

Modulation Classification Using a Deep Neural Network to Optimize Future Radio Communications

Final Report

University of Sussex

School of Engineering and Informatics

Candidate Number: 181395

BEng Electrical and Electronics Engineering with Robotics

Supervisor: Prof. Maziar Nekovee

Submission Date: 18/05/21

Word Count: 10826

Summary

The purpose of the project was to research how issues surrounding future radio communications can be solved using modulation classification techniques. Successful classification requires artificial intelligence applications under noisy conditions, therefore a convolutional neural network (CNN) was trained to identify signals of various modulation types that are used in communication systems today.

A pre-trained CNN designed by MATLAB was used to identify modulation signals and the accuracy of this was analyzed by plotting classification scores. After it was clear that the pre-trained CNN had low classification accuracies, the pre-trained CNN was trained using transfer learning techniques. Frames were generated for training, validating, and testing the CNN and a classification accuracy of 92% was achieved.

The trained network was again analyzed, this time using over-the-air signals rather than synthetic data. This was made possible using two separate Analog Devices ADALM-Pluto SDRs to transmit and receive live wireless RF signals. The distance between the SDRs was changed and varieties of obstacles were placed to cover an SDR. It was determined that the trained neural network performance did not suffer from any change in transmission distance up to 2.5 m and the different materials had very low impacts on network performance.

Statement of Originality

This report is submitted as part of the requirement for the degree of BEng Electrical and Electronic Engineering with Robotics at the University of Sussex. I declare that my work contains no examples of misconduct, such as plagiarism, collusion, or fabrication of results. I have referenced all sources of information.

Software and hardware used in this project were the industry standard MATLAB and the ADALM-Pluto SDR respectively. Much of the code and experiments used in this project were broadly based on the MathWorks Modulation Classification with Deep Learning example. However, my own code and experiments were adapted to suit the requirements of this project. I certify that except where indicated in the text, all other parts are my own original work.

A handwritten signature in black ink, reading "H.R. Bayford". The signature is written in a cursive style with a horizontal line underneath the name.

Acknowledgements

For her continued support and encouragement throughout this project and my entire degree, I would like to thank my wonderful partner Jada, without whom I would not have been able to achieve my dreams.

Contents

| | | |
|-----------|-------------------------------------------------------------------------------------------------------|-----------|
| 1 | Introduction..... | 3 |
| 1.1 | Objectives | 3 |
| 2 | Background | 4 |
| 2.1 | Future Telecommunications Technology..... | 4 |
| 2.2 | Solution to Spectrum Limits | 6 |
| 2.3 | Modulation | 6 |
| 2.4 | Modulation Classification..... | 8 |
| 2.5 | Deep Learning..... | 9 |
| 3 | Hardware and Software Research..... | 10 |
| 3.1 | Analysis of Software Platforms | 10 |
| 3.2 | Analysis of Hardware Options | 11 |
| 3.3 | ADALM-Pluto SDR | 13 |
| 4 | Neural Network Design and Training | 15 |
| 4.1 | Pre-trained Neural Network Analysis | 15 |
| 4.2 | Frame Generation and Feature Extraction for Training | 18 |
| 4.3 | Network Design | 20 |
| 4.4 | Classification Accuracy Evaluation | 21 |
| 5 | Over-the-air Testing | 22 |
| 5.1 | Test Results | 22 |
| 5.2 | Distance vs Accuracy | 23 |
| 5.3 | Obstacle Tests | 26 |
| 6 | Project Plan | 31 |
| 7 | Ethical Review | 32 |
| 8 | Health and Safety | 32 |
| 9 | Conclusions and Recommendations | 35 |
| 10 | References | 36 |
| 11 | Appendices | i |
| | Appendix A: Simulink Model for Simulated 64QAM Modulated Signals..... | i |
| | Appendix B: MATLAB Script for Untrained CNN Predictions for Relevant Digital Modulation Types..... | ii |
| | Appendix C: MATLAB Script for Frame Generation and Feature Extraction..... | v |
| | Appendix D: MATLAB Script for Storing Data MAT Files into Computer Memory | viii |

| | |
|------------------------------------------------------------------------------------------------------------------|-------|
| Appendix E: MATLAB Script for Designing and Training CNN | ix |
| Appendix F: MATLAB Function Script for Testing Over-the-air Signals | xi |
| Appendix G: MATLAB Script Testing Radio Connection and Calling the Testing Function | xiv |
| Appendix H: Full Table of Distance vs Accuracy Results | xv |
| Appendix I: MATLAB Script for Plotting Graph of Test Accuracy Against Distance Between Radios | xvi |
| Appendix J: MATLAB Script for Plotting Graph of Test Accuracy Against Distance for Each Modulation Type | xvii |
| Appendix K: Confusion Matrices generated from obstacle experiments | xviii |
| Appendix L: Full Table of Material vs Accuracy Results | xx |

1 Introduction

The need for future radio technology like 5G is ever-growing with the increasing volumes of data transmitted and received daily as well as the demand for faster speeds and more reliable connection from consumers. With this increase in demand, there are concerns that there will not be enough capacity in the radio frequency spectrum to sustain the large traffic volumes.

