Specific emitter identification with multiple transmission codes and receivers

by

Lodewicus Johannes Diedericks

Submitted in partial fulfillment of the requirements for the degree
Master of Engineering (Electronic Engineering)

in the

Department of Electrical, Electronic and Computer Engineering Faculty
of Engineering, Built Environment and Information Technology

UNIVERSITY OF PRETORIA

February 2024

SUMMARY

SPECIFIC EMITTER IDENTIFICATION WITH MULTIPLE TRANSMISSION CODES AND RECEIVERS

by

Lodewicus Johannes Diedericks

Supervisor:	du Plessis, Warren P.
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Electronic Engineering)
Keywords:	Remote keyless-entry (RKE), software-defined radio (SDR), specific emitter identification (SEI), radio-frequency fingerprinting (RFF).

The critical challenge of enhancing wireless communication security is addressed in this research by introducing a specific emitter identification (SEI) system. SEI systems use the small analogue differences caused by hardware tolerances to uniquely identify transmitters. The primary focus of this SEI system is identifying transmitters within a remote keyless-entry (RKE) system. RKE systems were identified as a vulnerable wireless communication system and was chosen for their accessibility and relevance.

The SEI system's development primarily focuses on the discrimination of thirteen gate access remotes and three software-defined radios (SDRs), specifically utilized for replay attacks. It's worth noting that previous studies in this field have often employed a limited number of transmitters for their investigations. This raises questions regarding the scalability of SEI systems, particularly concerning the similarity of features when dealing with a more extensive set of transmitters.

Two different digital codes were recorded for each transmitter. The different transmission codes were introduced into the dataset as it is a limitation of prior designed SEI systems. The SEI system in this research was designed to overcome this limitation and improve the robustness of the system.

Multiple receivers were used when recording the transmitter's analogue signal. Four receivers of the same make and model were used to record the signal. Multiple receivers were used to test the effect of using different receivers in training and testing combinations to test the robustness of the system, which was not explored in prior research.

Statistical analysis was utilized for feature extraction to describe the analogue signal. Through statistical analysis, the receiver sampling rate required to capture the digital code's characteristics are reduced compared to other methods.

Feature reduction was implemented by testing various feature management methods. It was found that the feature management method employed after feature extraction has a significant impact on the accuracy of the SEI system. The classifier's performance is directly impacted by the number of features, and features with lower importance can reduce the accuracy of the system. Principal component analysis (PCA) was chosen as the feature management technique because it demonstrated the highest accuracy for this dataset. While offering advantages such as dimensionality reduction, noise reduction, and feature decorrelation.

Similarly, different classifier methods underwent thorough evaluation, to ensure the reliability of the accuracy scores, a rigorous approach was taken. K-fold cross-validation was used to train and validate the models, enhancing their generalizability. The random forest classifier was selected for its superior performance and benefits, including robustness against overfitting, feature importance ranking, and suitability for high-dimensional data. Furthermore, the data was divided into training and testing data, where the training dataset was used to train and validate the model and the testing dataset was employed for testing, providing a robust assessment of the SEI system's performance.

The results showcase the SEI system's effectiveness. When dealing with a single transmission code and receiver, it achieves an accuracy rate of 98.6%, establishing a crucial performance baseline. This result is comparable with several other SEI systems, demonstrating the system's effectiveness. When multiple digital codes are considered with a single receiver, the system maintains an accuracy rate of 95.4%, demonstrating robustness.

When multiple receivers are used, the system's performance is greatly affected by the receiver combination used. If a receiver is used for testing, and no data from that receiver was used during training, the accuracy rate drops to less than 33%. When integrating a receiver's characteristics into the training data, the system attains an accuracy rate of 92%. This was achieved through using a 90-10 train test split, where 90% of the receiver pair's data was used for training and validation and 10% of the receiver pair's data was used for testing. This result demonstrates the importance of incorporating receiver characteristics into the classifier when using multiple receivers.

LIST OF ABBREVIATIONS

ADALM	Analog Devices active learning module
ADC	analogue-to-digital converter
AUC	area under curve
CNN	convolutional neural network
DAC	digital-to-analogue converter
DFT	discrete Fourier transform
ELINT	electronic intelligence
ES	electronic support
FFT	fast Fourier transform
FIR	finite impulse response
FNR	false negative rate
FPR	false positive rate
GSM	Global System for Mobile Communications
GUI	graphical user interface
ICA	independent component analysis
IF	intermediate frequency
IIR	infinite impulse response
ISM	industrial, scientific and medical
KNN	k^{th} nearest neighbour
LDA	linear discriminant analysis
MLP	multi-layer perceptron
NN	neural network
OOK	on-off keying
PCA	principal component analysis
PWM	pulse width modulation
RBF	radial basis function
RF	radio-frequency
RFF	radio-frequency fingerprinting
RKE	remote keyless-entry

RMSP	root-mean-square propagation
ROC	receiver operating characteristic
RWR	radar warning receiver
SDR	software-defined radio
SEI	specific emitter identification
SGD	stochastic gradient descent
SNR	signal-to-noise ratio
STD	standard deviation
SUC	secret unknown cipher
SVM	support vector machine
USRP	Universal Software Radio Peripheral

TABLE OF CONTENTS

CHAPTER 1 INTRODUCTION

1.1 PROBLEM STATEMENT

1.1.1 Context of the Problem

In electronic support (ES) systems, specific emitter identification (SEI), also known as radio-frequency fingerprinting (RFF), is used to identify and distinguish different radio-frequency (RF) transmitters. SEI systems use the analogue RF characteristics of the signal to identify the transmitter source [1]. SEI is possible because the tolerances of the RF hardware introduce small, measurable variances in the signal [2, 3]. The effects that the hardware has on the signal do not alter the digital code that is transmitted but introduce a small variance in the analogue signal.[1] These hardware-induced variations do not affect the normal operation of the transmitter system itself, ensuring that the system's functionality remains intact. This small variance is difficult to reproduce, and as a result, replay attacks on SEI are far more challenging, if not impossible [4].

Potential applications of SEI include cases where there is a need to uniquely identify a specific transmitter. For example, the ability to identify a specific radio would allow one to track the movement of that radio, even if its signal is lost for some time (e.g. when it is not used). Such applications include law enforcement in cases where there is a need to prove that a given transmitter (e.g. a cellular telephone) was used during a crime such as smuggling. Another application that was considered in this research is the ability to identify different access remotes. This would allow one to distinguish between a trusted transmitter and allows one to detect a replay attack.

Access remotes transmit an RF signal with a specific code to gain access to cars, gates, doors, and

[1]In this document, code refers to the digital code of the transmitter and signal refers to the analogue signal.

security alarms. This process is known as remote keyless-entry (RKE) [5]. In an RKE system, the access remote transmits an RF signal which contains a digital code. The RF signal is then received by an RF receiver, and if the code is correct, it will allow access [6]. This provides a level of security as only the remotes with the correct code can gain access. This also creates vulnerability in the security, as the code that is transmitted by the access remote can be recorded by an RF receiver and replayed by an RF transmitter [1, 5, 6]. This weakens the security and grants unauthorized access. Some access remotes are programmed by cloning this digital code from another access remote, which indicates that this digital code is easily cloned and reduces security [1].

Newer RKE systems vary the digital code of the RF transmission each time the remote is pressed to improve security compared to static code systems. These newer remotes are synchronized with the receiver to ensure that the transmitter and receiver both use the same code [5]. Although this increases security, another vulnerability is found in this case, where a jammer is used to desynchronize the receiver and transmitter. Previous codes of the receiver are used to re-synchronize with the transmitter, and access is gained through a replay attack of a previous code [5]. This demonstrates that these newer RKE systems are still vulnerable to replay attacks because the RF signal is easy to intercept and the digital code of the remote is easily cloned.

SEI systems demonstrate the ability to uniquely identify transmitters using characteristics that are challenging to reproduce, such as the analogue properties of the transmitted signal. This capability would be extremely useful in RKE systems and would improve the security of these wireless communication systems. SEI systems can be deployed to detect different transmitters, thereby enhancing security by effectively eliminating the risk of cloning access remotes and replay attacks.

1.1.2 Research Gap

Although published results indicate that system security can be greatly improved using an SEI system [1], there were some limitations in the results. The classifier used classified the same transmitter as a different transmitter when transmitting a different digital code due to the feature extraction method used [1]. The number of transmitters, training data, and testing data were very limited [1].

In this research, more transmitters will be used with multiple digital codes to determine if using an alternative feature extraction method can detect the same transmitter transmitting different codes. This will indicate if an SEI system is robust and can detect the same transmitter transmitting a different code.

The influence of employing multiple receivers on the SEI system has not been extensively examined in previous research and will be a focal point of investigation in this study. This aspect is particularly relevant, as SEI systems are often employed in scenarios involving multiple receivers, such as electronic intelligence (ELINT) systems, which record SEI data for use in radar warning receivers (RWRs), among others.

Furthermore, a noteworthy finding of this research is the high dependence of the SEI system on the training data utilized, making it challenging to replicate results without access to the specific dataset. To address this limitation and facilitate further research in the field, this study offers the opportunity to establish a comprehensive database, enabling the recreation of results and the exploration of diverse methods and scenarios using the same dataset. This resource can serve as a valuable asset for future investigations in the realm of SEI systems.

1.2 RESEARCH OBJECTIVE AND QUESTIONS

The objective of this research is to design and implement an SEI system to identify different access remotes and distinguish them from replay attacks. The data should include different transmitter codes and the system should be able to identify individual transmitters, even when a transmitter transmits different codes. Multiple receivers should also be used in the data capture to determine the effect of different receivers on the SEI system. A database of the received signals should be captured and made available.

The primary research questions arising from these objectives are summarised below.

- Can an SEI system identify a transmitter even when the transmitter is transmitting a different digital code?
  The goal of SEI is to identify the transmitters solely based on the analogue characteristics of their transmitted signals, but the results produced by some SEI systems differ depending on the code transmitted [1]. This question aims to address this limitation.

- What is the effect of using multiple receivers on the accuracy of an SEI system?
  There are many scenarios where a signal from a transmitter will be recorded using one receiver, while SEI will be performed using a different receiver. This question aims to determine whether the differences between receivers will affect SEI.

1.3 APPROACH

Due to the lack of data available to reproduce results from previous implementations, the data from multiple transmitters with the same make and model will be captured.

Multiple software-defined radio (SDR) receivers of the same make and model will be used to record the transmitter and replay-attack signals. The receivers will be connected to an RF splitter with all the receivers connected to the same antenna. This will ensure that the effect of different antennas is not included in the measurements. Operational scenarios may involve diverse antenna setups which would affect the received signal with both the differences in antenna properties as well as receiver properties. This research focuses on the difference in receivers and therefore the antenna setup will be kept constant to ensure only the receiver properties will be measured.

To determine if the same transmitter transmitting different digital codes can be identified, the feature extraction will have to be modified to be insensitive to the digital code. Feature analysis will be performed to reduce the number of features and find the effectiveness of the features, with the most effective features being used in the classifier. The feature data will be split into testing and training data ensuring that testing data will only be used for testing to ensure that the results are reliable.

Multiple classifiers will be trained and evaluated in a systematic manner using the training data. To maintain the integrity of the experimentation, great care will be taken to ensure a clear separation between the training and testing datasets. K-fold cross-validation will be employed with the training data for the initial training and validation stages. While the testing data will exclusively serve for the final evaluation of the trained classifiers. This approach adheres to best practices in machine learning, where the testing data is used solely for assessing the classifiers' performance after all other aspects of system development, such as classifier selection and hyperparameter tuning, have been completed.

A crucial aspect of this research involves examining the impact of utilizing different receivers within the SEI system. To comprehensively assess this, various combinations of training and testing will be rigorously explored, involving different receivers of the same type. This examination will encompass a range of receiver pairings to thoroughly evaluate the effect of varying receiver combinations on the system's performance. By systematically analyzing these combinations, the system's robustness and adaptability across different receivers are evaluated.

In the evaluation of the effect of different receivers on the SEI system, a comprehensive approach will be adopted. Multiple receiver pairings will be considered for testing, including scenarios where all the data from the test receiver is deliberately excluded from the training and validation phases. Additionally, an alternative approach will involve training the classifier with the characteristics of the receiver pair incorporated into the model. This will be accomplished through the implementation of a 90-10 train-test split, with 90% of the data from the receiver pairs dedicated to training and validation, and the remaining 10% reserved for testing. By contrasting the results of these two approaches, this research aims to shed light on whether the SEI system exhibits a dependency on the receiver and whether the inclusion of receiver characteristics significantly impacts the system's accuracy. This investigation is anticipated to contribute valuable insights into the role of receivers and their characteristics in the SEI system's performance.

1.4 RESEARCH GOALS AND CONTRIBUTIONS

Previously published results have demonstrated the potential for significant improvements in system security through the use of SEI systems [1]. However, these prior findings had certain limitations. This research aims to enhance an existing SEI system [1] by focusing on the identification of transmitters of the same make and model that transmit different digital codes. By incorporating a larger set of transmitters with multiple digital codes, alternative feature extraction methods can be implemented to effectively identify the same transmitter when transmitting different digital codes. This investigation serves to improve the robustness of the SEI system in detecting transmitters with varying digital code transmissions.

One aspect that has not been adequately explored in prior research is the influence of different receivers on the accuracy of the SEI system. This study addresses this research gap by examining the impact of using different receiver training and testing pairs. Multiple receiver pairings will be considered, including scenarios where all the data from one receiver is used to train and validate the system, and another receiver of the same type is used for testing. Alternatively, the characteristics of the receiver can be trained into the classifier. This exploration is significant as it sheds light on the role of receivers in the SEI system's performance.

Additionally, an important observation is that the performance of the SEI system appears to be highly dependent on the training data used, making it challenging to reproduce results effectively. This research provides an opportunity to create a comprehensive dataset, enabling replication of the results, and exploration of various methods and scenarios using the same dataset. This dataset availability is a valuable contribution of this work, fostering reproducibility and further research endeavors.

1.5 RESEARCH OUTPUTS

By the time of writing this, a journal article is in preparation to be submitted. The journal paper will be the first known publication that presents results on multiple receivers in a SEI system, and successfully identify transmitters with multiple digital codes.

> L. J. Diedericks and W. P. du Plessis, ‘Specific emitter identification with multiple transmission codes and receivers,’ IEEE Transactions on Aerospace and Electronic Systems, in preparation.

A crucial outcome of this research is the creation of a comprehensive dataset comprising analogue signal recordings used throughout the study. This dataset contains ten recordings of thirteen gate access remotes, and three replay attack SDRs retransmitting a recorded signal from two remotes, and each transmitter transmits two different digital codes. This dataset was challenging and time-consuming to create, and it is a valuable resource for replicating the results obtained, conducting further investigations, and exploring various methods and scenarios using the same data. Researchers interested in signal processing, machine learning, or security-related fields can benefit from access to this dataset. To promote collaboration and knowledge-sharing, this dataset will be made accessible to the research community. It encourages interdisciplinary collaboration and the exchange of ideas, ultimately contributing to the advancement of knowledge in the field. Detailed documentation of the dataset, including data collection methods and variable descriptions, will be made available to ensure its usability.

1.6 OVERVIEW OF STUDY

Chapter 1 introduces the research problem by examining the challenges faced by secure transmission systems in contemporary digital environments. It defines the scope of the study, sets research objectives, and outlines the rationale for the development and evaluation of the SEI system.

In Chapter 2, the theoretical underpinnings of SEI systems are explored in-depth. It investigates several topics, including signal processing, feature extraction, feature management, and classifiers. The chapter delves into various techniques and assesses their relevance to the SEI system. Furthermore, it critically analyzes existing literature, identifying both strengths and weaknesses in current approaches. Key theories and concepts that form the basis of the SEI system's design and implementation are highlighted.

Chapter 3 marks a transition from theory to practical implementation. It offers a comprehensive view of the development of the SEI system, encompassing critical design choices like algorithm selection, hardware components, and software tools. Within this chapter, a step-by-step account of system creation is provided, covering aspects such as data collection, preprocessing, feature estimation and management, and classifier development. A noteworthy highlight in this chapter is the incorporation of multiple receivers, a distinctive feature of this research that significantly bolsters the system's robustness and adaptability, granting it a competitive edge in the SEI systems domain. While this chapter provides an extensive level of detail, especially when compared to similar studies [1], it is crucial to emphasize that this wealth of information is presented with the primary objective of bolstering the credibility of the implementation and measurements. This transparency assumes paramount significance, particularly in light of the research's primary goal, which is to publish the research findings and make a research dataset available to the wider community.

Chapter 4 is dedicated to unveiling the outcomes of the SEI system's rigorous evaluation. To facilitate a comprehensive understanding of the system's performance, this chapter is organized into three distinct sections, each addressing various facets of the system's capabilities. The chapter commences by presenting results obtained when employing a single transmission code, essentially establishing a crucial accuracy baseline. This initial phase serves not only as an assessment but also as a validation, ensuring the consistency and reliability of the system's outcomes, which can be compared with other published results. Subsequently, the narrative of this chapter delves into the examination of system performance when confronted with multiple transmission codes. This exploration underscores the system's proficiency in distinguishing between transmitters and its efficacy in detecting replay attacks. This section essentially validates the system's robustness in real-world scenarios. Furthermore, the chapter ventures into an exploration of the system's adaptability concerning diverse receiver setups and configurations. It systematically assesses how the SEI system performs under various receiver scenarios, shedding light on the impact of different receiver combinations on the system's accuracy. To facilitate clear comprehension of the findings, this chapter utilizes a combination of statistical analyses, graphical representations, and visual aids, ensuring that the results are effectively communicated and interpreted.

Chapter 5 serves as the thesis's culmination. It revisits research objectives outlined in Chapter 1 and summarizes key findings from Chapter 4. The conclusion underscores how the SEI system addresses the research problem and advances secure transmission systems. Practical implications of the findings are discussed, suggesting real-world applications. Limitations are acknowledged, and areas for future research are identified.

CHAPTER 2 LITERATURE STUDY

2.1 CHAPTER OBJECTIVES

This chapter constitutes the literature review section of the research, focusing on the application of specific emitter identification (SEI) systems in the context of transmitter identification. Specifically, the research employs SEI technology to identify transmitters within remote keyless-entry (RKE) systems. The choice of RKE systems as the subject of investigation stems from their widespread use and accessibility, as the remotes used for gate access are relatively inexpensive and readily available. This accessibility enables the acquisition of multiple transmitters of the same make and model for research purposes. The literature review encompasses an examination of RKE systems to identify their inherent security vulnerabilities and explores how SEI can be leveraged to enhance their security. This comprehensive review is further subdivided into various subsections, each addressing specific aspects of the SEI system and its application within RKE systems.

Section 2.2 provides an overview of SEI systems in the context of transmitter identification, with a specific focus on their application within RKE systems. The accessibility and affordability of RKE transmitters make them suitable for research and development. The section highlights the security concerns associated with RKE systems, including the distinction between fixed code and rolling code access remotes. Furthermore, it introduces how SEI methods can enhance the security of RKE systems by thoroughly analyzing both the transmitter and the digital code before granting access.

A flow diagram is illustrated in Section 2.3 where the key components of a typical SEI system, which includes the radio-frequency (RF) receiver, feature estimation, and classification are shown. Each component's role in the SEI process is discussed in detail, setting the stage for a comprehensive understanding of SEI technology.

The significance of the RF receiver within SEI systems is emphasized in Subsection 2.3.1, as the quality of data for feature extraction is highly dependent on the performance of the receiver. Various types of receivers are introduced, along with their potential impact on the accuracy of the SEI system. The subsection also explores how the performance of SEI systems can vary depending on the quality of the RF receiver.

Subsection 2.3.2 dives into the critical role of feature estimation within SEI systems, emphasizing the need for features that are consistent between multiple recordings of the same transmitter and distinct between different transmitters, effectively describing the transmitter signals. Various feature extraction methods are discussed, along with the importance of mitigating noise and interference that may affect feature estimation. In Subsection 2.3.2 feature analysis algorithms are presented as methods to reduce the dimensionality of feature data while maintaining accuracy, addressing the challenges of high-dimensional data in SEI systems.

Finally in Subsection 2.3.3 various classifiers are introduced as options for SEI systems. The choice of a classifier is discussed in the context of the nature of the input data, emphasizing that different types of data may require different classification methods.

2.2 REMOTE KEYLESS-ENTRY SYSTEMS

RF access remotes are divided into two categories: fixed code and rolling code access remotes [6–8]. Fixed code access remotes transmit a static code. This code is programmed into the transmitter and the receiver. When the transmitter is activated, the receiver determines if the code is correct, and if the received code is correct access is granted [6–8]. Access remotes are vulnerable to replay attacks as the modulation of the code in access remotes are generally pulse width modulation (PWM), on-off keying (OOK), or a combination of the two [6, 8]. As a result of the modulation, the code that is transmitted can be easily replicated by receiving the signal and playing it back with an RF transmitter. The static code access remotes are extremely vulnerable to this attack as they transmit the same code each time the transmitter is activated. Once the transmitted signal is intercepted, the attacker can gain access by playing back the received signal.

Rolling code access remotes are more robust against replay attacks as the transmission code changes for each transmitter activation. Unfortunately, the rolling code security method relies on synchronization

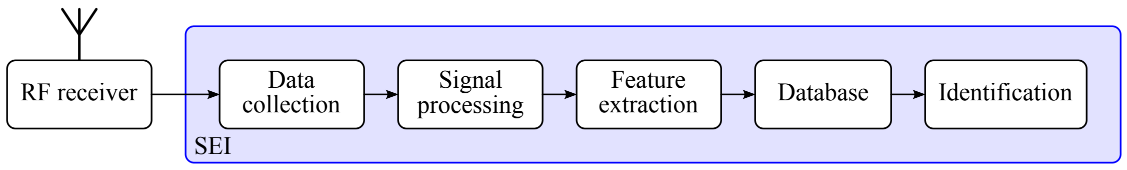


Figure 2.1. Flow diagram of a typical SEI system [3].

which can be exploited by jamming the receiver. Once the synchronization is broken, a previous code can be transmitted to re-synchronize the system and access is granted through a previous code [5–9].

