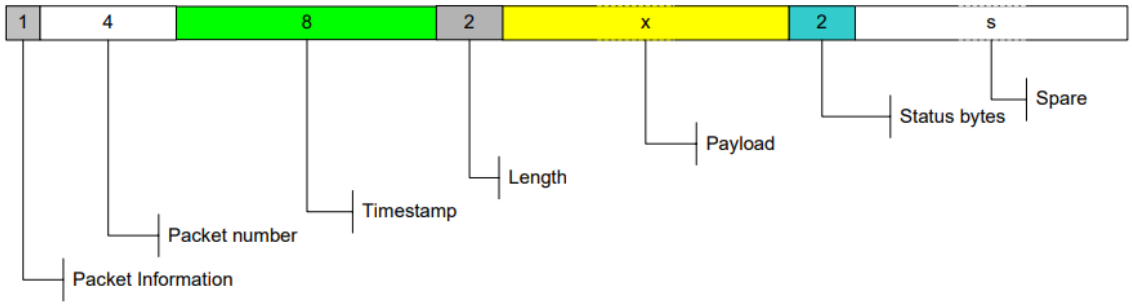

Figure 3: A packet structure (in bytes) inside a PSD file.

In the manual [3] it is specified that the information in a PSD file uses the **little-endian format** since the sniffer chip is an ARM processor [4], meaning, for instance, that the number 0x12345678 is stored as 0x78, 0x56, 0x34, 0x12 in the PSD file. Therefore, the PSD format has the information written by pairs of bytes written in little endian format and then stored as hexadecimal numbers. Thus, the program for decoding a PSD file consisted basically of an algorithm that reads bytes or pairs of bytes of hexadecimal numbers, inverts its paired order (little endian) and then converts the hexadecimal into its decimal equivalent, following the structure in Figure 3 so as to know when a new packet begins and ends, until the end of file.

It is also important to note that every packet contains its **Frame Control Type**, inside the payload block [1], and by following the manual *Generation of IEEE 802.15.4 Signals*[6], there can only be 8 different frame types: BCN, DATA, CMD, ACK, 1_OCT_HEADER, CSL_WAKEUP, CSL_SECURE_ACK and RFID_BLINK, with each having a 3-bit code. The frame type is crucial since it refers to the type of command that sent a package, which in turn specifies if the sender is either the collector/master (Beacon) or one of the sensor nodes. Secondly, in Figure 3 there is a section of the packet consisting of two bytes called **Status Bytes**, which becomes the **class** of the packet in our study: OK (1), when a packet was captured as it is meant to be received; and ERROR (0), when a packet is captured after a collision or fragmentation that causes the loss of a packet in the network. This will become the target column or class column for the study's data analysis.

Finally, after the coded program translates a PSD byte-wise information into decimal form, the said program writes a CSV file per PSD file with all the decoded information. The complete amount of columns in the dataset were 34 columns named: 'FCS', 'CORR', 'BUFF_OVERFLOW' and 'GENERIC' are columns containing the Packet Information byte; 'PCKT_NUM' is basically a counter of the packet number; 'TIME(MS)' is the microseconds that the packet took to be received; 'LENGTH', 'PAYLOAD',

'RSSI'; 'CRC_OK', which is the frame status column (class); 'FRAME_TYPE', which contains any of the eight frame types mentioned; 'SECURITY_ENABLED', 'FRAME_PENDING', 'ACKNOWLEDGE_REQ' and 'PAN_COMPRESSION' were columns coming from the Frame Control Field details; 'DEST_PAN', 'DEST_ADD', 'SRC_PAN' and 'SRC_ADD', which were columns that might be filled with an hexadecimal PAN address depending on the Frame Control Field and the frame type [2]. The remaining 15 columns were the **label binarization** of the eight frame types, the column containing the type of test of the file (6 different possibilities coming from the tests performed), and the **label binarization** of these six test types just mentioned. The binarization of labels was stored for an easier data processing in later phases of the study.

3 Data Analysis

3.1 Dataset Characteristics

Once the complete CSV file was constructed by the program mentioned, we could start understanding the dataset. It contained 33 thousand rows and 34 columns, which meant 33 thousand packets to analyze. Most of the analysis is done **by frame type**, since each frame type packet is sent under common conditions.