Different techniques are being developed to solve this issue including multi-input multi-output (MIMO) techniques, more efficient traffic management, radio resource allocation and “small cells”. However, a promising solution to this issue is spectrum sharing, a bandwidth shared between multiple network operators that offer various services. The issue with this is the signal interference associated with multiple sources sharing the same bandwidth, particularly in diverse applications such as military-based operations and call requests from mobile phones. One way to solve this issue is waveform modulation classification, the identification of different signals. However, there is not a high degree of accuracy associated with this technique, but deep learning applications could improve this accuracy. This project is based on improving this accuracy by implementing deep neural networks, so 5G communications can be improved in spectrum sharing scenarios.

This project's purpose was to explore the design and implementation of a deep neural network for the classification of modulated signals. The project will cover how a convolutional neural network (CNN) was designed and trained to identify signals through simulated waveform data using the numeric computing platform MATLAB and how the trained network was tested with over-the-air signals using ADALM-Pluto software defined radios (SDRs) as well as various experiments carried out to observe the effects of changing external factors on the classification accuracy of the trained CNN.

1.1 Objectives

The project objectives were:

- To carry out an extensive literature review of the background information of the project.
- To analyze and choose the most suitable software and hardware for this project.
- To observe and examine the behaviour of a pre-trained CNN.
- To perform feature extraction for different modulation types by generating synthetic, simulated waveforms.
- To design and train a CNN using synthetic data.
- To identify different waveform modulations using the newly trained CNN.
- To test the performance of the CNN using Software Defined Radios.
- To observe effects on performance from external factors such as different distances and obstacles.
- To evaluate how future techniques could further increase performance accuracy.

2 Background

2.1 Future Telecommunications Technologies

Developments in communications are most often from advancements in cellular or mobile technology with the latest being 5G. 5G is the 5th generation of mobile technology, following on from 1G, 2G, 3G and 4G networks. Each new generation has built on the predecessor with improved technologies to boost user experience and 5G is no different. While speeds will vary between location and network provider, they are much higher than that of 4G, with 5G being currently around 10 times faster than 4G at average speeds of 150 – 200 Mbps and peak speeds of over 1 Gbps, compared to average speeds of 23 – 35 Mbps and a max speed of 150 Mbps for 4G [1]. However, with the development and expansion of 5G, speeds are expected to reach up to 100 times faster than 4G [2]. 5G will also have lower latency and response times, so networks will take less time to respond to a request, ultimately improving speeds for devices. Currently, there are latencies of 21 – 26ms but in theory, 5G could reach latencies as low as 1ms. This is compared to the latency of 4G which is at 35 – 50ms [3].

By analyzing the numbers, we can see that the newest generation will be faster than previous generations which is extremely useful for streaming, internet browsing and gaming, but the greatest features of 5G are in the emergence of new and improved technologies that come with it such as self-driving cars, augmented/virtual reality, holograms, drones, and remote working through robotics. The increase in capacity of 5G networks could lead to the development of IoT devices including advanced smart homes and cities.

To achieve the high speeds that power these new technologies, 5G devices will operate at higher frequency bands in both the sub six range (600 MHz to 6 GHz), part of which is used by current 4G devices, as well as a higher band from 24 GHz to 52 GHz [4] where wavelengths are around 1mm as seen in figure 1. These frequency bands have been previously utilized by space and military sectors but have never been used by mobile devices before so there is a higher bandwidth for mobile devices leading to the faster data speeds and lower latencies previously discussed. However, there are problems when operating at these higher frequency bands as the wavelengths are very short and the waveforms cannot travel as far as those with longer wavelengths and smaller frequencies. Millimetre waves can only travel shorter distances and they struggle to penetrate obstacles like buildings while also getting absorbed by plants and rain. Various new technologies are being implemented to allow 5G to operate using millimetre waves.