Several different methods are proposed to increase the security of RKE systems. These systems involve adding components such as secret unknown cipher (SUC) to the transmitter as well as receiver to alter the transmission code or to increase the complexity of the system and therefore reduce the effect of replay attacks on the system [5, 8, 10, 11]. These methods result in an expensive solution, especially if a large number of access remotes are considered.

The SEI method can increase the level of security as the transmitter and the digital code are considered before access is granted. This method results in a more secure, and less expensive solution when compared to some of the other solutions, as only the receiver has to be changed.

2.3 SPECIFIC EMITTER IDENTIFICATION SYSTEMS

A flow diagram of a typical SEI system is shown in Figure 2.1. This flow diagram illustrates the subsections of the SEI system. The precursor to the SEI system is an RF receiver which receives the incoming RF signal and converts the data from the analogue domain to the digital domain for processing. The received data is then stored and processed. This processing will vary based on the application of the SEI system, but would typically include the removal of frequency offsets from the received data, and filtering to remove noise and unwanted frequencies [3, 12–14]. Feature estimation is then performed to determine distinct features of the received RF signal [3, 15]. These distinct features are stored in a database which is used to identify the RF signal. The database is then used in a classifier to associate the estimated features with a transmitter [3, 13, 15]. This classifier is then used to identify the transmitter of a received RF signal.

2.3.1 Radio-frequency Receiver

The RF receiver is an important precursor to the SEI system, as the quality of data that is used for feature extraction is dependent on the RF receiver. The performance of the RF receiver is emphasized in several different SEI systems, as the quality of data will directly impact the performance of the SEI system [3, 4, 13, 16–18].

A study was performed to test the performance of an SEI system with different receivers [18]. The different receivers were all tested independently. The effect of training a system with one receiver and testing it with another was not investigated. Three Universal Software Radio Peripheral (USRP) N210 receivers were used to test low-end receivers and an Agilent spectrum analyzer and oscilloscope were used for a high-end receiver. The low-end receivers had a 14-bit analogue-to-digital converter (ADC) with a sampling rate of 100 MHz and the high-end device used the spectrum analyzer as an intermediate frequency (IF) converter which is connected to a 4 GSamples/s oscilloscope. The results of the tests implemented showed that the accuracy of the SEI system using the lower-end receivers depended on the signal-to-noise ratio (SNR) of the signal. It was found that the accuracy of the system increased as the SNR of the signal increased. The accuracy of the system using the high-end receiver did not change significantly with a change in SNR. The low-end receivers could successfully classify the transmitter with an accuracy of 70% at a SNR of 15 dB. The same feature extraction method was used across all of the tests and it was noted that the feature extraction was less effective for lower SNR.

These results indicate that lower-end RF receivers can be used in SEI systems if the SNR of the received signal is high enough for accurate classification. A RTL2832U software-defined radio (SDR) with an 8-bit ADC with a maximum sampling rate of 2.56 MSamples/s was used for an RF receiver [1]. The classification accuracy in [1] was 97%. This indicates that a lower-end RF receiver can be used in SEI systems if the SNR is high enough.

2.3.2 Feature Estimation

The success of SEI classification depends on the feature estimation [2, 3, 13, 14, 19, 20]. A feature is a numerical value given for specific characteristics of a signal [14]. Typically many different measurements are used for feature estimation and a validation process is then used to determine the preferred features [3, 14, 20]. This process is executed as the signal of each SEI system is different and some feature extraction methods work for one system but are ineffective for another [3, 13, 14]. A good feature needs to be consistent for all the transmissions of the used transmitter while being distinct from the features of other transmitters [3, 13, 14]. Feature estimation will be affected by noise, multipath, interference, false data, and lost data. Therefore it is important to remove or reduce these effects on the data before feature estimation is implemented [3, 13, 14, 18].

Typically the feature measurements used for SEI in radar systems are implemented on a single pulse and include, but are not limited to rise time, the slope of the rise time, overshoot, fall time, and the slope of the fall time [2, 13, 19]. The disadvantage of these measurements is that a high sampling rate and high SNR are required to ensure that these measurements are accurately taken. On the other hand, typical feature measurements used for SEI in wireless systems such as Global System for Mobile Communications (GSM) devices use statistical measurements [1, 21]. The statistical measurements were used in [1] for feature extraction as the sampling rate of the SDR was a limiting factor, and feature extraction measurements such as the rise and fall time can not be accurately measured with the SDR.

It was found that feature extraction on a per-pulse basis results in more accurate feature estimation [2, 3, 14]. Extracting features per pulse significantly increases the number of features considered for classification. If the features were calculated per pulse in [1] the number of features is, 13 pulses with 4 features per pulse for both amplitude and frequency. This results in 104 features per data burst. To reduce the number of features, the method used in [1] was to divide the data burst into five regions and the features were determined for each region. Although dividing the data burst into five regions might result in feature estimation errors as different transmission codes will migrate regions, the number of features is significantly reduced when compared to the per pulse measurement. The number of features can be reduced even further through feature management [2, 3, 14, 20].

Feature management is used to reduce the number of features as higher dimensionality in classifiers results in lower accuracy, this is known as the curse of dimensionality [3, 14]. Feature analysis algorithms can be used to determine the relevance of each feature and return a score based on how effectively the feature describes the data. This score is then used to determine which features are the best to use for the given data and classifier [2, 3, 14, 20].

Some feature analysis algorithms include but are not limited to independent component analysis (ICA), linear discriminant analysis (LDA), gamma test, and the area under curve (AUC) of the receiver operating characteristic (ROC) [3, 14, 20]. Through using feature analysis, the number of features can be reduced and will ensure that the best features are used in the SEI system. All of the feature analysis algorithms mentioned above are feasible for SEI systems [14, 20]. Three different feature analysis methods were tested in [14] to reduce the number of features and the accuracy of the SEI system was tested for each method. The methods tested were: the correlation method, gamma test, and lastly LDA. The SEI system had 18 features with an SEI accuracy of 82.32%, whereas the correlation method resulted in 13 features with a system accuracy of 85.42%. The gamma test resulted in 7 features with an accuracy of 84.97%, and the LDA resulted in 4, 6, and 8 features with an accuracy of 77.4%, 74.93%, and 73.95% respectively. From the results in [14], it is seen that the correlation or the gamma test resulted in the best options for the dataset and feature measurement method used in [14].

It is important to note that for different datasets and different feature measurement methods, the results will differ for each test [14]. As a result, different feature analysis methods are tested to determine which features will result in the most accurate system for the dataset and feature measurements used in the system.

2.3.3 Classification

The classifier can be implemented once the feature estimation is complete and the best features have been selected. A classifier requires a training dataset that will be used to train the classifier and a testing dataset that will evaluate the performance of the classifier. The training and testing datasets must remain separate during the training and testing phases as it would create bias and inaccurate results if the testing data is included in the training data or if the training data is used to test the system.

The training dataset is used to associate the measured features with the transmitter [3, 12, 15]. This association is done by labeling the corresponding features with the transmitter. This is called the training class. After the training data and training class is used to train the classifier, an association is formed between the feature data and the transmitter. The classifier can be tested by predicting which transmitter results in the feature output [3, 14, 15]. The selection of the classifier will depend on data that is used for training and testing, as each classifier uses different methods to classify the data.

Several different classifiers can be used for SEI systems. Some of these classifiers include but are not limited to: support vector machine (SVM), k$^{\text{th}}$ nearest neighbour (KNN), neural network (NN), naive Bayes Classifiers, and K-means clusters [3, 15, 22]. SVM is a supervised classifier algorithm that divides the training class into smaller groups through different types of calculations such as linear, polynomial, radial basis function (RBF), and sigmoidal calculations. The SVM classifier is an accurate and robust classifier [3, 15, 22].

KNN is a supervised classifier algorithm, which determines the distance between the testing value and the training values and selects the label that corresponds to the training data with the smallest distance [15, 22]. The training phase of the KNN method is efficient and computationally simple, but the classification phase of the KNN method is less efficient as the computations are more complex in the classification phase [15].

NN is a supervised classifier method that uses weights to form a connection between the input and output of the classifier. The predicted output is calculated by changing the weights of the connection. The NN method is robust against noise and delivers excellent results in data with low SNR.

A naive Bayes classifier is a supervised classifier that calculates a distance measure for the training data and determines a confidence score given that the testing data is from a specific class [22]. The advantage of using the naive Bayes method is that the confidence score is very large if there is high confidence that the data belongs to a given class, and the confidence score is very low if there is low confidence that the data belongs to the class. A new class can then be introduced as an unknown class and if a set threshold of confidence is not met, the classifier can classify the data as unknown [1].

K-means clustering is an unsupervised classifier that divides the training data into smaller clusters. The prediction is then made with the test data. The test data is assigned to the label of the corresponding cluster with the closest mean value [15].

The choice of the classifier will depend on the input data. It was found that the classifier can have an impact on the accuracy of the system, but the RF receiver and the feature extraction methods have a bigger impact on the accuracy of the system [3, 15].

A transfer learning-based model to detect replay attacks on RKE controlled vehicle systems was proposed in [9]. The replay attacks involve intercepting and replaying the signals to unlock vehicle doors. The model uses a pre-trained ResNet50 deep neural network and classifies signals into three categories: real signals, replay attack signals with high gain, and replay attack signals with low gain. The dataset used in this study was a KeFRA-2022 dataset, and a pre-trained ResNet50 deep neural network. A HackRF One SDR was used for generating the replay attack signals. Image preprocessing was used to classify the signals with data augmentation, and dataset splitting. Three different optimizers stochastic gradient descent (SGD), Adam, and root-mean-square propagation (RMSP) were compared for training. SGD solver was the selected optimizer for the final model due to its superior performance. The author also conducted a five-fold cross-validation method to evaluate the model's performance. The model achieved a final validation accuracy of 99.71% and a low validation loss of 0.29%. The confusion matrix analysis showed minimal false classifications (1 wrong classification out of 340). The model proved to be effective for detecting both high and low-gain replay attacks. The proposed model outperformed other existing models in the same area of study in terms of accuracy and precision. Overall, the paper presents a highly effective model for detecting replay attacks on remote keyless controlled vehicles, with a focus on signal classification and a strong emphasis on model performance and evaluation.

The results in [9] prove that deep learning models can be used for SEI systems. And the security of an RKE system can be improved through the use of SEI systems. Although the model proved to be effective in detecting replay attacks on RKE systems, the number of transmitters was limited to one remote and one replay attack SDR.

2.4 CHAPTER SUMMARY

In this chapter, an in-depth literature study on SEI systems was conducted, with a focus on their applications and the security challenges they address. The following key concepts and discussions emerged from the investigation.

An introduction to SEI systems as a means of uniquely identifying transmitters, specifically, how SEI systems could be applied in the context of RKE systems. The accessibility and cost-effectiveness of RKE transmitters were highlighted as reasons for their selection.

Security concerns associated with RKE systems, differentiating between fixed code and rolling code access remotes were explored. Fixed code remotes transmit static codes, making them vulnerable to replay attacks, while rolling code systems rely on synchronization, but can still be exploited.

How SEI methods can enhance RKE security by analyzing both the transmitter and the digital code before granting access was discussed. This approach was presented as a more secure and cost-effective alternative compared to other solutions involving additional components.

A flow diagram of a typical SEI system, illustrating its key components, including the RF receiver, feature estimation, and classification was presented. Each component's role in the SEI process was described.

The significance of the RF receiver in SEI systems was emphasized, as the quality of data for feature extraction is highly dependent on receiver performance. Various receiver types and their impact on SEI system accuracy were discussed.

The critical role of feature estimation in SEI systems, including the need for consistent and distinct features that describe transmitter signals effectively was discussed. Different feature extraction methods were explored, along with the importance of mitigating noise and interference.

Feature analysis algorithms, such as ICA, LDA, gamma test, and ROC, were introduced as methods for reducing the dimensionality of feature data while maintaining accuracy.

Various classifiers, including SVM, KNN, NN, naive Bayes Classifiers, and K-means clusters, were presented as options for SEI systems. It was found that the choice of classifier should align with the nature of the input data. This chapter laid the foundation of the research by reviewing existing literature on SEI systems, RKE security challenges, and the crucial components of SEI technology. The insights gained will inform the design and implementation decisions discussed in Chapter 3.

CHAPTER 3 SPECIFIC EMITTER IDENTIFICATION IMPLEMENTATION

3.1 CHAPTER OBJECTIVES

Chapter 3 serves as the core of the research, offering an in-depth exploration of the methodology used to design and implement the SEI system. While Chapter 2 provided a broad overview of the SEI system's architecture, Chapter 3 delves into the intricate details of how the system is constructed and how it addresses the research questions.

This chapter provides a comprehensive survey of relevant research, theories, and methodologies related to this SEI system. It offers valuable insights into existing techniques, challenges, and trends in the field of SEI. By the end of Chapter 3, a clear understanding of how the SEI system is constructed, the rationale behind its design choices, and the integration of literature and concepts to address the research questions. This chapter serves as the bridge between the theoretical foundations established in Chapter 2 and the results showcased in Chapter 4.

This chapter commences by introducing the core components of the SEI system, highlighting the significance and implementation of RF transmitters in Section 3.2, and receivers in Section 3.3. With a clear visualization of the hardware setup, the process of capturing RF signals and subsequent signal processing in Section 3.4, shedding light on the intricacies of the SEI approach.

Feature estimation, a pivotal aspect of SEI, is explored in depth in Section 3.5. The method used to transform raw signal data into interpretable features is explained, emphasizing the need for consistent and distinctive features. The journey continues with an exploration of various feature management

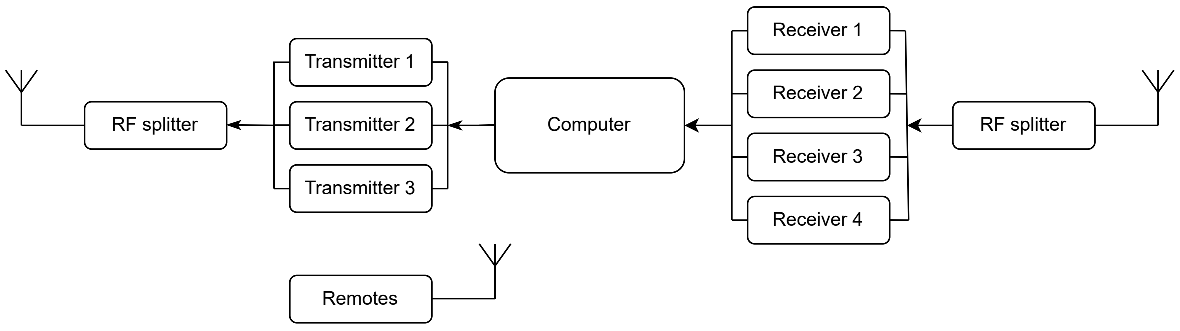


Figure 3.1. Flow diagram of the experimental configuration.

techniques in Subsection 3.5.1. The importance of reducing feature dimensionality while preserving essential information is discussed. The advantages and disadvantages of feature management are discussed, demonstrating its suitability for SEI tasks.

Section 3.6 presents a rigorous evaluation and discussion of multiple classifiers. The importance of hyperparameter tuning and cross-validation in improving classifier performance is discussed in this section as well.

The SEI system was designed and implemented in Python 3.10, and GNU radio was used for the replay attack SDRs. This software was chosen as it is open-source and the software has a large community of users, and optimized libraries that were utilized in the design. The hardware used in the SEI system and the experimental configuration of the hardware is shown in Figure 3.1.

3.2 RADIO-FREQUENCY TRANSMITTER

The transmitters used in this research can be divided into two main categories, namely the replay attack SDRs and the gate access remotes. The replay attack SDRs is used to retransmit the recorded signals, while the gate access remotes are used to transmit the original signals. The gate access remotes were all trained from a single remote ensuring that all of the remotes transmit the same signal. Two different codes were trained on all of the remotes to ensure that different codes can be used for each remote for training and testing of the SEI system.

The gate access RF transmitters are static code access remotes with a transmission frequency of 403 MHz and 434 MHz. This frequency band is part of the industrial, scientific and medical (ISM)

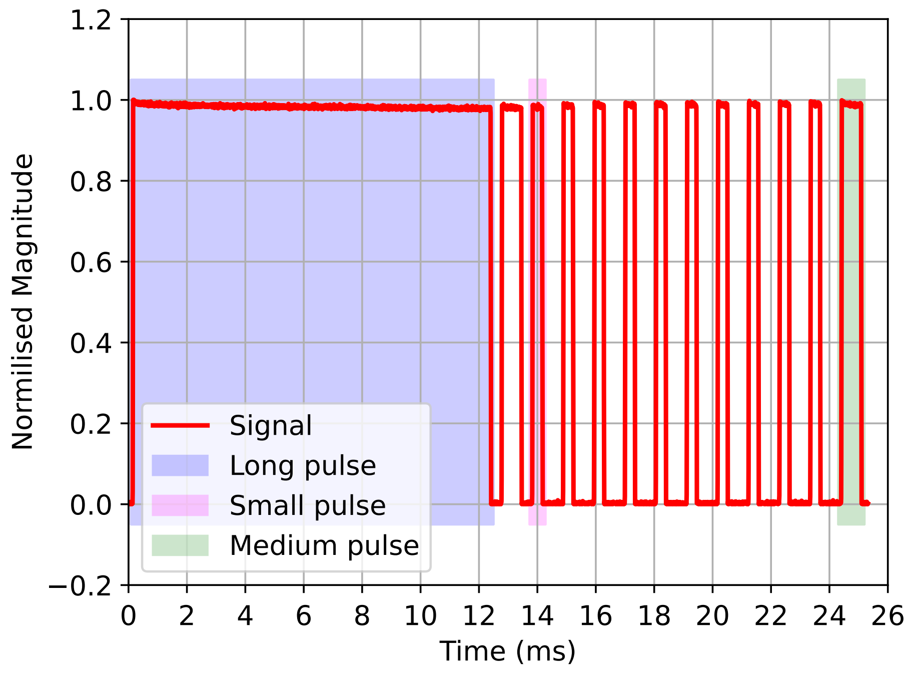


Figure 3.2. Received burst of an access remote.

band that is used for low power RF transmissions such as RF remotes, and RF transmitters for light switches [1,6]. The access remotes transmit a digital code which is modulated with a PWM signal, a very long pulse is used to indicate the start of transmission and the code follows with medium and small pulses. Where a medium pulse denotes a one and a small pulse denotes a zero code. An example of a single burst from remote 1 is shown in Figure 3.2. In the figure, the long burst indicates the start of transmission, and the medium pulses and the small pulses are shown resulting in a code of 100000000001.

In this research, a total of 13 remotes were used. Ten of the remotes were bought at the same time from the same manufacturer to ensure the same components were present in the different transmitters. The other three remotes were older remotes that were used to introduce a different transmitter into the dataset. For the 13 remotes two codes were recorded for each remote, the codes were: 100000000001 and 101010101010. The two different codes were chosen to test if the system could detect the same transmitter transmitting the two different codes. The difference in codes where chosen to ensure the number of one and zero codes are different between the two codes.

For replay attacks three different SDRs were used: A HackRF One with an operating range of 1 MHz to 6 GHz, a maximum sampling rate of 20 MSamples/s, and an 8-bit ADC and digital-to-analogue converter (DAC) [23]. An Analog Devices active learning module (ADALM) Pluto SDR with an operating frequency range of 325 MHz to 3.8 GHz, a maximum sampling rate of 61.44 MSamples/s, and a 12-bit ADC and DAC [24]. And a Lime SDR with an operating range of 100 kHz to 3.8 GHz, a maximum sampling rate of 61.44 MSamples/s, and a 12-bit ADC and DAC [25]. The replay attack SDRs were chosen as they operate in the frequency band of interest, they can be purchased off the counter, and the author had access to the hardware.

Each of the replay attack SDRs recorded a recording from a new remote: remote 1 and an old remote: remote 11 for both codes and retransmitted the recorded signals. This was implemented in GNU radio, where the recorded signals were stored in a binary file and replayed at a later stage. The experimental configuration of the replay SDRs in GNU radio is shown in Appendix A. The replay attack SDRs was connected to GNU radio software via a USB connection to the computer. The SDRs were connected to the same antenna for both receiving as well as transmission to ensure that the differences measured in the signals were due to the SDR hardware and not different antennas. The replay attack design and the values used in the GNU radio design are shown in Appendix A.

In GNU radio an SDR source block was used to receive the signals. This source block can be configured with multiple parameters, the center frequency of the source block was set to the center frequency of the gate access remotes and the sampling rate was set to 3 MSamples/s. This sampling rate was chosen to ensure that the signal will be received without aliasing as it is higher than the Nyquist sampling rate, and the receiver SDR sampling rate to ensure that the transmitter is not the sampling limiting factor.

The Nyquist sampling rate, also known as the Nyquist-Shannon sampling theorem, is a fundamental concept in digital signal processing [26]. It states that in order to accurately reconstruct a continuous signal from its samples, the sampling rate must be at least twice the highest frequency component in the signal. The Nyquist sampling rate f_s can be calculated using

$$f_s \geq 2 \cdot f_{\max}, \tag{3.1}$$

where f_s is the Nyquist sampling rate in Hz, and $f_{\max}$ is the maximum frequency component in the continuous signal in Hz. This equation ensures that there are enough samples per cycle to accurately represent the signal during the digitization process. If the sampling rate is less than twice the maximum frequency $f_{\max}$, aliasing can occur, leading to distorted or incorrect signal reconstruction. To avoid aliasing and accurately capture the signal's information, the sampling rate must meet or exceed the Nyquist rate.

The gain of each transmitter had to be changed independently as the same gain values did not result in the same SNR between different SDRs. The received signals were then stored in a binary file using a file sink block. The graphical user interface (GUI) sink block was used to display the received signals, to ensure the data was received correctly.