The first concept to understand in signal analysis is *how the signals happen throughout time*, and thus a time series plot of the packets **per frame type** lets the reader visualize the experiment's trendings. For that, a time series using the column 'PCKT_NUM' and the packet's frame type was used. The average number of packets through time per FCF type is shown in Figure 4 (top).

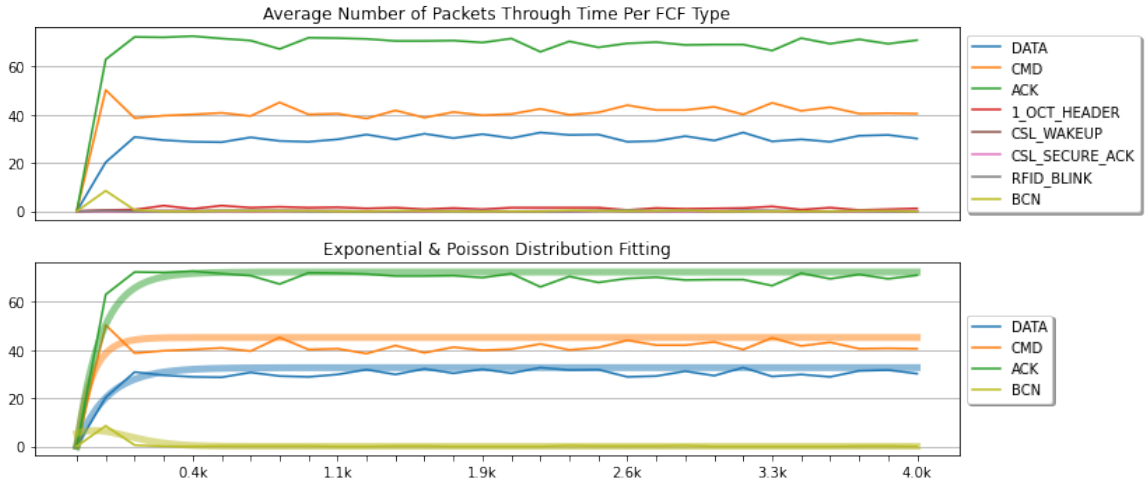

Figure 4: (Top) Average Number of packets through time per frame type. (Bottom) Parametric distribution fittings for top plots.

In the bottom plot in Figure 4 a **parametric distribution** fitting is presented, which confirms the veracity of the tests performed: inter-arrival events are the best example of **exponential distribution**, and such distribution was the one used for the modeling of DATA, CMD and ACK packet arrival behaviour, since those packets are continuously present in the network. BCN frame type packets, as it stands for the Beacon or collector node, just appears in the beginning of the network pipeline and thus follows a **poisson distribution**. The exact anaylitical functions are described below for DATA, CMD, ACK and BCN frames, respectively.

$$f(x) = 1 - e^{-\frac{1}{120}x} \tag{1}$$

$$f(x) = 1 - e^{-\frac{1}{75}x} \tag{2}$$

$$f(x) = 1 - e^{-\frac{1}{150}x} \tag{3}$$

$$f(x) = \frac{1.15^x}{x!}e^{-1.15} \tag{4}$$

These curves do not include the remaining frame types since those were barely present in the experiments to have a mathematically defined behaviour.

In order to decide with which model proceed with the classifier, all columns have to be analyzed individually. Most of the columns are binary data, which is explained by the fact that the PSD format stores data by bytes and therefore most fields in the dataset are binary or boolean representatives. The only continuous fields in the dataset are LENGTH, RSSI and TIME(MS) columns. Thus, a box plot was done per continuous column, but before so the columns were grouped by TEST_TYPE (interference type) in order to output Figure 5.

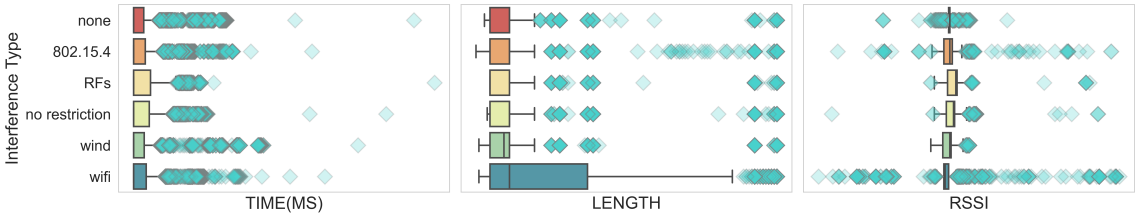

Figure 5: Box plot of the continuous columns per test type.