The recorded signals were then replayed using SDR sink block with the data from the file source block. The center frequency and sampling rate of the SDR sink block were set to the same values used to record the data. The gain of the SDR sink block was set independently to ensure a similar SNR between the different SDRs. The GUI sink block was used to display the transmitted signal to ensure that the signal was transmitted correctly.

The transmitters used in this research thus result in 13 remotes, with 3 replay attack SDRs, each transmitting two different codes. The variations in transmitters were selected to determine if the SEI system is able to differentiate between remotes with the same code, the same remote with different codes, and replay attacks. This selection of remotes covers a wide range of possible attacks on an RKE system. When transmitting the gate access remotes signals the remotes and SDRs were 3 m away from the receiver antenna to ensure that the signal was not saturated. This distance was maintained throughout the test but due to the nature of user measurements the distance between measurements might have varied slightly.

3.3 RADIO-FREQUENCY RECEIVER

The RF receiver selected for this research to receive the RF signal from the RF transmitters is the RTL28232U SDR with an R820T tuner. This SDR was used because it has an 8-bit ADC resolution with a maximum sampling rate of 2.56 MSamples/s [27]. Another advantage of the SDR is the low cost as the prices for the SDR range from $28 to $40.

In this research the effect of different RF receivers was tested on the SEI system, therefore, four RTL28232U receivers were used to record the RF signals. The four receivers were bought at the same time from the same manufacturer to ensure that the receivers have the same hardware. The hardware configuration of the receivers can be seen in Figure 3.1. The four receivers were connected to the computer via a USB connection. The receivers were connected to the same antenna to ensure that the differences measured in the signals were due to the SDR hardware and not different antennas. The receivers were controlled in Python which ensured that the receivers could be configured and controlled from a single Python program.

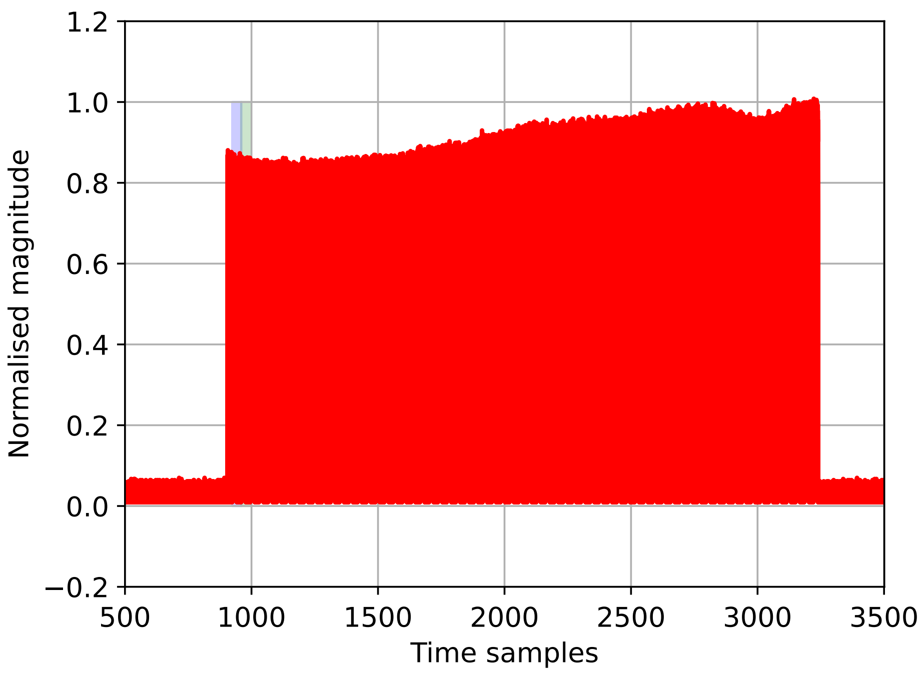
It was found that the new remote's center frequency is at 434 MHz, while the old remote's center frequency is 403 MHz. The center frequency of the receiver was adjusted when recording the specific transmitters. Due to the difference in center frequencies between the old and new remotes, frequency offset was not considered for one of the features. This decision was made as it would simplify the classifier and result in a biased solution. The sampling rate of the receivers was set to 2.4 MSamples/s, this is the highest stable sampling rate of the RTL28232U SDR. The RTL SDR can be set to a higher sampling rate but the SDR introduces artifacts and noise into the received signal, therefore it was set to 2.4 MSamples/s. The gain of the receivers was set to 32 dB to ensure a good SNR without saturating the signal. The received signals were then stored in a binary file, and this data was then used for data processing.

The four receivers were used to record ten 4-second samples of each remote-code pair with a sampling rate of 2.4 MSamples/s. The long recording time was selected to ensure that the entire transmission of the gate access remote was recorded for each receiver. The raw data of each receiver was saved to a binary file to be processed later. The data was saved in a binary file as the read and write speeds of binary files are fast while compressing the filesize as well. The data was received on the RTL SDRs in Python using the pyrtlsdr library version 0.3.0.

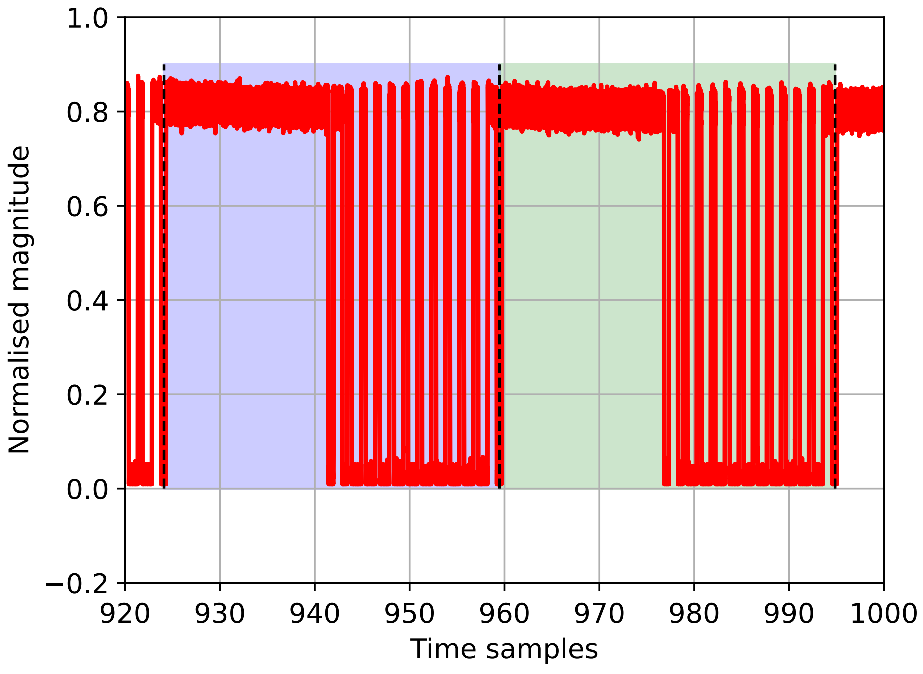
3.4 SIGNAL PROCESSING

The raw recording from the RF receiver is shown in Figure 3.3(a), this recording is a full transmission burst of remote 1. In the figure, it is seen that there are multiple code bursts in the signal. The number of bursts for the new remotes is 65 and the old remotes are 85. For the old remote recordings, the first 65 bursts were used for processing. This was done to ensure that the same number of bursts were used for each remote, to ensure that the data was balanced between the new and old remotes.

The bursts are detected and separated using a peak detection algorithm. The peak detection algorithm



(a)



(b)

Figure 3.3. Raw recording for remote 1 is shown in (a), and (b) shows the peak detection boundaries and isolated bursts.

used was the find_peaks function from the SciPy library version 1.11.2. The peak detection algorithm was implemented to determine the start and stop positions of each pulse in the signal. Each pulse was then used to implement the rest of the signal processing and feature estimation. The peak detection algorithm decision lines as well as detected bursts are shown in Figure 3.3(b).

The raw burst recordings from the RF receivers were added to a dataset for processing. The file format needs to consider the reading, and writing speed, as well as file size tradeoff. The pickle format was chosen as it compresses the filesize while maintaining a fast read and write speed.

A burst of data from remote 1 is shown in Figure 3.4(a) and the frequency response is shown in Figure 3.5(a). In Figure 3.5(a) it is seen that a frequency response of remote 1 has a frequency offset of 0.364 MHz. The received data from the RF receiver needs to be processed to remove frequency offsets from the data and to filter out the noise and unwanted frequencies. This signal was then mixed down to the center frequency by multiplying the received signal with

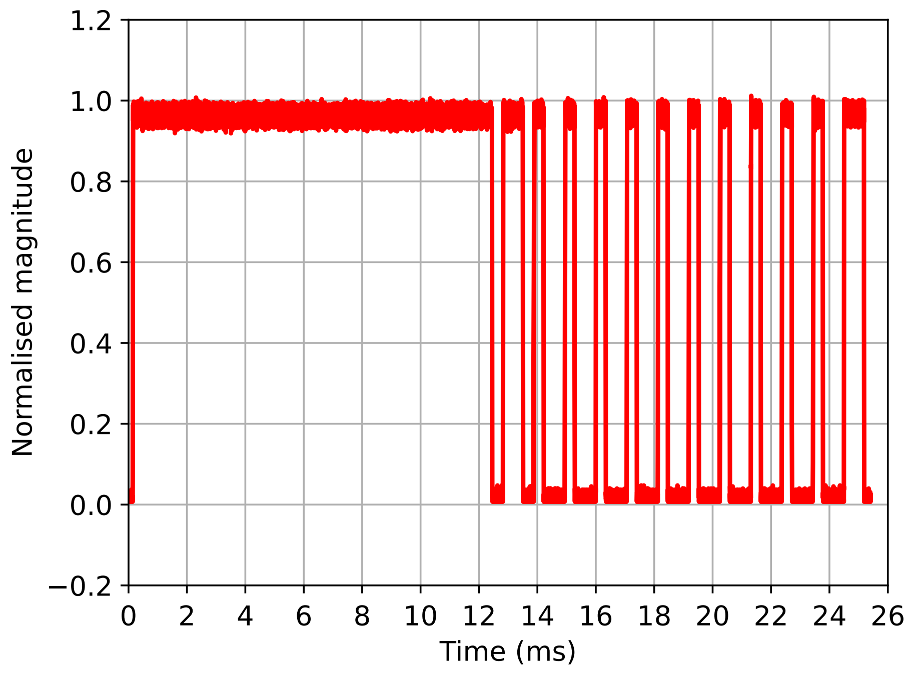
$$e^{-j2\pi f_{\text{offset}}t}. \tag{3.2}$$

The frequency offset needs to be removed for each burst of received data, as the frequency offset is not constant between bursts [1].

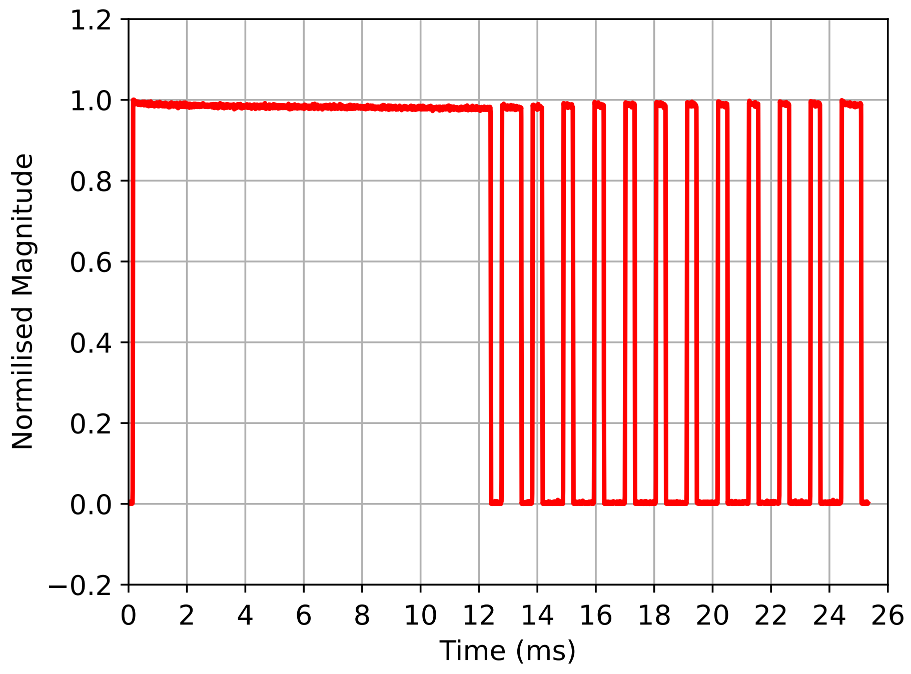
A finite impulse response (FIR) filter was applied to reduce the noise and filter out the unwanted frequency components. Typically a window function is used to filter data, several different window functions can be considered for implementation. There are advantages and disadvantages to each window function. The trade-off of the window functions that need to be considered is the relationship between the width of the center beam and the magnitude of the sidelobes [26].

FIR filters are a class of digital filters widely used in signal processing and digital communications. They are characterized by having a finite duration impulse response, which means that they only consider a finite number of past input samples to calculate the current output sample. FIR filters are known for their stability, linear phase response, and straightforward implementation.

Several advantages are associated with FIR filters, making them a valuable choice in various applications.

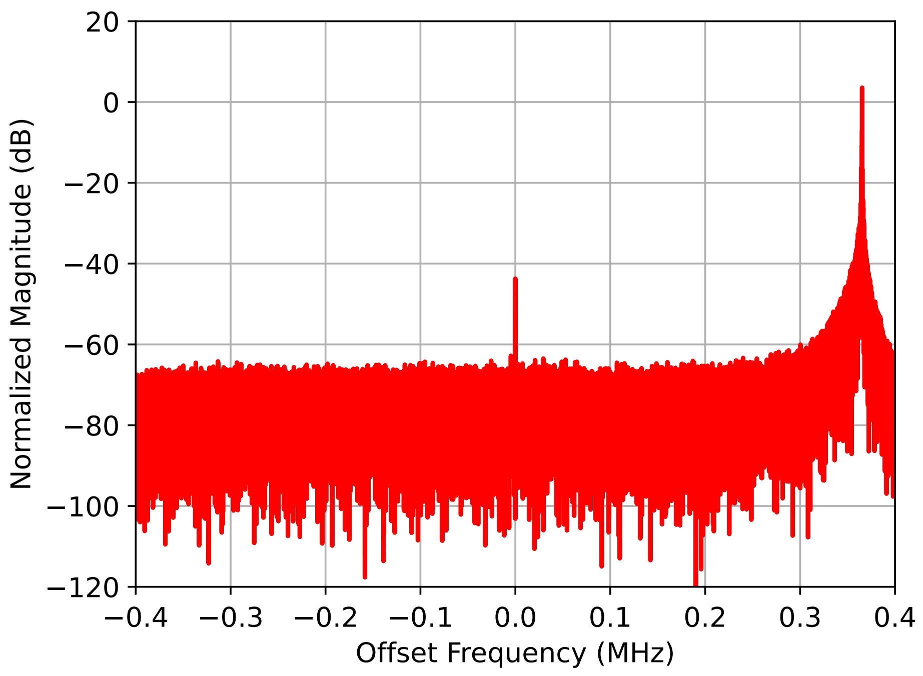


(a)

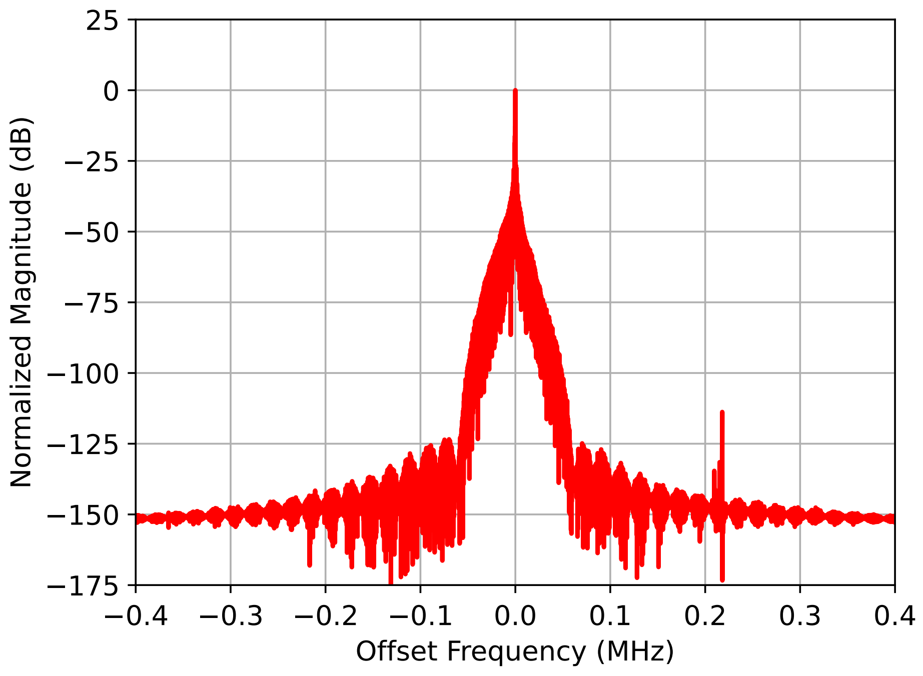


(b)

Figure 3.4. Burst of remote 1 data before processing (a), and after processing (b) in the time domain.



(a)



(b)

Figure 3.5. Burst of remote 1 data before processing (a), and after processing (b) in the frequency domain.

- FIR filters can achieve a linear phase response, which preserves the shape and timing of the input signal. This property is crucial in applications where phase distortion is undesirable, such as signal, audio and image processing [26, 28, 29].

- FIR filters are inherently stable and do not suffer from stability issues like infinite impulse response (IIR) filters can [28–30].

- FIR filters can be designed to have arbitrary frequency responses, making them versatile for various filtering requirements [26, 28–30].

- FIR filters do not have feedback loops, simplifying their implementation and analysis [26,28–31].

- Designing FIR filters is relatively straightforward and well-understood, with many established design methods [26, 28–31].

Despite their advantages, FIR filters also have some disadvantages to consider.

- FIR filters typically require a higher order (more coefficients) than IIR filters to achieve similar filter characteristics, which can lead to increased computational complexity [26, 28–31].

- Some FIR filter designs, especially those with linear phase responses, are non-causal, meaning they have a delay in processing the signal [26, 28–31].

FIR filters operate by convolving the input signal with a set of coefficients also known as the filter impulse response to produce the filtered output [26]. The filter's impulse response determines the filter's frequency response and behavior. The general formula for an FIR filter is

$$y(n) = \sum_{k=0}^{N} h(k) \cdot x(n-k), \tag{3.3}$$

where $y(n)$ is the filtered output and $x(n)$ is the input signal, at time index n. $h(k)$ is the FIR filter coefficient at index k and N is the filter order, which determines the number of coefficients.

A few window functions are shown in Figure 3.6. In the figures, it is seen that there is an inverse relationship between the time domain and frequency domain, a wide beamwidth in the time domain results in a narrow beamwidth in the frequency domain [26]. The sidelobes of the different window functions are shown in Figure 3.6(b), where the magnitude of the sidelobes is different for each of the window functions. For RF systems, a window function with low magnitude sidelobes and a narrow beamwidth is preferred [26]. This will ensure suppression of the unwanted frequencies and the low sidelobes will suppress noise in the data. A fifty coefficient FIR filter using the Blackman window was used to filter the data. The Blackman window was selected because of the trade-off between the main lobe and the low side-lobes [26]. The Blackman window's equation is

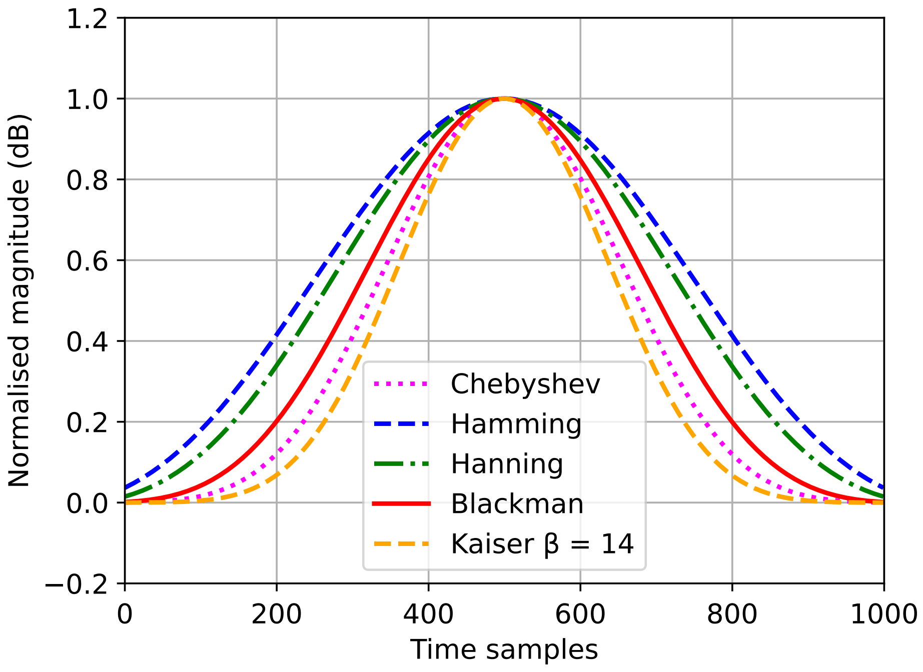
$$w(n) = 0.42 - 0.5 \times \cos\left(\frac{2\pi n}{M}\right) + 0.08 \times \cos\left(\frac{4\pi n}{M}\right), \tag{3.4}$$

where $w(n)$ is the value of the Blackman window at index n, and M is the length of the window.

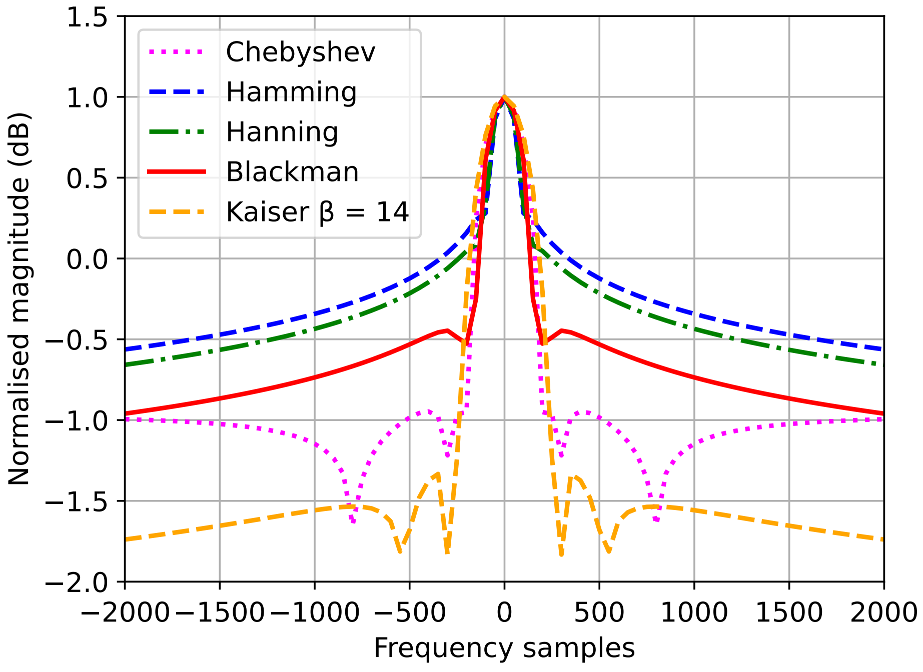
The Blackman window is often used in FIR filter design because it reduces the side lobes in the frequency response, improving the filter's performance in terms of sidelobe suppression and stopband attenuation.

The result after mixing the signal down to the baseband and filtering the data is shown in Figure 3.4(b) and the frequency response is shown in Figure 3.5(b). In Figure 3.5(b) it is seen that the signal is at the center frequency of the receiver and the noise of the signal is reduced after the filter. After the filter, the amplitude of each burst of data was normalized to ensure that there is no amplitude variance between bursts as this will affect the feature estimation. The filtered and normalized result of the data in the time domain is seen in Figure 3.4(b).

To determine the frequency data, the phase of the data is used as seen in Figure 3.7(a). It was found that when there is no transmission in the burst the phase data is random, this will harm the feature estimation. The random phase was discarded and only the phase of the transmissions was considered. The unwrapped phase of the transmission burst of remote 1 is shown in Figure 3.7(b). In the figure it is seen that there is an arbitrary phase offset between pulses, this will harm the feature estimation. To resolve this the frequency data is used instead of the phase data, this is implemented by determining the gradient of the unwrapped phase, this result is shown in Figure 3.8. In the figure, it is seen that the same information can be extracted from the phase data without the arbitrary offset.

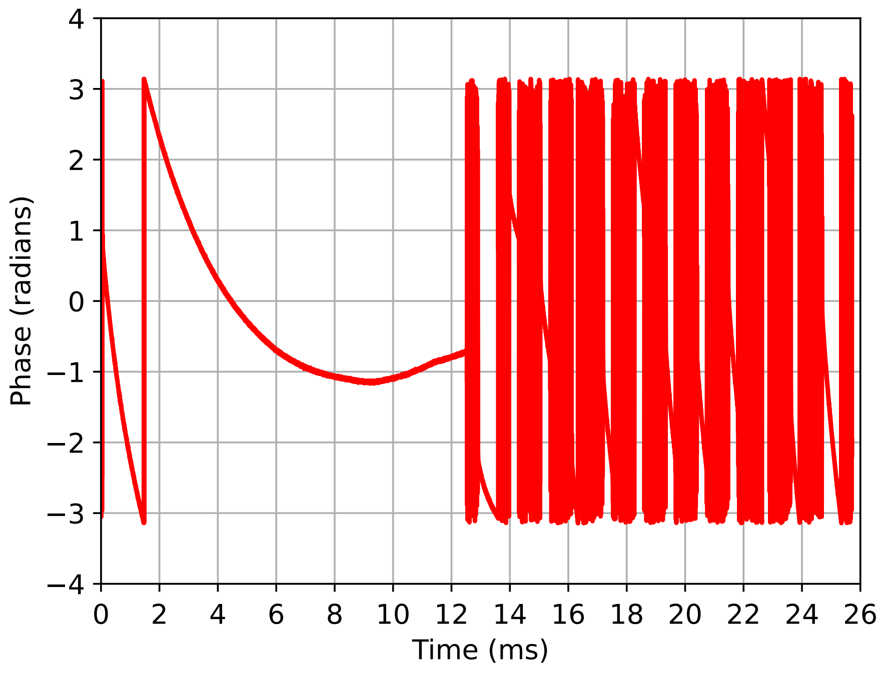


(a)

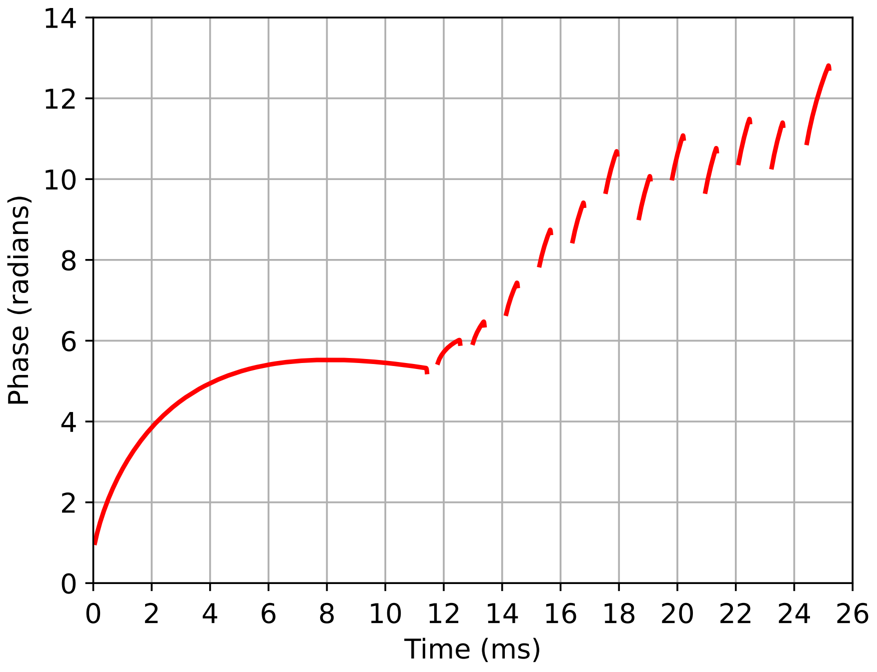


(b)

Figure 3.6. Different window functions (a) time domain, and (b) frequency domain.



(a)



(b)

Figure 3.7. Burst of remote 1 phase data before processing, where (a) is the received data, and (b) unwrapped phase excluding no transmission data.

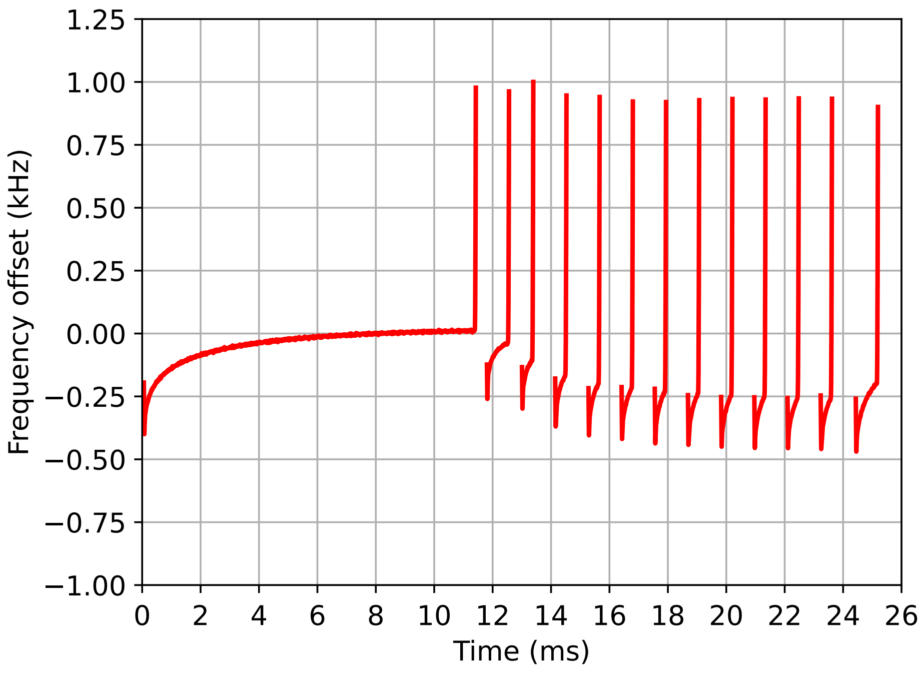
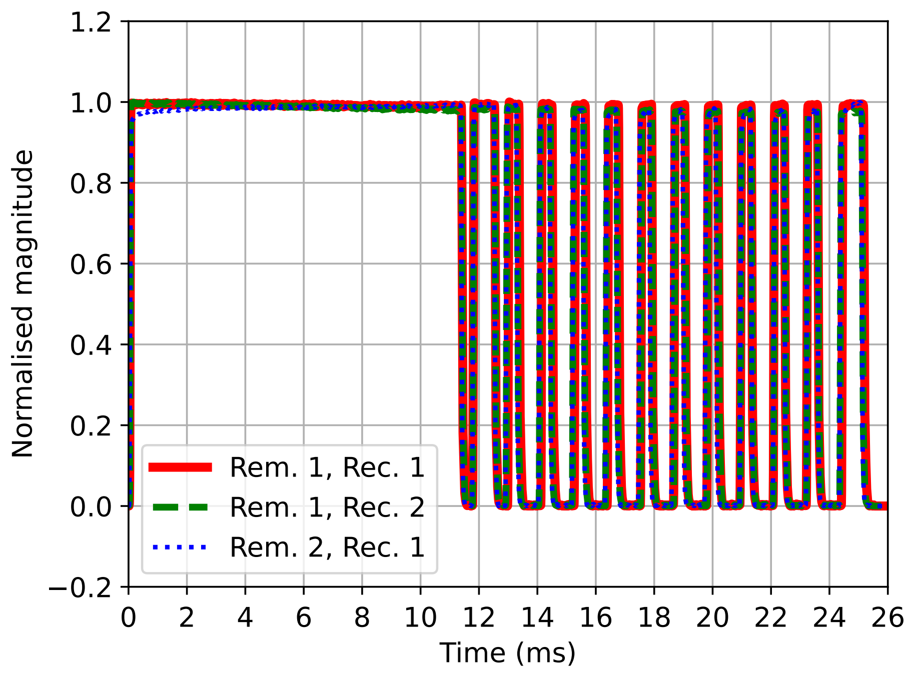


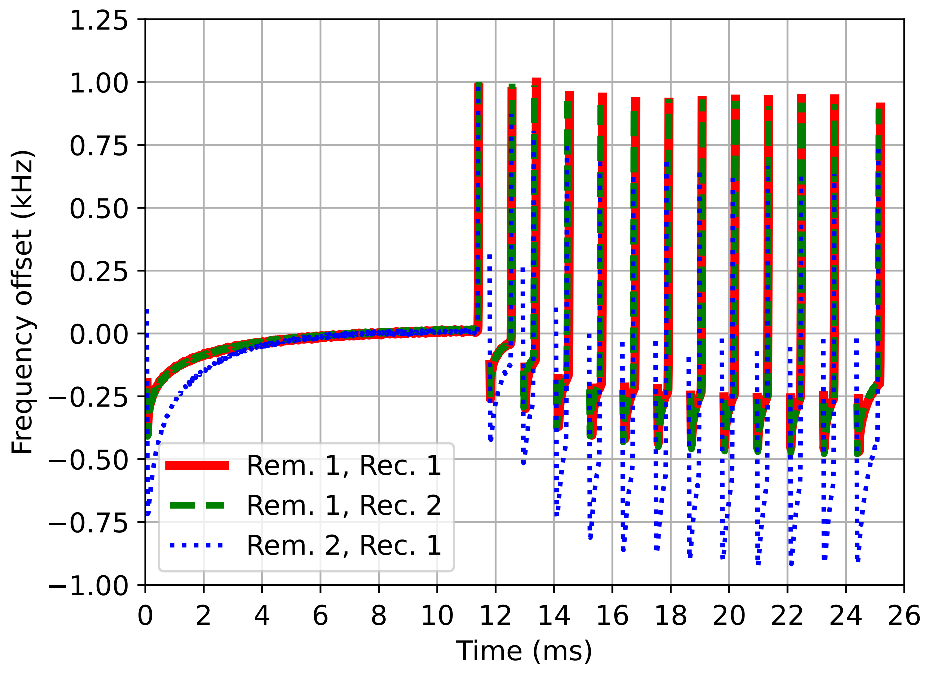
Figure 3.8. The gradient of the unwrapped phase data burst of remote 1.

3.5 FEATURE ESTIMATION

Feature estimation needs to be consistent for all the transmissions of a transmitter while being distinct from the features of other transmitters and their transmissions [1, 3, 14]. Therefore before feature estimation is implemented the processed data was inspected to ensure that feature estimation could be implemented successfully. The first burst comparison result for recording 1 and recording 2 of remote 1 is in Figures 3.9(a) and 3.9(b), where Figure 3.9(a) is the amplitude result and Figure 3.9(b) is the frequency result. The amplitude and frequency results for the first burst of remote 2 are shown as well. In Figure 3.9 it is seen that there is a significant difference between the frequency data of two different transmitters compared to the frequency difference between different recordings of the same transmitter. This shows that feature estimation can be implemented on the frequency data as there is a distinct difference between the data of different transmitters, while the data is consistent for different recordings of the same transmitter. The difference between different transmitters is not as distinct in the amplitude results, this indicates that the amplitude data will not be as effective for feature estimation as the frequency data. The small differences observed between different recordings of the same remote can be due to the heat of the components, component tolerances, multipath, or interference.



(a)



(b)

Figure 3.9. Comparison of recording one and two of remote 1, and recording one of remote 2 in (a) amplitude, and (b) frequency.

In this research, only the long transmission pulse indicating a transmission start, was used for feature estimation. This was done to determine if transmitters with different codes can be identified as the same transmitter, as long as all the different codes have the same start pulse. An example of how the burst data was used for feature estimation is shown in Figure 3.2, where only the start pulse data were used for feature estimation. The mean, variance, standard deviation (STD), skewness, kurtosis, and variance of frequency (after frequency offset was removed) were then calculated on the pulse. These statistical properties were chosen as they are commonly used in signal processing to describe a signal. Furthermore the results in [1, 3, 14] showed that these features are effective for SEI applications.

The mean μ of a set of N data points $x_1, x_2, \ldots, x_N$ is calculated as the sum of all data points divided by the number of data points [26, 32–34],

$$\mu = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{3.5}$$

Variance σ^2 measures the spread or dispersion of data points. It is calculated as the average of the squared differences between each data point and the mean [26, 32–34] giving

$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N} (x_i - \mu)^2. \tag{3.6}$$

The standard deviation σ is the square root of the variance and represents the typical or average deviation of data points from the mean [26, 32–34] using

$$\sigma = \sqrt{\sigma^2}. \tag{3.7}$$

Skewness γ_1 measures the asymmetry of the frequency distribution. Positive skewness indicates a right-skewed distribution (tail on the right), while negative skewness indicates a left-skewed distribution (tail on the left) [26, 32–34]. Skewness is calculated as

$$\gamma_1 = \frac{1}{N}\sum_{i=1}^{N} \left(\frac{x_i - \mu}{\sigma}\right)^3. \tag{3.8}$$

Kurtosis γ_2 measures the "tailedness" of the frequency distribution. It quantifies whether the distribution has heavier or lighter tails compared to a normal distribution [26, 32–34]. Kurtosis is calculated as

$$\gamma_2 = \frac{1}{N}\sum_{i=1}^{N} \left(\frac{x_i - \mu}{\sigma}\right)^4 - 3, \tag{3.9}$$

where subtraction of 3 is to normalize the kurtosis of a normal distribution to zero.

The variance in frequency of a signal measures the spread or dispersion of its frequency components. To compute this variance, a frequency analysis is required on the signal. One common approach is to use the discrete Fourier transform (DFT) to convert the signal from the time domain to the frequency domain [26, 32–34]. Once the frequency components are calculated, the variance can be calculated as follows.

- Use the DFT to transform the signal from the time domain to the frequency domain. The DFT of a discrete signal is given by

$$X(k) = \sum_{n=0}^{N-1} x(n) \times e^{-j2\pi kn/N}, \tag{3.10}$$

  where $X(k)$ is the complex DFT coefficient at frequency index k. $x(n)$ is the signal sample at time index n, N is the total number of samples in the signal, and j is the imaginary component.

- To analyze the frequency components' magnitudes, compute the magnitude $|X(k)|$ spectrum by taking the absolute value of the DFT coefficients

$$|X(k)| = \sqrt{\mathcal{Re}\{X(k)\}^2 + \mathcal{Im}\{X(k)\}^2}, \tag{3.11}$$

  where $\mathcal{Re}\{X(k)\}$ denotes the real part of $X(k)$, and $\mathcal{Im}\{X(k)\}$ is the imaginary part.

- Compute the mean frequency μ_f as a weighted average of the frequency components

$$\mu_f = \frac{\sum_{k=0}^{N-1} k \times |X(k)|}{\sum_{k=0}^{N-1} |X(k)|}. \tag{3.12}$$

- Finally, calculate the variance in frequency σ_f^2 by measuring how spread out the frequency components are around the mean frequency

$$\sigma_f^2 = \frac{\sum_{k=0}^{N-1} (k - \mu_f)^2 \times |X(k)|}{\sum_{k=0}^{N-1} |X(k)|}. \tag{3.13}$$

  This equation provides a way to quantify how dispersed the frequency components are around the mean frequency of a signal, which can be useful in various signal analysis applications.

These equations provide the means to calculate various statistical properties of a frequency distribution or a set of data points, characterizing the distribution's central tendency, spread, shape, and tail behavior of the signal.

The feature estimation was done in Python 3.10 using the stats.describe function from the SciPy library version 1.11.2, and kurtosis of the signal. Only the mean, variance, skewness, and kurtosis results were used as features from this library. The STD was determined using the np.std function from the NumPy library version 1.25. The variance of the frequency was determined by calculating the DFT of the pulse using the fast Fourier transform (FFT) function from the SciPy library version 1.11.2. The absolute value of the DFT was calculated by using the np.abs function and finally, the frequency variance was determined using the np.var function from the NumPy library version 1.25. It is important to note that the frequency offset of the signal was already removed in the signal-processing step. This results in 12 features as there is one region with six features each, for both the amplitude and frequency data. The feature data was stored in another pickle dataset.

3.5.1 Feature Management

As mentioned above it was noted that the feature estimation from the amplitude data would not be as accurate as the frequency data as the differences between the amplitude dataset is not as distinct as the frequency dataset. A feature management method was implemented to determine the best features and decrease the number of features used in the classifier. A few feature management methods were implemented and tested, the methods tested were the AUC, ICA, LDA, correlation coefficient and principal component analysis (PCA) [32, 35–39]. The feature management method selected was the PCA. PCA was chosen as the accuracy of the AUC method had similar results but the PCA computations were considerably faster. The PCA method provided a higher accuracy score for this dataset compared to ICA, LDA and the correlation method.

PCA is a dimensionality reduction technique and a fundamental tool in the field of multivariate statistics and machine learning. PCA aims to transform a dataset with potentially correlated features into a new coordinate system where the features are uncorrelated, and the variance of the data along each axis is maximized. It is widely used for data preprocessing, visualization, and noise reduction. PCA offers several advantages, making it a valuable tool for feature management.

- PCA reduces the number of features while retaining most of the original data's variance, which can be especially valuable in high-dimensional datasets [36–39].

- The new features (principal components) produced by PCA are orthogonal (uncorrelated), which can help remove redundancy in the data [35, 36, 38, 39].

- PCA can reduce the impact of noise in the data by focusing on the directions with the most significant variance [35, 36, 38, 39].

- PCA can be used for data visualization by projecting high-dimensional data into a lower-dimensional space, making it easier to explore and interpret [36–39].

- In some cases, PCA can highlight which original features contribute the most to the variance in the data, aiding in feature selection [32, 35–39].

Despite its advantages, Principal Component Analysis PCA also comes with some limitations and disadvantages.

- The new principal components may not have a straightforward interpretation in terms of the original features, making it challenging to explain the results [36, 38, 39].

- While PCA retains most of the data variance, it may still lead to some information loss, especially if only a subset of components is retained [36, 38, 39].

- PCA assumes that the relationships between variables are linear. If the data has nonlinear relationships, PCA may not be as accurate as other methods [36, 38, 39].

PCA works by finding the eigenvectors and eigenvalues of the covariance matrix of the original dataset. Implementing PCA involves several key steps, which can be summarized as follows [36, 38, 39].

- Begin by standardizing the original dataset to ensure that all features have equal importance. This step involves adjusting the data so that it has a mean of 0 and a standard deviation of 1.

- Compute the covariance matrix C for the standardized dataset. This matrix describes the relationships and variances between different features.

- Calculate the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_p$ and their corresponding eigenvectors $v_1, v_2, \ldots, v_p$ of the covariance matrix C. These eigenvalues and eigenvectors represent the directions of maximum variance in the data.

- Sort the eigenvalues in descending order and arrange the corresponding eigenvectors accordingly. This step ensures that the principal components are ranked by their importance in capturing variance.

- Choose the top k eigenvectors (principal components) that capture most of the variance in the data. The selection of k can be based on criteria like the explained variance ratio.