From Figure 5 we can conclude that WiFi interference generated the biggest fluctuations in all three columns. WiFi interference was also the test type that showed the largest absolute correlation to the target column (-0.13) out of all types of interference in the study.

3.2 LDA: Projecting Data in The Plane

Since the majority of columns in the dataset are binary, and the number of columns makes it difficult to visualize the columns' relationship with the target, a pair plot was computed. Pair plots are the same as random projections of two axes. Since the amount of non-repetitive and related to the study columns is actually 24, the amount of pairs should be 12. In Figure 6 the pair plot or random projections by pair columns is presented, where the color of the scatter dots is defined by the class or status of the packet.

From the pair plots there is really no perceivable separability between classes. Linear Discriminant Analysis was used to find a possible axis from which the model can separate the two classes. Since we have either class 0 or 1 (c = 2), and LDA reduces dimension to c - 1 dimensions, the model will reduce it to 1 eigen value that summarizes the whole relationship of the columns to the target. The LDA mentioned results in an axis shown with a dashed line in Figure 7. LDA reduced the dataset to one dimension, and thus the scatter plot should not have two dimensions, but for visualization purposes, x axis spreads the dots horizontally to showcase the amount of dots per class by each side of the found axis.

3.3 Performance Overview: ROC-AUC

To evaluate the classifier that was created using Linear Discriminant Analysis, we computed the confusion matrix. The number of packets with class 1 (OK) and 0 (Error) was equal, and thus the matrix shows

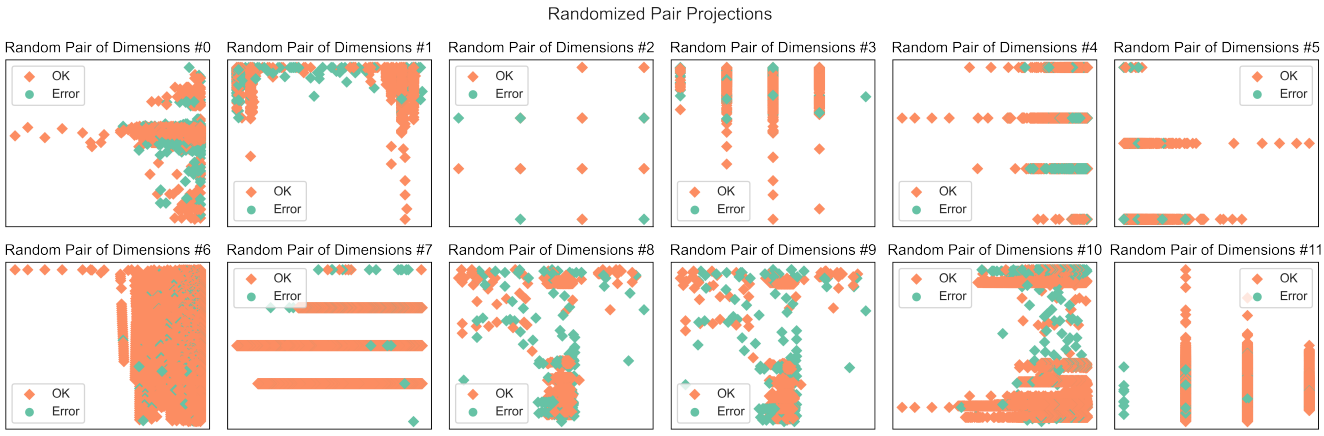

Figure 6: Pair projections of random pair of columns.