- Finally, project the original data onto the subspace spanned by the selected principal components. This projection transforms the data into a new feature space, where each feature (principal component) represents a linear combination of the original features. This reduced-dimensional representation can be used for analysis or visualization.

Mathematically, the projection of the data onto the first k principal components can be represented as [38, 39]

$$\mathrm{PCA}(X) = X \times V_k, \tag{3.14}$$

where X is the standardized data matrix. V_k contains the first k eigenvectors of the covariance matrix.

PCA likely provided the best results for feature management in the SEI task due to its ability to reduce feature dimensionality while preserving the most critical information. PCA effectively reduces the dimensionality of the feature space, which can help prevent overfitting and reduce computational complexity. Uncorrelated features are produced by PCA which can be beneficial when dealing with features that might be highly correlated in the original dataset. The PCA method retains the features that capture the most variance in the data [36, 38, 39]. In SEI, this might mean that the most discriminative information for emitter identification is concentrated in a lower-dimensional subspace. PCA can also help in reducing the impact of noise in the data, which can be crucial for accurate classification.

Table 3.1. Number of training and testing bursts for 16 transmitters.

Transmitter	1 to 13	14 to 16
Training	1170	2340
Testing	130	260

However, it's essential to note that the effectiveness of PCA can vary depending on the dataset and the specific problem at hand. Other dimensionality reduction techniques like ICA, LDA, and correlation coefficient analysis might excel in different scenarios, so it's crucial to consider the characteristics of the data and the goals of the classifier when selecting a feature management method.

The PCA method was implemented in Python using the PCA function from the scikit-learn decomposition library version 1.3.0. The pca.fit_transform function was implemented on the training data and the testing data to scale the data and determine the most effective features. This results in the features being ranked from most impactful to least impactful and the 10 most impactful features were used.

The feature dataset is then divided into training and testing data. The data consists of the managed features from 13 remotes, with 10 recordings and two different codes for all of the remotes. The replay attack data was also included resulting in 3 replay SDRs for two remotes with 10 recordings and two codes. The training dataset was divided as the first nine recordings of all of the transmitters, and recording ten was used for testing resulting in a 90-10 split. An example of the training and testing data split for one SDR is shown in Table 3.1.

3.6 CLASSIFICATION

Once the features have been calculated and managed the classifier can be implemented with the resulting training and testing datasets. In this research, a few classifiers were considered Ada Boost, KNN, naive Bayes, multi-layer perceptron (MLP) NN, random forest and SVM classifiers were among the classifiers considered. The result of the different classifiers is shown in Table 3.2. In the table, it is seen that the choice of the classifier has an impact on the classification accuracy.

The classifier that yielded the highest initial accuracy was the random forest classifier. The random forest classifier is a powerful and versatile machine learning algorithm that belongs to the ensemble

Table 3.2. Different classifiers used for SEI implementation.

Classifier	Accuracy
Ada boost	22.2%
Decision tree	56.5%
Gradient boosting	65.2%
KNN	62.7%
Logistic regression	45.6%
Naive Bayes	56.9%
Neural network (MLP classifier)	50.1%
Random forest	74.2%
SVM	65.3%

learning family [40–43]. It is widely used for both classification and regression tasks. The core idea behind random forest is to build a multitude of decision trees during the training phase and combine their predictions to make more accurate and robust predictions.

The random forest classifier offers several key advantages that make it a popular choice for various machine-learning tasks, including classification and regression.

- Random forest classifiers are known for their high accuracy and robustness. They often outperform single decision trees and many other classifiers in various scenarios [42, 43].
- By aggregating predictions from multiple trees, random forest reduces overfitting, making it less sensitive to noise in the data [40–43].
- Random forest provides a measure of feature importance, which helps in feature selection and understanding the relevance of different features in the classification task [42, 43].
- Random forest can handle missing data and outliers effectively without much preprocessing [42, 43].
- It is a non-parametric algorithm, meaning it doesn't make strong assumptions about the distribution of the data [40, 41, 43].

- Random forest can be easily parallelized, making it suitable for large datasets and distributed computing environments [40–43].

While the random forest classifier offers several advantages, it is important to be aware of its limitations and potential disadvantages.

- The ensemble of decision trees can make the model complex and harder to interpret compared to individual decision trees [42].
- Training a random forest with a large number of trees and features can be computationally expensive [40–43].
- Random forests may exhibit bias towards dominant classes in imbalanced datasets. Proper class balancing techniques may be required [40, 41, 43].

The random forest algorithm operates through the following steps [40–43].

- Randomly select subsets (with replacement) from the training data, creating multiple "bootstrap samples."
- For each bootstrap sample, build a decision tree. At each node of the tree, randomly select a subset of features (typically $\sqrt{\text{total features}}$), and choose the best feature to split the data based on a criterion like Gini impurity or entropy.
- During prediction, each tree in the forest independently classifies the input data point. For classification tasks, the majority class among all trees is chosen as the final prediction.

Mathematically, the prediction of the random forest classifier can be represented as [40–43]

$$\hat{Y} = \text{mode}(h_t(X)), \tag{3.15}$$

where, $\hat{Y}$ is the predicted class. $h_t(X)$ is the prediction of the t-th decision tree, and mode represents the mode (most common) class among all tree predictions.

random forest's success in SEI classification task can be attributed to its ability to handle complex and high-dimensional data effectively, reduce overfitting, and provide reliable predictions [40–43]. It can capture intricate relationships between RF signals and emitter artifacts imposed on the signals, making it a suitable choice for this application. Additionally, the ensemble nature of random forest helps mitigate the risk of model bias, which might be present in other classifiers.

The classifier results were obtained by training the models with the model.fit function, where the variables used were the training data and the training class associated with the data. The classifier was tested by calling the model.predict function, where the variable used in this function was the testing data. The random forest classifier has hyperparameters that can be adjusted to increase the accuracy. It was found for this dataset the hyperparameters of $n_{\text{estimators}} = 1000$, $max_{\text{depth}} =$ None, and $max_{\text{features}} =$ 'auto' yielded the highest accuracy of 94.5%. The hyperparameters were adjusted by using the GridSearchCV function from the scikit-learn model selection library version 1.3.0. This library allows the user to specify the hyperparameters to be tested and utilizes a k-fold cross-validation method to determine the best hyperparameters.

K-fold cross-validation is a commonly used technique in machine learning, especially when developing and evaluating classifiers.

- K-fold cross-validation assesses how well the classifier will generalize unseen data. By splitting the dataset into multiple subsets (folds) and training and testing the model on different combinations of these subsets, resulting in a more robust estimate of its performance [44–46].

- Overfitting occurs when a model performs exceptionally well on the training data but poorly on new, unseen data. K-fold cross-validation helps identify overfitting by evaluating the model on different data subsets. If the model consistently performs well on all folds, it's more likely to generalize well [44, 46].

- In many cases, datasets are limited in size. K-fold cross-validation allows one to make the most out of one's data by repeatedly using different parts for training and testing. This is particularly useful in the case of a small dataset [46].

- The performance of a machine learning model can vary depending on the specific data used for training and testing. K-fold cross-validation provides multiple performance metrics (e.g. accuracy, precision, recall) across different folds, which provides a better sense of the model's true performance [44].

- When fine-tuning hyperparameters or comparing different algorithms, K-fold cross-validation provides a more reliable way to assess which combination of hyperparameters or algorithms performs consistently well across folds [46].
- Data leakage occurs when information from the test set inadvertently influences the training process. K-fold cross-validation ensures that each data point is used for testing exactly once, reducing the risk of data leakage [44].
- In cases where the dataset has imbalanced classes (i.e. some classes have significantly fewer examples than others), K-fold cross-validation helps ensure that each class gets represented in both training and testing subsets [44–46].

However, it's important to note that K-fold cross-validation also has some disadvantages, such as increased computational cost (since the model is trained and evaluated K times) and potential sensitivity to the choice of K [44–46]. In practice, the choice of K depends on factors like the size of your dataset and computational resources.

K-fold cross-validation is a valuable tool for assessing and improving the performance of classifiers, as it provides a more robust and reliable estimate of a model's generalization performance compared to a single train-test split [44–46]. For this reason K-fold cross-validation was used to determine the best hyperparameters for the random forest classifier. The GridSearchCV function was used with 4 folds to determine the best hyperparameters for the random forest classifier. An illustration of a K-fold cross-validation method is shown in Figure 3.10. The cross-validation accuracy result of the random forest classifier is 98%. It is important to note that the cross-validation accuracy is not the same as the accuracy of the classifier. The cross-validation accuracy is expected to be higher than the accuracy of the classifier as the cross-validation is not as isolated as the testing data as shown in Figure 3.10.

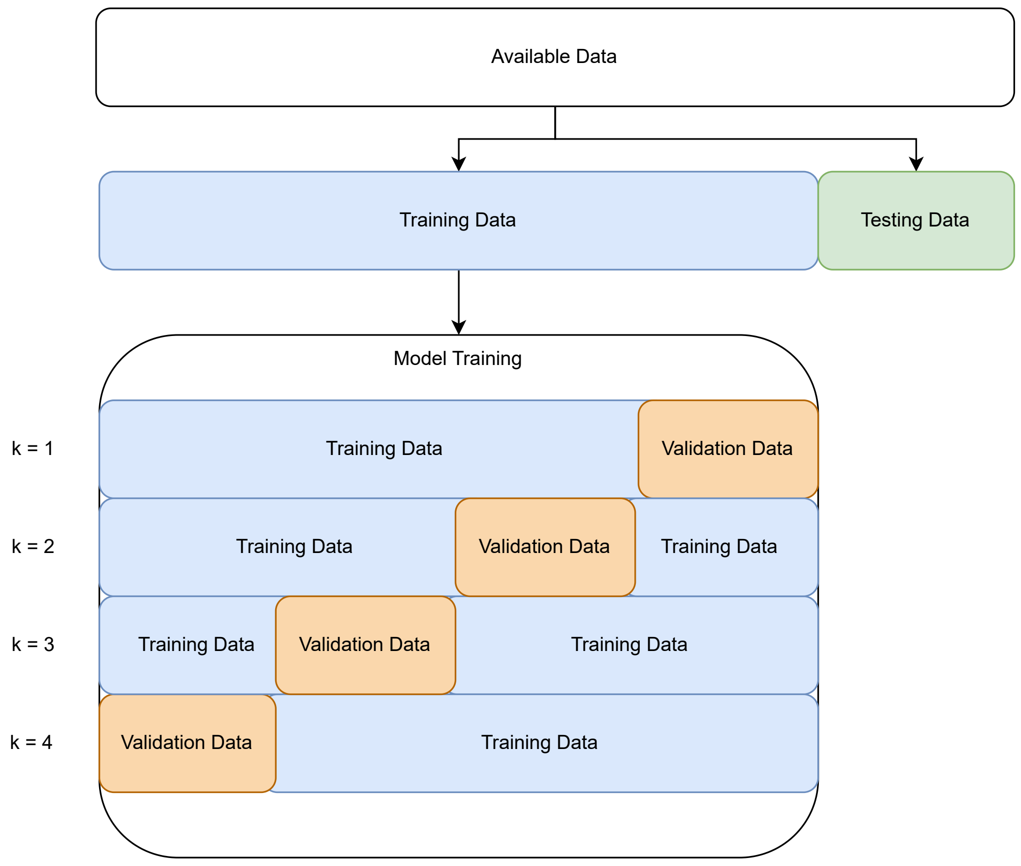


Figure 3.10. Flow diagram of a K-fold cross-validation method.

3.7 CHAPTER SUMMARY

Chapter 3 presented a comprehensive methodology for implementing the SEI system, offering insights into key decisions and findings. This chapter served as the practical core of the research, detailing the steps involved in constructing the SEI system and addressing specific research questions.

The chapter commenced by introducing the key components of the SEI system, namely the RF transmitters and receivers. It proceeded to offer a comprehensive visualization of the hardware setup, enabling a lucid comprehension of the process through which RF signals were captured and subsequently processed.

Feature estimation, a critical aspect of SEI, was explored in detail. The transformation of raw signal data into meaningful features was explained, emphasizing the importance of feature consistency and distinctiveness.

Various feature management methods were investigated. PCA emerged as the preferred technique due to its ability to reduce feature dimensionality while retaining essential information. The advantages and disadvantages of PCA were discussed, supporting its suitability for SEI tasks.

The chapter evaluated multiple classifiers, including Ada Boost, KNN, naive Bayes, MLP NN, random forest, and SVM. The random forest classifier resulted in the highest accuracy for this dataset, with its effectiveness explained. The significance of hyperparameter tuning and cross-validation in optimizing classifier performance was emphasized.

CHAPTER 4 RESULTS AND DISCUSSION

4.1 CHAPTER OVERVIEW

The primary objective of Chapter 4 is to provide an in-depth exploration and analysis of the results obtained from the implementation of the SEI system, as previously detailed in Chapter 3. This chapter builds upon the foundation laid out in the preceding chapter and is structured into three distinct sections, each offering valuable insights and findings.

Section 4.2 presents the outcomes of employing a single transmission code within the SEI system during both the training and testing phases. This section serves a critical purpose by establishing a baseline for comparison with previous research, notably, the findings reported in [1]. By examining the results in this context, a comprehensive understanding of the SEI system's performance when confronted with a single transmission code.

Section 4.3 of this chapter is dedicated to unraveling the implications and capabilities of the SEI system when multiple transmission codes are introduced into the training and testing datasets. The system's ability to differentiate between distinct transmitters, detect replay attacks, and effectively handle situations where the same transmitters emit multiple digital codes is thoroughly explored. This in-depth analysis and comparison will provide valuable insights into the SEI system's capability to enhance security by scrutinizing a broader range of transmission scenarios.

Finally, Section 4.4 explores the performance of the SEI system when subjected to variations in receiver training and testing pairs. By conducting experiments with different receiver configurations, the system's adaptability and reliability across diverse scenarios are tested. This investigation not only verifies the system's robustness but also opens up possibilities for its practical implementation in contexts involving a variety of receivers.

4.2 ONE TRANSMISSION CODE

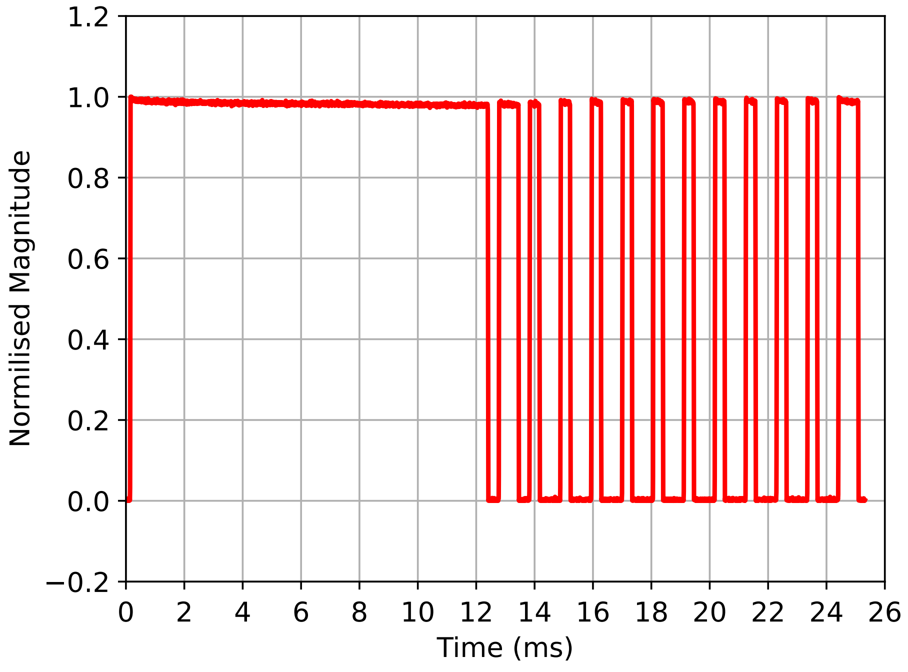


Figure 4.1. First transmission code considered.

Once the features have been calculated and managed as described in Chapter 3 the classifier can be implemented with the resulting training and testing datasets. For the first set of tests, the classifier was trained and tested with one transmission code. The transmission code is shown in Figure 4.1, this transmission code was used in the training and testing datasets in the following results. The total number of bursts for each remote used for training and testing is shown in Table 4.1. In the table, the training and testing bursts for each transmitter are shown, where transmitter 1 to 10 is the new remotes, transmitter 11 to 13 is the old remotes and transmitter 14 to 16 is the replay attacks retransmitting remote 1 and remote 11. The HackRF One is transmitter 14, the Pluto is transmitter 15 and the Lime SDR is transmitter 16.

The classifier selected in this research is the random forest classifier, this classifier results in a confidence score that a testing feature is from a known class. This confidence score can then be analyzed to determine if the confidence of the prediction is high or low. If the confidence is high the predicted class is selected but if the confidence score is low an unknown class is selected as the classifier is not

Table 4.1. Number of training and testing bursts for 16 transmitters with one transmission code.

Transmitter	1 to 13	14 to 16
Training	585	1170
Testing	65	130

Table 4.2. Confusion matrix result with 16 remotes and one transmission code.

		Predicted transmitter																	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U*	Correct
True transmitter	1	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	98.5
	2	0	62	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	95.4
	3	0	0	62	0	0	0	0	0	0	0	0	0	0	0	0	0	3	95.4
	4	0	0	0	64	0	0	0	0	0	0	0	0	0	0	0	0	1	98.5
	5	0	0	0	0	65	0	0	0	0	0	0	0	0	0	0	0	0	100
	6	0	0	0	0	0	62	0	0	0	0	0	0	0	0	0	0	3	95.4
	7	0	0	0	0	0	0	62	0	0	0	0	0	0	0	0	0	3	95.4
	8	0	0	0	0	0	0	0	65	0	0	0	0	0	0	0	0	0	100
	9	0	0	0	0	0	0	0	0	65	0	0	0	0	0	0	0	0	100
	10	0	0	0	0	0	0	0	0	0	65	0	0	0	0	0	0	0	100
	11	0	0	0	0	0	0	0	0	0	0	65	0	0	0	0	0	0	100
	12	0	0	0	0	0	0	0	0	0	0	0	65	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	65	0	0	0	0	100
	14†	0	0	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	100
	15†	0	0	0	0	0	0	0	0	0	0	0	0	0	0	130	0	0	100
	16†	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	129	1	99.2

* Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

confident in the prediction. The result of the prediction is an array of confidence scores. This array results in a confidence score for each class, where high confidence scores are larger than 60%, and low confidence scores are smaller than 60%. This property is used to create an unknown class with a score of 0.6, where the result is classified as an unknown class if a prediction is made with a low confidence score. The maximum confidence score of each row of the prediction array was calculated to result in the highest confidence class prediction for each test in the test dataset. The high confidence score of 60% was chosen to ensure that no false positive classifications were made. This creates a tradeoff where more unknown classifications are made as some classifications are below the 60% confidence level. This tradeoff is deemed acceptable as one would rather classify a detection as unknown rather than classify the transmitter incorrectly. The accuracy of the classifier is the result of the number of correctly predicted classifications over the total predictions.

The prediction result is then shown in a confusion matrix to illustrate the predicted class compared to the expected class. The confusion matrix result of one transmission code is shown in Table 4.2. If the confusion matrix in Table 4.2 is considered, the rows of the table indicate the true class of the testing data and the columns are the predicted class. If the classification of the system is perfect the diagonals would correspond with the testing data in Table 4.1. In Table 4.2 it is seen that some incorrect predictions were made, this result is expected as the system is not perfect and some wrong predictions are expected. That said the worst prediction was 62/65 = 95.38% correct, the SEI system can distinguish different transmitters, with an average accuracy of 98.6%.

False negative classifications refer to cases where known transmitters were incorrectly classified as unknown, while false positive refers to cases where unknown transmitters were incorrectly classified as known. These are important metrics for evaluating the system's performance in distinguishing between known and unknown transmitters.

The false negative rate (FNR) can be calculated by

$$\text{FNR} = \frac{\text{Number of false negatives}}{\text{Number of false negatives} + \text{Total number of true positives}}, \tag{4.1}$$

and false positive rate (FPR) by

$$\text{FPR} = \frac{\text{Number of false positives}}{\text{Number of false positives} + \text{Total number of true negatives}}. \tag{4.2}$$

In Table 4.2 it is seen that no false positive classifications were detected as the classes were correctly predicted or the confidence level was too low to predict a class. In the table, it is seen there are a few false negative classifications for remote 1 to 7, and remote 16 where the confidence level was too low to make a confident prediction. For Table 4.2, the FNR is

$$\text{FNR} = \frac{15}{15 + 1220} \times 100\% = 1.21\%. \tag{4.3}$$

From the results in Table 4.2, it is seen that the SEI system can distinguish different transmitters transmitting the same code (remotes 1 to 13), and could distinguish the transmitter from a replay attack (remotes 14 to 16). These results indicate that the system can successfully determine the transmitter with an accuracy of 98.6% if all 16 transmitters transmitting one code is included in the training data.

Table 4.3. Confusion matrix result with 8 transmitters transmitting one code, and the other transmitters are excluded from the training dataset.

		Predicted transmitter									
		1	2	3	4	5	6	7	8	U*	Correct
True transmitter	1	63	0	0	0	0	0	0	0	1	98.5
	2	0	62	0	0	0	0	0	0	3	95.4
	3	0	0	62	0	0	0	0	0	3	95.4
	4	0	0	0	64	0	0	0	0	1	98.5
	5	0	0	0	0	62	0	0	0	3	95.4
	6	0	0	0	0	0	62	0	0	3	95.4
	7	0	0	0	0	0	0	65	0	0	100
	8	0	0	0	0	0	0	0	63	2	97.0
	9-16†	0	0	0	0	0	0	0	0	715	100

* Unknown (confidence level below 60%).
† Remotes 9 to 16 were not considered during training.

Another test was conducted by removing eight transmitters from the training data and only training the classifier with eight transmitters. This result is shown in Table 4.3, this test is more accurate as there are unknown transmitters tested that were not included in the training data.

The confusion matrix result in Table 4.3 shows that the unknown remotes were correctly classified as unknown, and no false positives were calculated. This result shows that the SEI system can correctly identify the known remotes and detect the different transmitters transmitting the same code as well as replay attacks as they are not classified as one of the known remotes. For Table 4.3, the FNR is

$$\text{FNR} = \frac{16}{16+1219} \times 100\% = 1.3\%. \tag{4.4}$$

An important result to note is that although the FNR increased by 0.2% when 8 transmitters were removed from the training dataset, none of the replay attacks (transmitter 14 to 16) were classified as real transmitters (transmitter 1 to 13) in any of the tests. And no false positives were introduced into the results. This indicates that the SEI system is successful in identifying the transmitter and distinguishing between different transmitters improving the security of the system as the system can detect replay attacks.

From the results, it is shown that a 4.62% false negative classification is calculated for transmitter 2, 3, 5, 6 which is known and was classified as unknown, but the transmitters are correctly identified 95.38%.

The false classification for remote 1, and 8 is 3.07% and the correct classification of these transmitters is 96.92%. Finally the false classification for transmitter 4 is 1.54% and the correct classification of the transmitter is 98.46%. The SEI system can identify known and unknown transmitters with an average accuracy of 97.1% when 8 transmitters are excluded from the training data. This system shows the advantage of an SEI system where different transmitters can be identified to distinguish between different threats and replay attacks. The results of the SEI system in this research are similar and slightly improved compared to the result in [1] if only one transmission code is considered.

In [1] the confidence level of transmiiter 2 was too low to make a confident prediction 12 times, and for transmitter 3 once, resulting in a system accuracy of 97%. The difference in results obtained in this research and [1] could be a result of a few factors. Different data was used in this research compared to [1]. The transmitter data was captured in this research and is different compared to the data used in [1]. The signal processing implemented on the received data changes some parameters of the signal, as the noise and frequency of the received signal are changed through filtering the data. A different feature extraction, feature management and classifier implementation was used in this research compared to [1]. These factors could result in the difference in results obtained in this research compared to [1] when one transmission code is considered. Although the accuracy of the SEI system in this research is higher than the accuracy obtained in [1], the results are similar indicating that the SEI system in this research is successful in identifying different transmitters transmitting the same code.

4.3 MULTIPLE TRANSMISSION CODES

Through replicating the results in [1] it was found that the feature estimation method used in [1] would result in different transmitter classifications for the same transmitters transmitting a different code. In [1] remote 12 would be expected to be identified as remote 1 as it is the same transmitter with a different code, but the SEI system identified it as a different transmitter. This result could be due to the feature estimation method used in [1]. The five regions used for feature estimation will result in different measurements if the transmission code is different in the regions, and different feature measurements result in a different transmitter identification. In this research an alternative feature estimation method was implemented to determine if the same transmitter could be identified in the SEI system when a different code is transmitted.

The same test was performed as above with two transmission codes per transmitter. All 16 recorded

Table 4.4. Number of training and testing bursts for 16 transmitters with two transmission codes.

Transmitter	1 to 13	14 to 16
Training	1170	2340
Testing	130	260

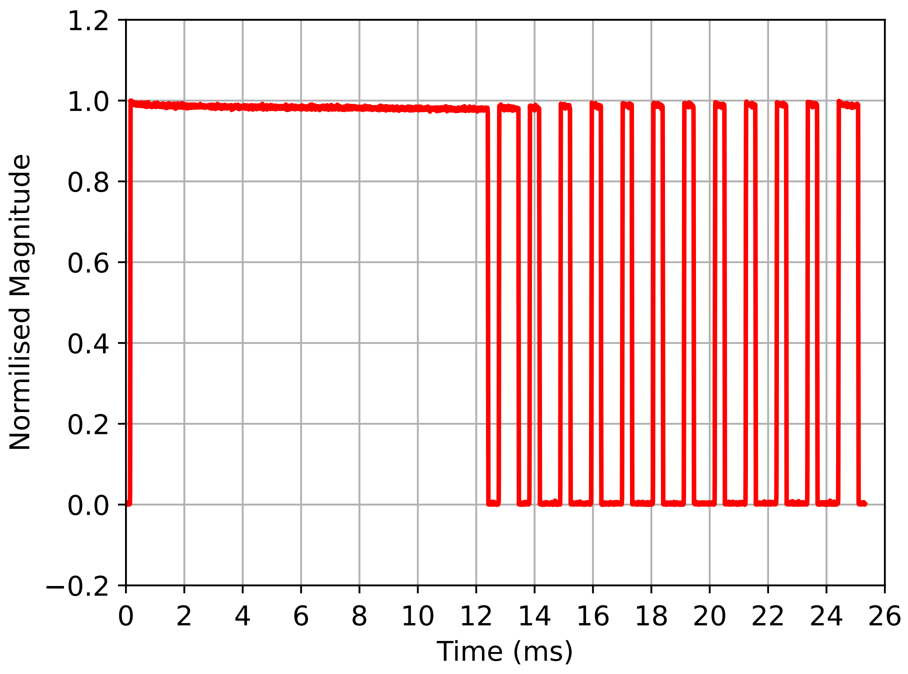
remotes with two transmission codes were used for the feature estimation. The two different transmission codes are shown in Figure 4.2 and the total number of bursts for each remote used for training and testing is shown in Table 4.4. In the table, the training and testing bursts for each transmitter are shown, where transmitter 1 to 10 is the new remotes, transmitter 11 to 13 is the old remotes and transmitter 14 to 16 is the replay attacks retransmitting remote 1 and remote 11. The HackRF One is transmitter 14, the Pluto is transmitter 15 and the Lime SDR is transmitter 16. It is seen that the number of bursts in Table 4.4 is twice the number of bursts in Table 4.1 this is due to the two transmission codes used in the training and testing data.

In Table 4.5 it is seen that the accuracy result of the system is lower compared to the result in Table 4.2, where only one transmission code was considered for each transmitter. This result indicates that the number of recordings used for training and testing the classifier could affect the system's performance. The accuracy of the SEI system using all the remotes with two transmission codes in the dataset is 95.5%. An important result to note is that no false positive classifications were detected as the classes were correctly predicted or the confidence level was too low to predict a class. This result shows that the SEI system can correctly identify the known remotes, detect transmitters transmitting different codes, as well as replay attacks as they are not classified as one of the known remotes.

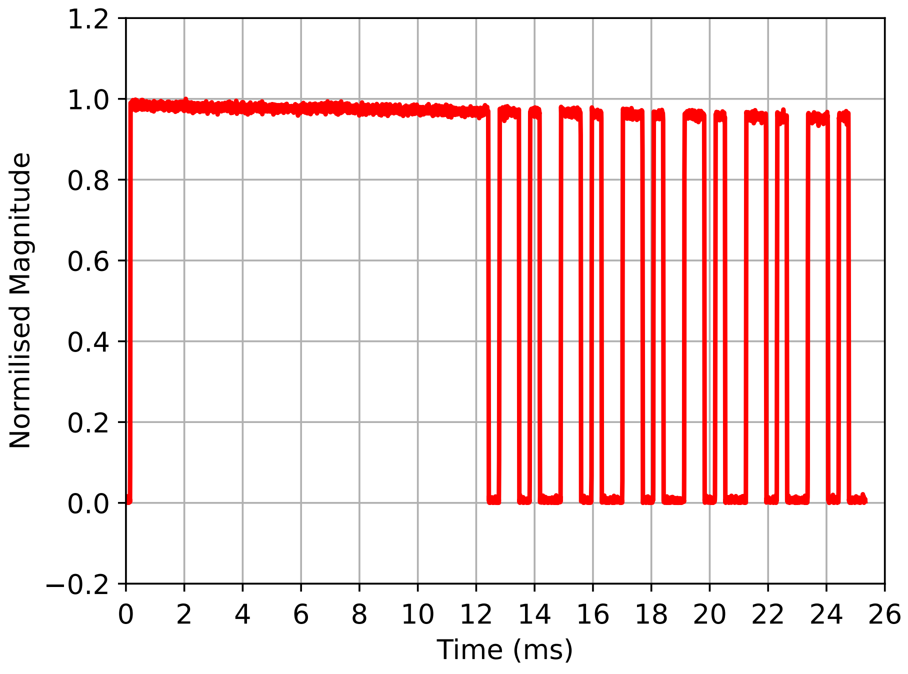
The confusion matrix result in Table 4.5 shows that although significantly more unknown classifications were made compared to Table 4.2, no false positives were classified. This result shows that the SEI system can correctly identify the known remotes and detect the different transmitters transmitting the different codes as well as replay attacks as they are not classified as one of the known remotes. Through introducing a second transmission code into the training and testing data the dataset doubled in size. For Table 4.5, the FNR is

$$\text{FNR} = \frac{97}{97+2373} \times 100\% = 3.93\%. \tag{4.5}$$

An important result to note is that although the FNR increased compared to Table 4.2 a different



(a)



(b)

Figure 4.2. Transmission codes, were (a) is code one, and (b) is code two.

Table 4.5. Confusion matrix result with all 16 transmitters with two transmission codes.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14[†]	15[†]	16[†]	U[⋆]	Correct
True transmitter	1	128	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	98.5
	2	0	115	0	0	0	0	0	0	0	0	0	0	0	0	0	0	15	88.5
	3	0	0	125	0	0	0	0	0	0	0	0	0	0	0	0	0	5	96.2
	4	0	0	0	121	0	0	0	0	0	0	0	0	0	0	0	0	9	93.1
	5	0	0	0	0	115	0	0	0	0	0	0	0	0	0	0	0	15	88.5
	6	0	0	0	0	0	108	0	0	0	0	0	0	0	0	0	0	22	83.1
	7	0	0	0	0	0	0	128	0	0	0	0	0	0	0	0	0	2	98.5
	8	0	0	0	0	0	0	0	119	0	0	0	0	0	0	0	0	11	91.5
	9	0	0	0	0	0	0	0	0	129	0	0	0	0	0	0	0	1	99.2
	10	0	0	0	0	0	0	0	0	0	127	0	0	0	0	0	0	3	97.7
	11	0	0	0	0	0	0	0	0	0	0	127	0	0	0	0	0	3	97.7
	12	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	100
	14[†]	0	0	0	0	0	0	0	0	0	0	0	0	0	260	0	0	0	100
	15[†]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	259	0	1	99.6
	16[†]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	254	6	97.7

[⋆] Unknown (confidence level below 60%).

[†] Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

transmission code is introduced into the data, doubling the size of the dataset. The system can distinguish different transmitters transmitting different codes as well as distinguish the transmitters (remotes 1 to 13) from the replay attacks (remotes 14 to 16). These results indicate that the system can successfully determine the transmitter with an average accuracy of 95.5%. Compared to the results found in Table 4.2 and [1], the accuracy of this system is lower. But the limitations in [1] were: fewer training and testing data were used, fewer transmitters, only one transmission code was used and only one replay attack SDR was considered. Given the limitations of [1] the system in this research is more robust as more data is used for training and testing, different transmission codes are considered and multiple replay attack SDRs are considered.

4.4 MULTIPLE RECEIVERS

Another test was conducted by receiving with multiple receivers and testing the effect of different receivers on the SEI system. Only two instances will be discussed in detail in this chapter the results of the other receiver train and test pairs are shown in Appendix B. This is done as the testing method is

Table 4.6. Number of training and testing bursts for 16 transmitters with multiple receivers (training with receivers 2, 3, and 4, testing with receiver 1).

Transmitter	1 to 13	14 to 16
Training	3900	7800
Testing	1300	2600

the same as discussed below and the differences are the combinations of receivers.

An example is shown below where the system is trained with the data from receiver 2, 3, and 4 and tested with receiver 1. The same method was used for testing and implementing the classier as discussed above, the differences in this test are the size of the train and test datasets are significantly increased, as well as the data from multiple receivers are considered. The number of bursts used for training and testing is shown in Table 4.6.

In Table 4.6 it is seen that the training and testing data is significantly larger compared to the data used in Table 4.4. The number of train bursts in Table 4.6 is the data from receivers 2, 3, and 4, and the number of test bursts is the data from receiver 1. All the data from each receiver is used in this instance compared to Table 4.4 where recording 1 to 9 was used for training, and recording ten was used for testing the classifier.

In Table 4.7 it is seen that the accuracy result of the system is significantly lower compared to the result in Table 4.5, where the data of one receiver was considered. The accuracy of the SEI system using multiple receivers for training and another for testing is 24.1%. A significant amount of false positive classifications are made indicating that the system is no longer effective in classifying the transmitters. The unknown classifications were 58 classifications indicating that the classifier was more than 60% confident in the false positive classifications. This result indicates that this SEI system is not effective in classifying the transmitters when multiple receivers are considered for training and another receiver is used for testing.

To determine if the significantly lower accuracy of the SEI system is due to the increase in data or if it is due to the characteristics of the receivers. The data from two receivers were used in a 90-10 train and test split. The first nine recordings from receiver 1 and 2 were used for training and recording 10 from both receivers were used for testing. The number of bursts used for training and testing the

Table 4.7. Confusion matrix were the classifier was trained with receiver 2, 3, and 4, and tested with receiver 1.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U*	Correct
True transmitter	1	269	0	14	29	0	5	0	12	612	0	0	0	0	0	232	126	1	20.7
	2	5	15	281	70	113	61	0	39	305	20	0	0	0	7	287	90	7	1.2
	3	5	60	9	425	45	563	3	3	8	80	0	0	0	0	18	79	2	0.7
	4	0	285	141	42	15	391	2	46	129	47	0	3	0	11	59	123	6	3.2
	5	23	41	31	112	134	32	10	32	364	1	0	0	0	1	230	279	10	10.3
	6	5	1	57	196	16	421	9	41	248	79	0	0	1	8	36	167	15	32.4
	7	0	4	20	11	3	558	11	202	480	0	0	0	0	0	10	0	1	0.8
	8	15	46	5	194	138	574	13	21	123	12	0	0	0	0	18	136	5	1.6
	9	26	54	10	201	75	432	10	20	134	72	0	0	0	0	26	236	4	10.3
	10	1	3	2	116	3	70	14	25	235	691	0	0	0	0	17	116	7	53.4
	11	0	0	0	0	0	0	0	0	0	0	1250	3	0	0	47	0	0	96.2
	12	0	0	0	0	0	0	0	0	0	0	70	1210	0	1	19	0	0	93.1
	13	0	0	0	0	0	0	0	0	0	0	0	0	163	560	0	577	0	12.5
	14†	0	0	0	317	0	487	0	0	0	0	1003	192	28	553	20	0	0	21.3
	15†	414	0	0	69	2	96	0	4	7	2	605	0	0	0	702	699	0	27
	16†	349	0	37	7	0	30	0	30	330	7	849	41	0	7	39	874	0	33.6

* Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table 4.8. Number of training and testing bursts for 16 transmitters training the classifier with the first nine recordings of receiver 1 and 2, and testing with recording 10 of both receivers.

Transmitter	1 to 13	14 to 16
Training	2340	4680
Testing	260	520

Table 4.9. Confusion matrix where the classifier was trained with the first nine recordings of receiver 1 and 2, and tested with recording 10 of both receivers.

		Predicted transmitter																	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U⋆	Correct
True transmitter	1	248	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	12	95.4
	2	0	213	0	0	0	0	0	0	0	0	0	0	0	0	0	0	47	81.9
	3	0	0	239	0	0	0	0	0	0	0	0	0	0	0	0	0	21	91.9
	4	0	0	0	237	0	0	0	0	0	0	0	0	0	0	0	0	23	91.2
	5	0	0	0	0	214	0	0	0	0	0	0	0	0	0	0	0	46	82.3
	6	0	0	0	0	0	213	0	0	0	0	0	0	0	0	0	0	47	81.9
	7	0	0	0	0	0	0	239	0	0	0	0	0	0	0	0	0	21	91.9
	8	0	0	0	0	0	0	0	232	0	0	0	0	0	0	0	0	28	89.2
	9	0	0	0	0	0	0	0	0	240	0	0	0	0	0	0	0	20	92.3
	10	0	0	0	0	0	0	0	0	0	255	0	0	0	0	0	0	5	98.1
	11	0	0	0	0	0	0	0	0	0	0	251	0	0	0	0	0	9	96.5
	12	0	0	0	0	0	0	0	0	0	0	0	260	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	260	0	0	0	0	100
	14†	0	0	0	0	0	0	0	0	0	0	0	0	0	520	0	0	0	100
	15†	0	0	0	0	0	0	0	0	0	0	0	0	0	0	519	0	1	99.8
	16†	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	505	15	97.1

⋆ Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

classifier is shown in Table 4.8.

In Table 4.8 it is seen that the number of bursts used for training and testing is double that of Table 4.4. This is expected as the data from two receivers are now used in training and testing the system. This will indicate if increasing the data size will affect the accuracy of the system.

Table 4.9 shows the confusion matrix result of the classifier trained with the first nine recordings of receiver 1 and receiver 2 and tested with recording 10 of both receivers. The average accuracy of the classifier trained and tested with the data of two receivers in a 90-10 train test split is 92.8%.

As with Table 4.5, the results in Table 4.9 indicate no false positives were classified. This result shows that the SEI system is successful when multiple receivers are considered. For Table 4.9, the FNR is

$$\mathrm{FNR} = \frac{295}{295+4645} \times 100\% = 5.97\%. \tag{4.6}$$

An important result to note is that although the FNR increased compared to Table 4.5 a different receiver is introduced into the data, doubling the size of the dataset. This result indicates that the accuracy of the system is not affected by the increase in data size. The accuracy of the system decreased by 2.7% compared to Table 4.5. This result indicates that the significant reduction in the accuracy of the system in Table 4.7 is due to the characteristics of the receivers and not the increase in data size.

The characteristics of both receivers are trained in with the data in this test. By using both the receivers to train the classifier and using both receivers to test the classifier the characteristics of the receiver are trained into the classifier. With the previous test done in Table 4.7 the characteristics of the receiver used for testing were not trained into the classifier and the accuracy of the system was significantly lower. This result indicates that the characteristics of the receiver affect the accuracy of the SEI system.

Other receiver test configurations were performed using both of the two test methods shown above and the results of these tests are shown in Appendix B. Table 4.10 is a summary of the receiver combinations and their accuracy scores of the system.

In Table 4.10 it is seen that the accuracy of the system is consistent when one receiver is considered for training and testing the classifier. Furthermore, Table 4.10 shows that the accuracy of the system is rather stable when multiple receivers are used. The accuracy of the system is significantly lower when the data from multiple receivers are used for training and another receiver is used for testing. This result indicates that the characteristics of the receiver affect the accuracy of the SEI system and that the characteristics of the receiver need to be considered as well when implementing an SEI system.

4.5 CHAPTER SUMMARY

This chapter presents the results of the SEI system and is divided into three sections.

Table 4.10. The accuracy score summary of several SEI systems with different receiver combinations used for training and testing.

Train receiver(s)				Test receiver(s)				Accuracy
1	2	3	4	1	2	3	4	
90%				10%				95.5%
	90%				10%			95.4%
		90%				10%		95.3%
			90%				10%	95.5%
90%	90%			10%	10%			92.8%
90%	90%	90%		10%	10%	10%		91.9%
90%	90%	90%	90%	10%	10%	10%	10%	91.2%
100%	100%	100%					100%	25.6%
100%	100%		100%			100%		32.8%
100%		100%	100%		100%			26.3%
	100%	100%	100%	100%				24.1%

In the first section of Chapter 4, the SEI system's performance with a single transmission code in both the training and testing datasets was assessed. The results revealed a high accuracy rate of 98.6%, establishing a vital baseline for further comparisons. This result is comparable with several other SEI systems, demonstrating the system's effectiveness. This section provided insights into the system's accuracy in authenticating transmissions from a single digital code. Moreover, eight transmitters were deliberately removed from the training dataset to test the system's robustness. The results indicated that the system successfully classified the untrained transmitters as unknown. This outcome highlights the system's ability to distinguish unknown transmitters from known ones and effectively mitigate replay attacks.

Moving to the second section, the performance of the SEI system was examined when confronted with multiple digital codes. Here, the system showcased its robustness, achieving an impressive accuracy rate of 95.5%. It effectively distinguished between different transmitters, identified replay attacks, and handled instances where the same transmitter sent multiple digital codes. This section demonstrated the system's potential to enhance security across diverse transmission scenarios. This result was not achieved in previous SEI systems, as it was a limitation of the systems.

In the third and final section, the SEI system's adaptability to various receiver training and testing pair configurations was tested. Through rigorous experimentation, it was observed that the system's accuracy depends on the specific receiver used for training and testing. When a different receiver was used for testing than the one used for training, the system proved to be ineffective, achieving an accuracy rate of 24.1%. However, when multiple receivers were used for training and testing in a 90-10 train-test split, and the receiver's characteristics were trained into the classifier, the system consistently maintained a high accuracy rate of 92.8%. This investigation underscores the system's reliability across diverse receiver setups, making it a viable choice for practical implementation in scenarios involving varying receiver configurations.

CHAPTER 5 CONCLUSION

In conclusion, the specific emitter identification (SEI) system developed in this research represents a significant milestone in the realm of secure transmission systems. The foundation for the SEI system was laid by expounding on the fundamental principles and concepts underpinning secure transmission systems. Various cryptographic techniques were introduced, setting the stage for the system's design and implementation. The decision to adopt a signal extraction approach, as discussed here, established the groundwork for subsequent phases.

The choice to utilize the random forest classifier for the classifier was backed up with tests and results and proved it the most accurate and efficient option. The decision to use the classifier was rooted in pattern recognition capabilities. This decision allowed to effectively train the system to effectively recognize transmission characteristics. Additionally, the choice of a 90-10 train-test split for receiver training was made, increasing the system's performance.