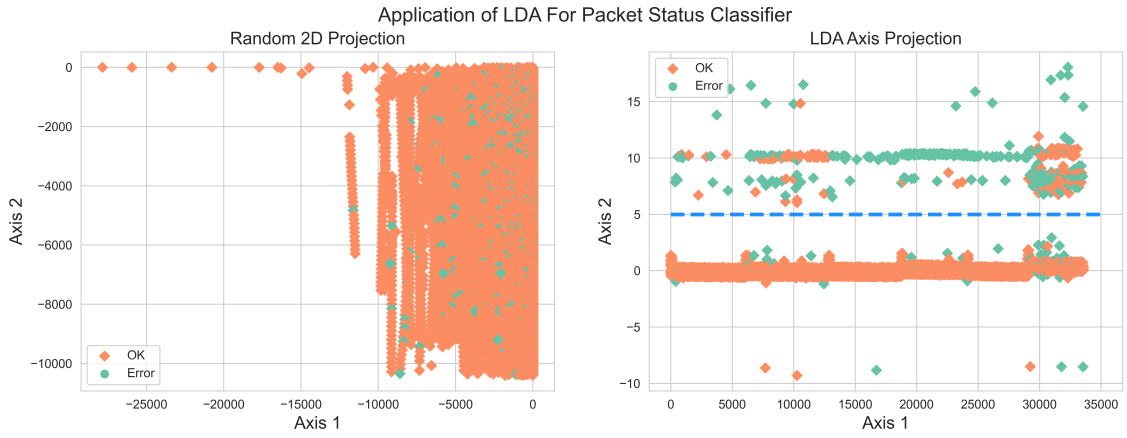

Figure 7: Separation axis found by LDA-reduction model.

a **high TPR** and almost **mid-valued FPR**, which is the desired behaviour for a failure related model: false positives are more dangerous to studies that rely on packet status than False negatives. The ROC Curve gave an AUC = 0.76 , suggesting the classifier is **moderately accurate**. The confusion matrix and ROC Curve for the model's behaviour is shown in Figure 8.

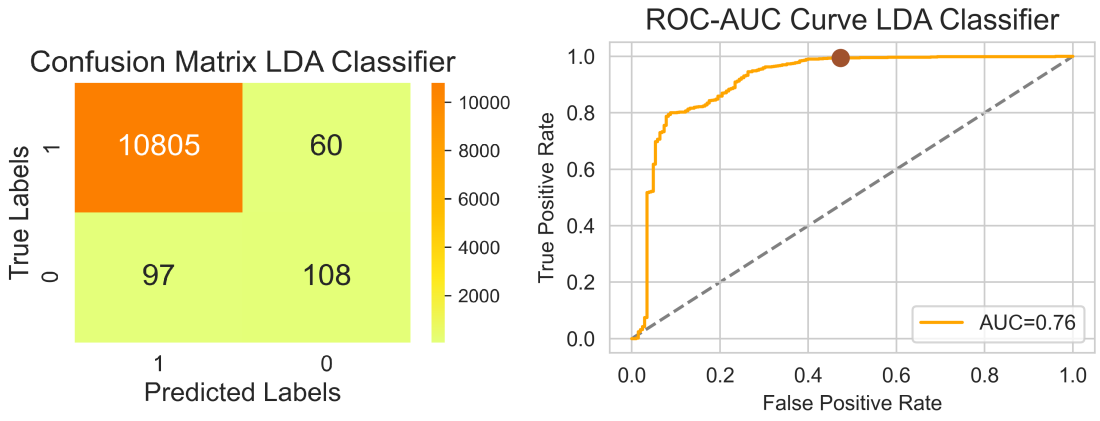

Figure 8: (Left) Confusion Matrix and (right) ROC Curve.

4 Conclusions

4.1 Discussion

The AUC = 0.76 shows a prominent suggestion that there exists in fact a relationship between the *combination of columns* and the prediction of a packet status, like it was stated in the hypothesis. There is not, however, a strong relationship - at least linear - between one column and the target, since the correlations of all the columns with the class never goes beyond 0.13. Therefore, the hypothesis is accepted but it must be noted that the features of a packet *combined* can predict what a handful or one feature cannot.