The results obtained in this research were divided into three distinct sections offering a comprehensive assessment of the SEI system's performance. The system's performance with a single transmission code was assessed, yielding an impressive accuracy of 98.6%. This result served as a critical baseline for further comparisons and highlighted the system's efficiency in authenticating transmissions from a single code source. The accuracy of this test is comparable to that of other SEI systems as only one digital code and receiver is considered. The accuracy of the SEI system proved to be as accurate as previous SEI systems while considering more transmitters and replay attacks.

When confronted with multiple digital codes, the SEI system maintained an impressive accuracy rate of 95.4%. It effectively differentiated between distinct transmitters, discerned replay attacks, and handled situations where the same transmitter sent multiple digital codes. The feature extraction technique was found to be a critical factor in the system's performance when confronted with multiple digital codes. The feature extraction was implemented on a start pulse that was consistent across all digital codes and transmitters. This approach allowed the system to extract the transmitters' unique features irrespective of the digital code being transmitted.

The system's adaptability to various receiver configurations was tested. The performance was found to be dependent on the receiver used for training. When the receiver's characteristics were trained into the classifier, the system demonstrated consistent and accurate results, with an accuracy rate of 92%. While the system's performance was significantly reduced when the receiver's characteristics were not trained into the classifier, it achieved an accuracy rate of less than 33%. This result highlighted the importance of training the receiver's characteristics into the classifier and emphasized the system's reliance on the receiver's characteristics.

The decisions made during the implementation of the SEI system, as detailed in Chapters 2, and 3, and the results shown in Chapter 4, collectively support the assertion that the SEI system is a robust and adaptable solution for enhancing secure transmission systems. Its ability to achieve high accuracy rates, distinguish between different transmitters, detect replay attacks, and accommodate various receiver configurations underscores its significance in the realm of secure communications. This work serves as a valuable contribution to the field, paving the way for the practical implementation of the SEI system in real-world scenarios, ultimately bolstering security and trust in transmission systems.

5.1 FUTURE WORK

In the pursuit of enhancing the SEI system's capabilities and expanding its applicability, several avenues for future research emerge. First, future investigations could explore the impact of a higher sampling rate during the reception and retransmission phases, particularly in the context of replay attacks using software-defined radios (SDRs). By conducting experiments with increased sampling rates, researchers can assess whether capturing more signal resolution improves the system's ability to incorporate distinctive signal characteristics.

Additionally, the SEI system's robustness to varying noise levels could be examined further. Currently, the results presented in this research are based on recordings with consistent noise levels. Introducing different noise levels into the dataset would provide valuable insights into how environmental noise affects the accuracy and reliability of the SEI system under realistic conditions.

Moreover, future work could explore alternative methods for feature extraction, such as utilizing spectrogram images of the received data. Convolutional neural networks (CNNs) could be employed as classifiers, as demonstrated effectively in related studies [3, 9, 15, 22]. This approach not only has the potential to enhance the system's accuracy but also significantly reduces the dataset's file size and expedites classifier training. The application of a CNN in the SEI system represents a promising avenue for research.

Lastly, to foster collaboration and innovation in the field, the dataset generated and used in this research will be made accessible to the broader research community. This dataset would enable other researchers to explore different methodologies and techniques, thereby facilitating a comprehensive understanding of how various methods impact the SEI system's accuracy and performance in diverse scenarios. The results obtained from exploring these avenues can then be compared to the results obtained in this research, thereby providing a benchmark for future studies.

REFERENCES

[1] J. Samuel and W. du Plessis, “Specific emitter identification for enhanced access control security,” *SAIEE Afr. Res. J.*, vol. 108, no. 2, pp. 71–79, Jun. 2017.

[2] A. Kawalec and R. Owczarek, “Specific emitter identification using intrapulse data,” in *Eur. Radar Conf.*, Amsterdam, Netherlands, Oct. 2004, pp. 249–252.

[3] K. I. Talbot, P. R. Duley, and M. H. Hyatt, “Specific emitter identification and verification,” *Technol. Rev. J.*, vol. 11, pp. 113–133, Jun. 2003.

[4] M. D. Williams, M. A. Temple, and D. R. Reising, “Augmenting bit-level network security using physical layer RF-DNA fingerprinting,” in *IEEE Glob. Telecommun. Conf. (GLOBECOM)*, Miami, FL, USA, Dec. 2010, pp. 1–6.

[5] K. Greene, D. Rodgers, H. Dykhuizen, K. McNeil, Q. Niyaz, and K. A. Shamaileh, “Timestamp-based defense mechanism against replay attack in remote keyless entry systems,” in *IEEE Int. Conf. Consum. Electron. (ICCE)*, Las Vegas, NV, USA, Jan. 2020, pp. 1–4.

[6] S. van de Beek and F. Leferink, “Vulnerability of remote keyless-entry systems against pulsed electromagnetic interference and possible improvements,” *IEEE Trans. Electromagn. Compat. (EMC)*, vol. 58, no. 4, pp. 1259–1265, Aug. 2016.

[7] O. Alaca, A. Boyaci, S. Yarkan, and M. A. Aydn, “A statistical modulation type identifier for remote keyless entry transmitters based on extended energy detector,” in *Int. Symp. Digit. Forensics Secur. (ISDFS)*, no. 7, Barcelos, Portugal, Jun. 2019, pp. 1–6.

[8] K. S. KiranRaj, C. Ananya, and S. Ratan, "Analysing remote keyless entity systems," *Int. J. Research Anal. Rev.*, vol. 6, pp. 136–138, Jun. 2019.

[9] A.-H. Abu, Qasem, and A. A. Alsulami, "Detection of fake replay attack signals on remote keyless controlled vehicles using pre-trained deep neural network," *Electronics*, vol. 20, pp. 2079–9292, Nov. 2022.

[10] E. Hamadaqa, A. Mars, W. Adi, and S. Mulhem, "Clone-resistant vehicular RKE by deploying SUC," in *Int. Conf. Emerg. Secur. Technol. (EST)*, no. 7, Canterbury, UK, Sep. 2017, pp. 221–225.

[11] G. Kyle, R. Deven, D. Henry, N. Quamar, A. S. Khair, and D. Vijay, "A defense mechanism against replay attack in remote keyless entry systems using timestamping and XOR logic," *IEEE Consum. Electron. (CE)*, vol. 10, pp. 101–108, Jan. 2021.

[12] S. Deng, Z. Huang, and X. Wang, "A novel specific emitter identification method based on radio frequency fingerprints," in *IEEE Int. Conf. Comput. Intell. Appl. (ICCIA)*, no. 2, Beijing, China, Sep. 2017, pp. 368–371.

[13] S. D'Agostino, "Specific emitter identification based on amplitude features," in *IEEE Int. Conf. Signal Image Process. Appl. (ICSIPA)*, Kuala Lumpur, Malaysia, Oct. 2015, pp. 350–354.

[14] M. Conning and F. Potgieter, "Analysis of measured radar data for specific emitter identification," in *IEEE Radar Conf.*, Arlington, VA, USA, May 2010, pp. 35–38.

[15] S. U. Rehman, K. W. Sowerby, S. Alam, and I. Ardekani, "Radio frequency fingerprinting and its challenges," in *IEEE Conf. Commun. Network Secur.*, San Francisco, CA, USA, Oct. 2014, pp. 496–497.

[16] N. Soltanieh, Y. Norouzi, Y. Yang, and N. C. Karmakar, "A review of radio frequency fingerprinting techniques," *IEEE J. Radio Freq. Identif.*, vol. 4, no. 3, pp. 222–233, Sep. 2020.

[17] Y. Ren, L. Peng, W. Bai, and J. Yu, "A practical study of channel influence on radio frequency fingerprint features," in *IEEE Int. Conf. Electron. Commun. Eng. (ICECE)*, Xi'an, China, Dec. 2018, pp. 1–7.

[18] S. U. Rehman, K. Sowerby, and C. Coghill, "Analysis of receiver front end on the performance of RF fingerprinting," in *IEEE Int. Symp. Pers. Indoor Mob. Radio Commun. (PIMRC)*, no. 23, Sydney, NSW, Australia, Sep. 2012, pp. 2494–2499.

[19] X. Jin-yong, Z. Hang-sheng, and L. Tao, "Method of empirical mode decompositions in radio frequency fingerprint," in *Int. Conf. Microw. Millim. Wave Technol.*, Chengdu, China, May 2010, pp. 1275–1278.

[20] X.-L. Zhang and C. Jiang, "Improved SVM for learning multi-class domains with ROC evaluation," in *Int. Conf. Mach. Learn. Cybern.*, vol. 5, Hong Kong, China, Aug. 2007, pp. 2891–2896.

[21] D. Reising, T. M. A, and M. M. J., "Improved wireless security for GMSK-based devices using RF fingerprinting," *Int. J. Electron. Secur. Digit. Forensics*, vol. 3, no. 1, pp. 41–59, Mar. 2010.

[22] S. J. Russell and P. Norvig, *Artificial intelligence: a modern approach*, 3rd ed. Hoboken, New Jersey: Pearson, 2010.

[23] (2023, Jul.) Great Scott Gadgets – HackRF One. [Online]. Available: https://greatscottgadgets.com/hackrf/one/

[24] (2021, Feb.) ADALM-Pluto detailed specifications. [Online]. Available: https://wiki.analog.com/university/tools/pluto/devs/specs

[25] (2023, Jul.) LimeSDR-USB. [Online]. Available: https://wiki.myriadrf.org/LimeSDR-USB

[26] P. Michael, *Digital Signal Processing 101: Everything You Need to Know to Get Started*. Boston, NY: Elsevier Science & Technology, 2010.

[27] (2023, Jul.) RTL-SDR blog v3 datasheet. [Online]. Available: https://www.rtl-sdr.com/wp-content/uploads/2018/02/RTL-SDR-Blog-V3-Datasheet.pdf

[28] H. Samueli, “An improved search algorithm for the design of multiplierless FIR filters with powers-of-two coefficients,” *IEEE Trans. Circuits Syst. (CSI)*, vol. 36, pp. 1044–1047, Jul. 1989.

[29] T. Saramaki, T. Neuvo, and S. Mitra, “Design of computationally efficient interpolated FIR filters,” *IEEE Trans. Circuits Syst. (CSI)*, vol. 35, pp. 70–88, Jan. 1988.

[30] X. Chen and T. Parks, “Design of FIR filters in the complex domain,” *IEEE Trans. Acoust., Speech, Signal Process. (ASSP)*, vol. 35, pp. 144–153, Feb. 1987.

[31] Z.-J. Mou and P. Duhamel, “Short-length FIR filters and their use in fast nonrecursive filtering,” *IEEE Trans. Signal Process. (SP)*, vol. 39, pp. 1322–1332, Jun. 1991.

[32] M. L. D. Wong and A. K. Nandi, “Automatic digital modulation recognition using spectral and statistical features with multi-layer perceptrons,” in *Int. Symp. Signal Process. Appl.*, vol. 2, Aug. 2001, pp. 390–393.

[33] C. Hory, N. Martin, and A. Chehikian, “Spectrogram segmentation by means of statistical features for non-stationary signal interpretation,” *IEEE Trans. Signal Process. (SP)*, vol. 50, pp. 2915–2925, Dec. 2002.

[34] A.-S. Wessam, L. Yan, and W. Peng, “Detecting sleep spindles in EEGs using wavelet Fourier analysis and statistical features,” *Biomed. Signal Process. Control (BSPC)*, vol. 48, pp. 80–92, Feb. 2019.

[35] M. Bahrololum, E. Salahi, and M. Khaleghi, “Machine learning techniques for feature reduction in intrusion detection systems: A comparison,” in *Int. Conf. Computer Sci. Conver. Inf. Technol.*, Seoul, Korea (South), Nov. 2009, pp. 1091–1095.

[36] S. Khalid, T. Khalil, and S. Nasreen, “A survey of feature selection and feature extraction techniques in machine learning,” in *Sci. Inf. Conf.*, London, UK, Aug. 2014, pp. 372–378.

[37] M. Saurabh and S. Neelam, "Intrusion detection using naive Bayes classifier with feature reduction," *2nd Int. Conf. Computer, Commun., Control Inf. Technol.*, vol. 4, pp. 119–128, Feb. 2012.

[38] P. Uddin, A. Mamun, and A. Hossain, "PCA-based feature reduction for hyperspectral remote sensing image classification," *IETE Technical Rev.*, vol. 38, pp. 377–396, Mar. 2020.

[39] P. A. Kumari and G. J. Suma, "An experimental study of feature reduction using PCA in multi-biometric systems based on feature level fusion," in *Int. Conf. Adv. Electr., Electron. Syst Eng.*, vol. 16, Mar. 2016, pp. 109–114.

[40] M. Pal, "Random forest classifier for remote sensing classification," *Int. J. Remote Sens.*, vol. 26, pp. 217–222, Oct. 2003.

[41] B. Mariana and D. Lucian, "Random forest in remote sensing: A review of applications and future directions," *ISPRS J. Photogrammetry, Remote Sens.*, vol. 114, pp. 24–31, Apr. 2016.

[42] A. Parmar, R. Katariya, and V. Patel, "A review on random forest: An ensemble classifier," in *Int. Conf. on Intell. Data Commun. Technol., Internet of Things*, J. Hemanth, X. Fernando, P. Lafata, and Z. Baig, Eds., vol. 26. Springer International Publishing, Dec. 2019, pp. 758–763.

[43] P. Gislason, J. Benediktsson, and J. Sveinsson, "Random forest classification of multisource remote sensing and geographic data," in *IEEE Geosci. Remote Sens. (GRS)*, vol. 2, Anchorage, AK, USA, Sep. 2004, pp. 1049–1052.

[44] T.-T. Wong and P.-Y. Yeh, "Reliable accuracy estimates from k-fold cross validation," *IEEE Trans. on Knowl. and Data Eng. (KDE)*, vol. 32, pp. 1586–1594, Apr. 2019.

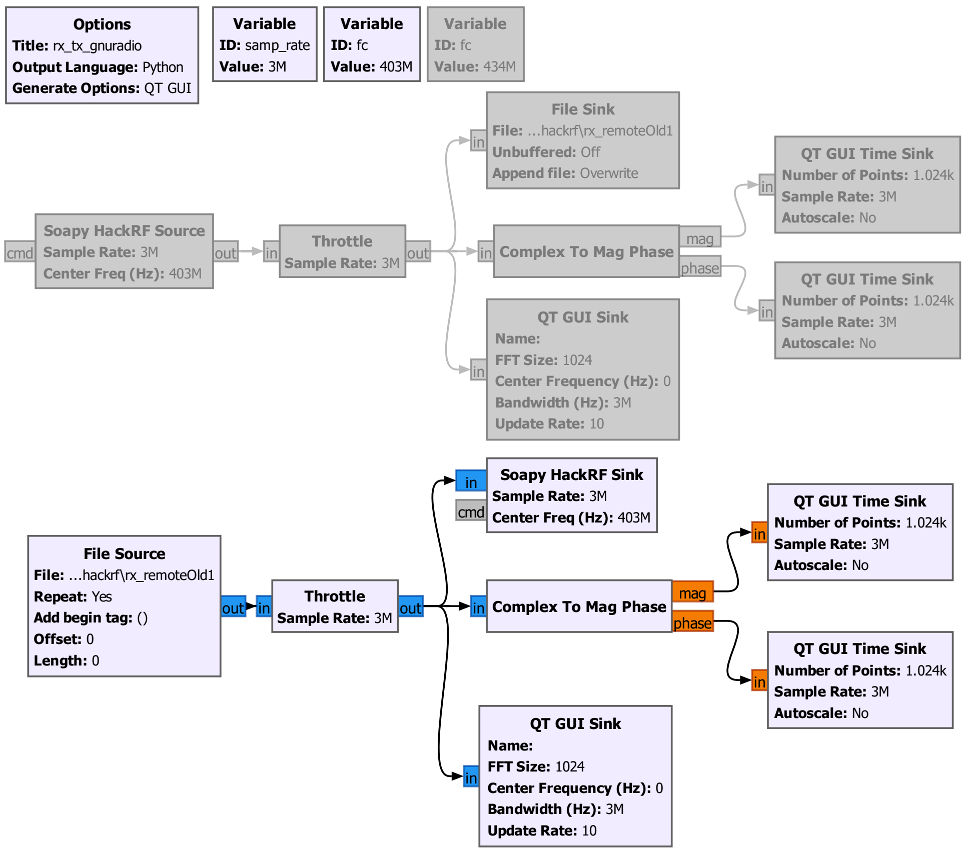
[45] J. D. Rodriguez, A. Perez, and J. A. Lozano, "Sensitivity analysis of k-fold cross validation in prediction error estimation," *IEEE Trans. Pattern Anal. Mach. Intell. (PAMI)*, vol. 32, pp. 569–575, Dec. 2009.

[46] S. Yadav and S. Shukla, "Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification," in *IEEE Int. Conf. Adv. Comput.*, Bhimavaram, India, Aug. 2016, pp. 78–83.

ADDENDUM A HARDWARE AND REPLAY ATTACK CONFIGURATION

This appendix provides visual representations of the GNU Radio configurations used for various replay attack scenarios. In Figures A.1 to A.3, the greyed-out blocks represent the reception configuration, while the highlighted blocks represent the transmission configuration.

Figure A.4 showcases the physical hardware configuration of the SEI system used in this research. It includes the gate access remotes as well as the replay attack SDRs used for transmitters, the four receiver SDRs connected to the laptop as well as the radio-frequency (RF) splitters, connections, and layout employed during the experiments.



Options
Title: rx_tx_gnuradio
Output Language: Python
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 3M

Variable
ID: fc
Value: 403M

Variable
ID: fc
Value: 434M

Soapy HackRF Source
Sample Rate: 3M
Center Freq (Hz): 403M

Throttle
Sample Rate: 3M

File Sink
File: ...hackrf\rx_remoteOld1
Unbuffered: Off
Append file: Overwrite

Complex To Mag Phase

QT GUI Sink
Name:
FFT Size: 1024
Center Frequency (Hz): 0
Bandwidth (Hz): 3M
Update Rate: 10

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

File Source
File: ...hackrf\rx_remoteOld1
Repeat: Yes
Add begin tag: ()
Offset: 0
Length: 0

Throttle
Sample Rate: 3M

Soapy HackRF Sink
Sample Rate: 3M
Center Freq (Hz): 403M

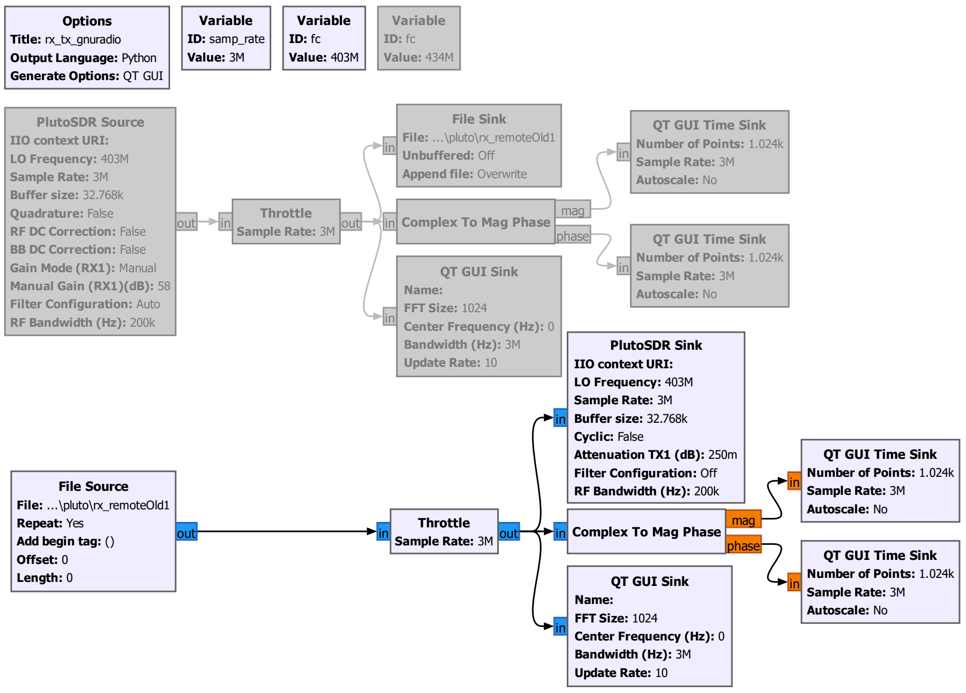
Complex To Mag Phase

QT GUI Sink
Name:
FFT Size: 1024
Center Frequency (Hz): 0
Bandwidth (Hz): 3M
Update Rate: 10

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

Figure A.1. Experimental configuration of the HackRF One replay attack SDR in GNU radio.



Options
Title: rx_tx_gnuradio
Output Language: Python
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 3M

Variable
ID: fc
Value: 403M

Variable
ID: fc
Value: 434M

PlutoSDR Source
IIO context URI:
LO Frequency: 403M
Sample Rate: 3M
Buffer size: 32.768k
Quadrature: False
RF DC Correction: False
BB DC Correction: False
Gain Mode (RX1): Manual
Manual Gain (RX1)(dB): 58
Filter Configuration: Auto
RF Bandwidth (Hz): 200k

Throttle
Sample Rate: 3M

File Sink
File: ...\pluto\rx_remoteOld1
Unbuffered: Off
Append file: Overwrite

Complex To Mag Phase

QT GUI Sink
Name:
FFT Size: 1024
Center Frequency (Hz): 0
Bandwidth (Hz): 3M
Update Rate: 10

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

PlutoSDR Sink
IIO context URI:
LO Frequency: 403M
Sample Rate: 3M
Buffer size: 32.768k
Cyclic: False
Attenuation TX1 (dB): 250m
Filter Configuration: Off
RF Bandwidth (Hz): 200k

File Source
File: ...\pluto\rx_remoteOld1
Repeat: Yes
Add begin tag: ()
Offset: 0
Length: 0

Throttle
Sample Rate: 3M

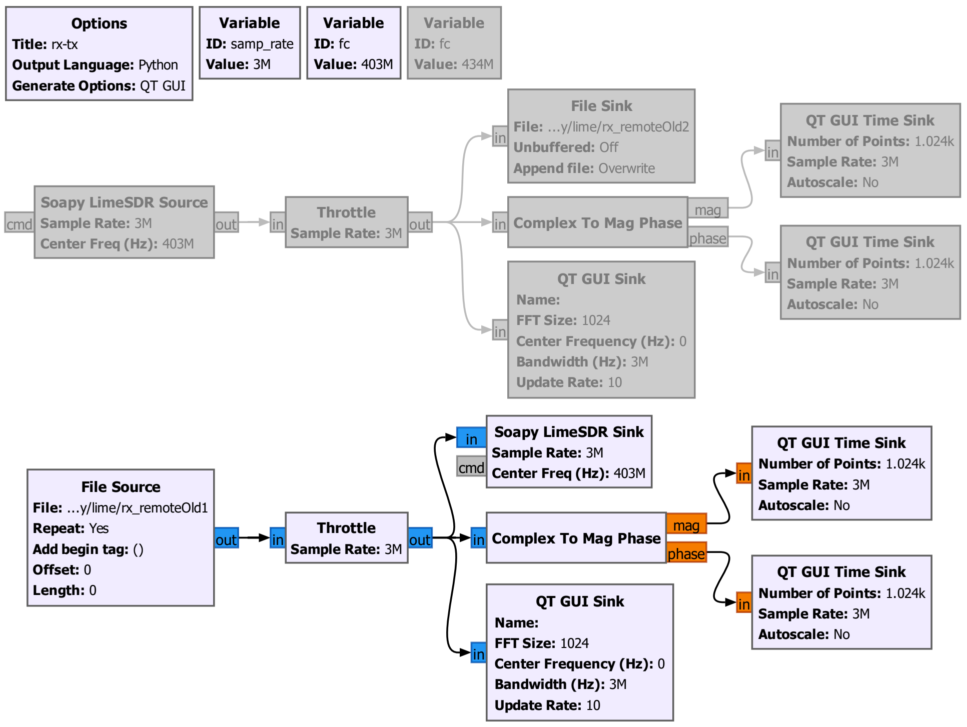
Complex To Mag Phase

QT GUI Sink
Name:
FFT Size: 1024
Center Frequency (Hz): 0
Bandwidth (Hz): 3M
Update Rate: 10

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

Figure A.2. Experimental configuration of the Pluto replay attack SDR in GNU radio.



Options
Title: rx-tx
Output Language: Python
Generate Options: QT GUI

Variable
ID: samp_rate
Value: 3M

Variable
ID: fc
Value: 403M

Variable
ID: fc
Value: 434M

File Sink
File: ...y/lime/rx_remoteOld2
Unbuffered: Off
Append file: Overwrite

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

Soapy LimeSDR Source
Sample Rate: 3M
Center Freq (Hz): 403M

Throttle
Sample Rate: 3M

Complex To Mag Phase

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

QT GUI Sink
Name:
FFT Size: 1024
Center Frequency (Hz): 0
Bandwidth (Hz): 3M
Update Rate: 10

Soapy LimeSDR Sink
Sample Rate: 3M
Center Freq (Hz): 403M

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

File Source
File: ...y/lime/rx_remoteOld1
Repeat: Yes
Add begin tag: ()
Offset: 0
Length: 0

Throttle
Sample Rate: 3M

Complex To Mag Phase

QT GUI Time Sink
Number of Points: 1.024k
Sample Rate: 3M
Autoscale: No

QT GUI Sink
Name:
FFT Size: 1024
Center Frequency (Hz): 0
Bandwidth (Hz): 3M
Update Rate: 10

Figure A.3. Experimental configuration of the Lime replay attack SDR in GNU radio.

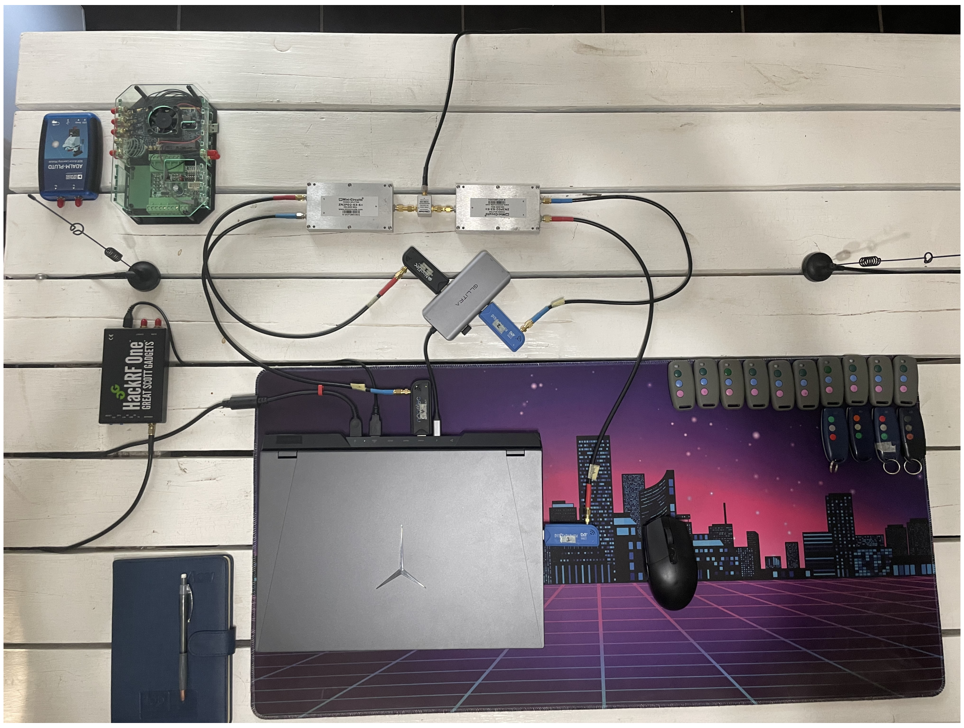


Figure A.4. Experimental configuration of the SEI system.

ADDENDUM B ADDITIONAL RESULTS

The tables in this chapter are the results of the experiments performed in this research. These tables were used to create the summary tables in Chapter 4.

The number of training and testing bursts used to train and test the classifier when one receiver is considered are as follows. The number of training bursts for remotes 1 to 13 is 1170 and the testing bursts are 130. The number of training and testing bursts of remotes 14 to 16 are 2340 and 260 respectively. Table B.1 shows the confusion matrix result of the classifier trained with the first nine recordings of receiver 2, and tested with recording 10 of receiver 2. The average accuracy of the classifier trained with receiver 2 and tested in a 90-10 train test split is 95.4%. Table B.2 shows the confusion matrix result of receiver 3, with an average accuracy of 95.3%. Finally, table B.3 shows the confusion matrix result of receiver 4, with an average accuracy of 95.5%. The average accuracy of the classifier trained with one receiver and tested with the same receiver in a 90-10 train test split is 95.4%.

Table B.4 shows the confusion matrix result of the classifier trained with the first nine recordings of receiver 1, 2, and receiver 3, and tested with recording 10 of all three of the receivers. The resulting number of training bursts for remotes 1 to 13 is 3510 and the number of testing bursts is 390. The training and testing bursts for remotes 14 to 16 are 7020 and 780 respectively. The average accuracy of the classifier trained with three receivers and tested with three receivers in a 90-10 train test split is 91.9%.

Table B.5 shows the confusion matrix result of the classifier trained with the first nine recordings of receiver 1, 2, 3, and receiver 4, and tested with recording 10 of all four of the receivers. The resulting number of training bursts for remotes 1 to 13 is 4680 and the number of testing bursts is 520. The training and testing bursts for remotes 14 to 16 are 9360 and 1040 respectively. The average accuracy of the classifier trained with four receivers and tested with four receivers in a 90-10 train test split is 91.2%.

When all the data of three receivers are considered for training and the other receiver is used for testing, the training and testing bursts for each transmitter are as follows. The number of training bursts for remotes 1 to 13 is 3900 and the number of testing bursts is 1300. The training and testing bursts for remotes 14 to 16 are 7800 and 2600 respectively. The receiver used for testing is different between tests but the number of bursts used is consistent. Table B.6 shows the confusion matrix result of the classifier trained with all the data of receivers 1, 3, and 4, and tested with all the data of receiver 2. The average accuracy of the classifier trained with all the data of receivers 1, 3, and 4, and tested with all the data of receiver 2 is 26.3%.

Table B.7 shows the confusion matrix result of the classifier trained with all the data of receivers 1, 2, and 4, and tested with all the data of receiver 3. The average accuracy of the classifier trained with all the data of receivers 1, 2, and 4, and tested with all the data of receiver 3 is 32.8%.

Table B.8 shows the confusion matrix result of the classifier trained with all the data of receivers 1, 2, and 3, and tested with all the data of receiver 4. The average accuracy of the classifier trained with all the data of receivers 1, 2, and 3, and tested with all the data of receiver 4 is 25.6%.

The results shown in this appendix are similar to the results in chapater 4 where the accuracy of the system is consistent when the classifier is trained with the data of all the considered receivers. When a receiver is not used for training the accuracy of the system is considerably lower. This is due to the fact that the classifier is not able to learn the characteristics of the receiver and therefore the classifier is not able to classify the transmitters correctly.

Table B.1. Confusion matrix where the classifier was trained with the first nine recordings of receiver 2, and tested with recording 10 receiver 2.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U*	Correct
True transmitter	1	128	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	98.5
	2	0	123	0	0	0	0	0	0	0	0	0	0	0	0	0	0	7	94.6
	3	0	0	125	0	0	0	0	0	0	0	0	0	0	0	0	0	5	96.2
	4	0	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	12	90.8
	5	0	0	0	0	110	0	0	0	0	0	0	0	0	0	0	0	20	84.6
	6	0	0	0	0	0	110	0	0	0	0	0	0	0	0	0	0	20	84.6
	7	0	0	0	0	0	0	128	0	0	0	0	0	0	0	0	0	2	98.5
	8	0	0	0	0	0	0	0	117	0	0	0	0	0	0	0	0	13	90.0
	9	0	0	0	0	0	0	0	0	129	0	0	0	0	0	0	0	1	99.2
	10	0	0	0	0	0	0	0	0	0	127	0	0	0	0	0	0	3	97.7
	11	0	0	0	0	0	0	0	0	0	0	126	0	0	0	0	0	4	96.9
	12	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	100
	14‡	0	0	0	0	0	0	0	0	0	0	0	0	0	260	0	0	0	100
	15‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	260	0	0	100
	16‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	253	7	97.3

* Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table B.2. Confusion matrix where the classifier was trained with the first nine recordings of receiver 3, and tested with recording 10 receiver 3.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U*	Correct
True transmitter	1	125	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	5	96.2
	2	0	122	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8	93.8
	3	0	0	127	0	0	0	0	0	0	0	0	0	0	0	0	0	3	97.7
	4	0	0	0	111	0	0	0	0	0	0	0	0	0	0	0	0	19	85.4
	5	0	0	0	0	119	0	0	0	0	0	0	0	0	0	0	0	11	91.5
	6	0	0	0	0	0	118	0	0	0	0	0	0	0	0	0	0	12	90.8
	7	0	0	0	0	0	0	126	0	0	0	0	0	0	0	0	0	4	96.9
	8	0	0	0	0	0	0	0	116	0	0	0	0	0	0	0	0	14	89.2
	9	0	0	0	0	0	0	0	0	119	0	0	0	0	0	0	0	11	91.5
	10	0	0	0	0	0	0	0	0	0	128	0	0	0	0	0	0	2	98.5
	11	0	0	0	0	0	0	0	0	0	0	129	0	0	0	0	0	1	99.2
	12	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	100
	14‡	0	0	0	0	0	0	0	0	0	0	0	0	0	260	0	0	0	100
	15‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	258	0	2	99.2
	16‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	258	2	99.2

* Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table B.3. Confusion matrix where the classifier was trained with the first nine recordings of receiver , and tested with recording 10 receiver 4.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14[†]	15[†]	16[†]	U*	Correct
True transmitter	1	128	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	98.5
	2	0	117	0	0	0	0	0	0	0	0	0	0	0	0	0	0	13	90.0
	3	0	0	125	0	0	0	0	0	0	0	0	0	0	0	0	0	5	96.2
	4	0	0	0	118	0	0	0	0	0	0	0	0	0	0	0	0	12	90.8
	5	0	0	0	0	124	0	0	0	0	0	0	0	0	0	0	0	6	95.4
	6	0	0	0	0	0	115	0	0	0	0	0	0	0	0	0	0	15	88.5
	7	0	0	0	0	0	0	116	0	0	0	0	0	0	0	0	0	14	89.2
	8	0	0	0	0	0	0	0	118	0	0	0	0	0	0	0	0	12	90.8
	9	0	0	0	0	0	0	0	0	125	0	0	0	0	0	0	0	5	96.2
	10	0	0	0	0	0	0	0	0	0	127	0	0	0	0	0	0	3	97.7
	11	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	0	0	100
	12	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	130	0	0	0	0	100
	14[‡]	0	0	0	0	0	0	0	0	0	0	0	0	0	260	0	0	0	100
	15[‡]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	260	0	0	100
	16[‡]	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	257	3	98.8

* Unknown (confidence level below 60%).

[†] Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table B.4. Confusion matrix where the classifier was trained with the first nine recordings of receiver 1, 2, and 3, and tested with recording 10 of all three receivers.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U⋆	Correct
True transmitter	1	368	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	22	94.4
	2	0	329	0	0	0	0	0	0	0	0	0	0	0	0	0	0	61	84.4
	3	0	0	365	0	0	0	0	0	0	0	0	0	0	0	0	0	25	93.6
	4	0	0	0	350	0	0	0	0	0	0	0	0	0	0	0	0	40	89.7
	5	0	0	0	0	322	0	0	0	0	0	0	0	0	0	0	0	68	82.6
	6	0	0	0	0	0	262	0	0	0	0	0	0	0	0	0	0	128	67.2
	7	0	0	0	0	0	0	364	0	0	0	0	0	0	0	0	0	26	93.3
	8	0	0	0	0	0	0	0	352	0	0	0	0	0	0	0	0	38	90.3
	9	0	0	0	0	0	0	0	0	347	0	0	0	0	0	0	0	43	89.0
	10	0	0	0	0	0	0	0	0	0	381	0	0	0	0	0	0	9	97.7
	11	0	0	0	0	0	0	0	0	0	0	381	0	0	0	0	0	9	97.7
	12	0	0	0	0	0	0	0	0	0	0	0	390	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	390	0	0	0	0	100
	14‡	0	0	0	0	0	0	0	0	0	0	0	0	0	780	0	0	0	100
	15‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	776	0	4	99.5
	16‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	760	20	97.4

⋆ Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table B.5. Confusion matrix where the classifier was trained with the first nine recordings of receiver 1, 2, 3, and 4, and tested with recording 10 of all four receivers.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U*	Correct
True transmitter	1	496	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	24	95.4
	2	0	438	0	0	0	0	0	0	0	0	0	0	0	0	0	0	82	84.2
	3	0	0	479	0	0	0	0	0	0	0	0	0	0	0	0	0	41	92.1
	4	0	0	0	452	0	0	0	0	0	0	0	0	0	0	0	0	68	86.9
	5	0	0	0	0	416	0	0	0	0	0	0	0	0	0	0	0	104	80.0
	6	0	0	0	0	0	335	0	0	0	0	0	0	0	0	0	0	185	64.4
	7	0	0	0	0	0	0	477	0	0	0	0	0	0	0	0	0	43	91.7
	8	0	0	0	0	0	0	0	451	0	0	0	0	0	0	0	0	69	86.7
	9	0	0	0	0	0	0	0	0	470	0	0	0	0	0	0	0	50	90.4
	10	0	0	0	0	0	0	0	0	0	508	0	0	0	0	0	0	12	97.7
	11	0	0	0	0	0	0	0	0	0	0	511	0	0	0	0	0	9	98.3
	12	0	0	0	0	0	0	0	0	0	0	0	520	0	0	0	0	0	100
	13	0	0	0	0	0	0	0	0	0	0	0	0	520	0	0	0	0	100
	14‡	0	0	0	0	0	0	0	0	0	0	0	0	0	1040	0	0	0	100
	15‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1037	0	3	99.7
	16‡	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1022	18	98.3

* Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table B.6. Confusion matrix where the classifier was trained with all the data of receivers 1, 3, and 4, and tested with all the data of receiver 2.

		Predicted transmitter																	
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U*	Correct
True transmitter	1	831	3	1	6	2	23	140	4	3	37	0	0	0	0	148	93	9	63.9
	2	5	21	124	0	268	293	1	72	151	320	0	0	0	0	0	27	18	1.6
	3	0	0	578	4	34	423	0	123	43	60	0	0	0	0	0	30	5	44.5
	4	6	6	169	46	31	538	1	99	47	282	0	0	0	2	0	63	10	3.5
	5	172	8	223	2	201	107	26	58	46	405	0	0	0	0	0	38	14	15.5
	6	91	5	122	15	218	446	11	141	49	136	0	0	1	0	0	28	37	34.3
	7	90	0	204	84	151	211	45	31	4	291	0	0	0	0	0	183	6	3.5
	8	13	0	340	7	189	187	7	356	68	16	0	0	0	0	0	97	20	27.4
	9	8	4	388	10	145	249	2	41	341	86	0	0	0	0	0	17	9	26.2
	10	72	8	436	1	154	19	0	25	10	452	0	0	0	0	6	114	3	34.8
	11	0	0	0	203	0	0	0	0	0	0	566	97	5	50	247	132	0	43.5
	12	0	0	0	214	0	0	0	0	0	0	110	444	0	44	0	488	0	34.2
	13	0	0	0	0	0	0	0	0	0	0	0	0	1144	154	0	2	0	88
	14‡	0	0	0	1	0	0	0	0	0	0	715	482	0	1376	25	1	0	52.9
	15‡	184	0	7	0	2	0	0	0	7	0	1085	72	0	0	469	774	0	18.0
	16‡	393	0	1	65	0	4	0	0	0	0	73	560	160	0	823	518	3	19.9

* Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table B.7. Confusion matrix where the classifier was trained with all the data of receivers 1, 2, and 4, and tested with all the data of receiver 3.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14[†]	15[†]	16[†]	U*	Correct
True transmitter	1	0	20	7	416	378	339	10	121	2	1	0	0	0	0	0	0	6	0
	2	5	31	0	569	88	5	14	5	469	52	0	0	0	0	0	42	20	2.4
	3	82	6	124	28	2	254	0	84	30	130	0	0	0	0	52	500	9	9.5
	4	167	3	0	669	56	18	0	3	47	15	1	6	2	9	140	146	18	51.5
	5	7	214	24	34	337	57	7	272	274	15	0	0	0	1	1	31	26	25.9
	6	135	40	28	21	349	12	14	203	14	39	0	0	1	8	194	176	66	0.9
	7	230	19	0	32	175	2	299	149	3	26	0	0	0	0	85	273	7	23
	8	41	11	48	2	107	8	45	342	62	48	0	0	0	0	198	361	27	26.3
	9	60	156	126	3	163	3	2	73	108	142	0	0	0	0	150	300	14	8.3
	10	0	308	82	78	33	43	0	157	166	425	0	0	0	0	0	0	8	32.7
	11	0	0	0	0	0	0	0	0	0	0	245	972	0	14	0	69	0	18.8
	12	0	0	0	0	0	0	0	0	0	0	339	3	0	47	911	0	0	0.2
	13	0	0	0	0	0	0	0	0	0	0	0	0	1298	1	0	0	1	99.8
	14[‡]	0	0	0	0	0	0	0	0	0	0	0	137	0	2463	0	0	0	94.7
	15[‡]	0	5	0	417	158	663	0	44	13	0	0	1269	0	9	3	19	0	0.1
	16[‡]	0	18	94	312	13	727	0	78	51	0	309	0	0	24	0	967	7	37.2

* Unknown (confidence level below 60%).

[†] Remotes 14 to 16 are replay attacks of Remotes 1 and 11.

Table B.8. Confusion matrix where the classifier was trained with all the data of receivers 1, 2, and 3, and tested with all the data of receiver 4.

		Predicted transmitter																
		1	2	3	4	5	6	7	8	9	10	11	12	13	14†	15†	16†	U*	Correct
True transmitter	1	356	0	276	0	0	0	0	0	0	0	0	0	0	0	3	653	12	27.4
	2	48	468	606	0	0	0	0	0	0	0	0	0	0	0	146	0	32	36
	3	132	28	1115	0	0	0	0	0	0	2	0	0	0	0	8	0	15	85.8
	4	403	322	146	0	0	0	0	0	0	0	0	0	9	19	379	0	22	0
	5	105	0	1019	0	0	0	0	0	0	5	0	1	0	0	109	2	59	0
	6	302	0	722	0	0	0	0	0	0	11	0	0	10	5	114	29	107	0
	7	147	1	641	0	0	0	0	0	0	0	0	0	0	0	496	2	13	0
	8	154	0	994	0	0	0	0	0	0	1	0	0	0	0	109	0	42	0
	9	222	0	953	0	0	0	0	0	0	66	0	0	0	0	16	11	32	0
	10	39	0	980	0	0	0	0	0	0	193	0	0	0	0	48	35	5	14.8
	11	0	0	0	0	0	0	0	0	0	0	0	0	0	1300	0	0	0	0
	12	0	0	0	0	0	0	0	0	0	0	0	902	0	398	0	0	0	69.4
	13	0	0	0	0	0	0	0	0	0	0	0	44	111	1033	0	111	1	8.5
	14‡	0	0	0	0	0	0	0	0	0	0	0	62	0	2538	0	0	0	97.6
	15‡	6	0	697	0	0	0	0	0	0	0	0	0	0	1300	555	42	0	21.3
	16‡	4	0	782	0	0	0	0	0	0	1	0	0	1	665	85	1057	5	40.6

* Unknown (confidence level below 60%).

† Remotes 14 to 16 are replay attacks of Remotes 1 and 11.