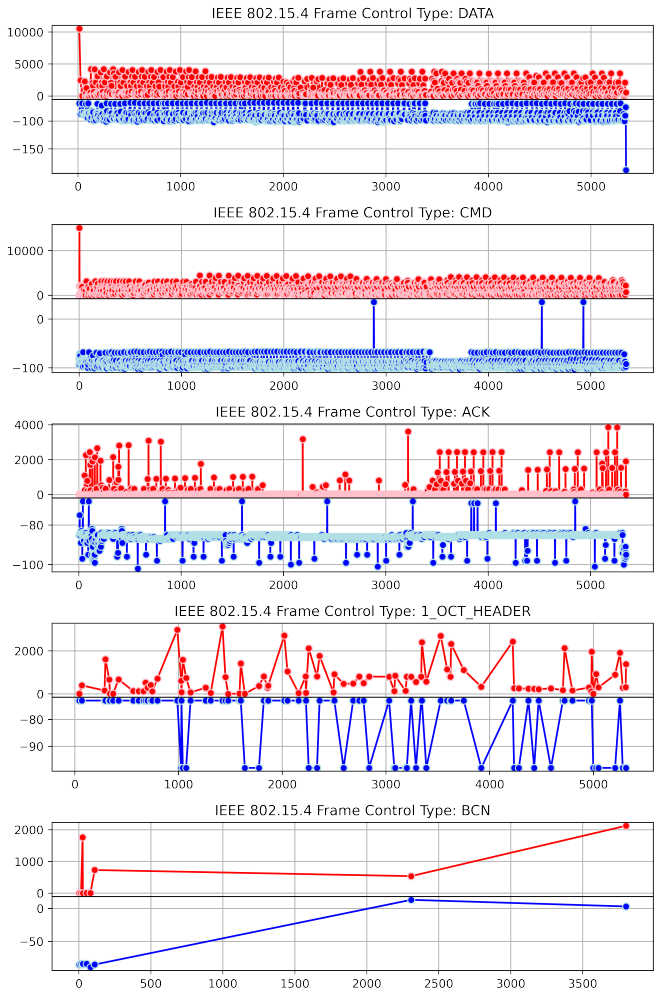

Figure 9: RSSI per frame type throughout time

As a side note, out of all the interference tests performed, WiFi interference seemed to cause the largest fluctuations in data, resulting in larger amounts of outliers as well, shown in Figure 5, which coincides with the highest linear correlation with the target out of all interference exposed to the network. It is important to note that the frame type 1_OCT_HEADER stands for 1 Octal Header Low Latency, and it resulted in 0.44 (highest) linear correlation to the target class, but it is a trivial result since Low Latency packets are sent when the conditions of the network are forcing low response from the devices, and therefore packet loss is expected.

All the resources, including source codes developed for this study are in:

https://github.com/the-other-mariana/star-network

4.2 Future Work

The information regarding the RSSI column in the dataset can suggest further work aside from the presented classifier, due to the fact that if we group again by frame type and plot RSSI across time, we get the plots in Figure 9 which suggest certain RSSI is picked for certain frame types and it prevails for the cases of CMD, DATA and ACK, but fluctuates for 1_OCT_HEADER. Further RSSI conclusions can be done in future work.

References

[1] Vladimir Alemasov. *Whsniff*. URL: https://github.com/homewsn/whsniff/blob/master/src/whsniff.c.

[2] Renwei Huang et al. "Analysis and comparison of the IEEE 802.15. 4 and 802.15. 6 wireless standards based on MAC layer". In: *International Conference on Health Information Science*. Springer. 2015, pp. 7–16.

[3] Texas Instruments. *SmartRF™ Packet Sniffer User's Manual*. URL: https://www.ti.com/lit/ug/swru187g/swru187g.pdf?ts=1649867575476&ref_url=https%5C%253A%5C%252F%5C%252Fwww.google.com%5C%252F.

[4] ARM Limited. *Memory formats*. URL: https://developer.arm.com/documentation/ddi0337/e/Programmer-s-Model/Memory-formats#:~:text=Little%5C%2Dendian%5C%20is%5C%20the%5C%20default,to%5C%20data%5C%20lines%5C%207%5C%2D0..

[5] Henning A Sanneck and Georg Carle. "Framework model for packet loss metrics based on loss runlengths". In: *Multimedia Computing and Networking 2000*. Vol. 3969. International Society for Optics and Photonics. 1999, pp. 177–187.

[6] Rohde & Schwarz. *Generation of IEEE 802.15.4 Signals*. URL: https://scdn.rohde-schwarz.com/ur/pws/dl_downloads/dl_application/application_notes/1gp105/1GP105_1E_Generation_of_IEEE_802154_Signals.pdf.