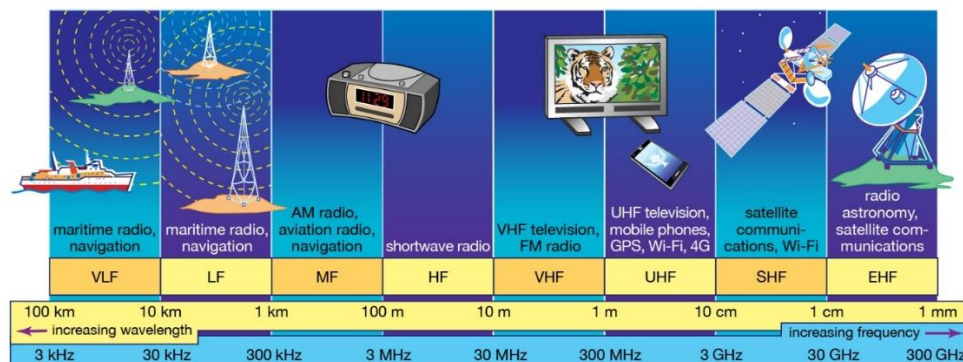


Figure 1: Radio Frequency Spectrum [5]

The first technology is small cells. They are small, low powered cellular nodes that act like cellular base or substations, transmitting and receiving signals to and from devices. Since they are low powered, they cannot transmit signals as far as base stations, but they are very small so can be placed in close proximity to one another and in locations more convenient to users receiving signals, including outside buildings with “Microcells” or even inside buildings with “Picocells” and “Femtocells”. This allows devices to use signals with high frequencies and high data rates in busy environments like city centres with many obstacles, by switching to a small cell that is not obstructed.



Figure 2: Samsung Indoor Link Cell [6]

Another technology being implemented with 5G is massive MIMO (Multi Input Multi Output) which will increase the capacity for devices using a particular station. Current stations used for 4G have 12 antennas that handle all traffic, but massive MIMO stations would be able to support around 100 antennas. However, if data is broadcasted in all directions as they currently are, there will be lots of interference from the crossing signals. Another piece of technology called beamforming is being implemented to counter this issue. This will allow stations to send focused streams of data to specific devices instead of broadcasting data in every direction at once. The location of each device is derived from algorithms taking the timing and direction of the original signal into account.

These technologies are being implemented to tackle the issues that come with operating in the 24 – 52 GHz bandwidth. The high-frequency millimetre waves lead to higher data rates, however, another reason that 5G is being introduced in this region is that lower bandwidths in the spectrum are getting too crowded. Due to the increasing volume of data from more devices going online, there will be slower services and more dropped connections from so much traffic in specific bandwidths. Many bandwidths in the spectrum are used for existing data traffic so it is extremely difficult to clear old bands for new uses like 5G devices, particularly due to legal reasons as bandwidths are often bought by Multi-Network Operators (MNOs).

Operators are executing 5G technology by opening up new bandwidths at higher frequencies for cellular devices, but eventually, we will run into the same issue with bandwidth becoming crowded. Furthermore, the technology needed to carry out 5G in these bandwidths is very expensive and will only benefit users in urban areas for the foreseeable future while those in rural areas will be stuck with 4G data rates. An alternative approach that avoids these issues is spectrum sharing, the enabling of network access to additional frequency bands when other services are not using them.

2.2 Solution to Spectrum Limits

Spectrum sharing has a lot of potential for solving the issue of the capacity of 5G communication, however, it is a relatively new concept and is still being developed. The issue is that different spectrum users cannot operate effectively and without interference. The National Institute of Standards and Technology (NIST) has conducted research into deep learning techniques that can detect when radars are operating. This could lead to improved spectrum sharing capabilities, particularly in the case of sharing the 3.5 GHz band in the USA. In 2015, this band was allocated to be shared between the Citizens Broadband Radio Service (CBRS) and other commercial users when not in use by the CBRS [7]. Deep learning algorithms improve the accuracy of radar detection and waveform modulation identification, allowing users to identify when the band is being used.

2.3 Modulation

Modulation is the process of converting data into radio waves by adding information to a carrier signal. It is used in many fields of communication and is vital for transmitting and receiving information. Most often, modulation is used to allow the use of smaller antennas since the size of an antenna is proportional to the length of a waveform. If a waveform with a low frequency, therefore a high wavelength was transmitted from a radio tower, the size of the antenna required would be extremely high, perhaps in the range of kilometres. Signals need to be modulated to higher frequencies to ensure this is not the case.

The signal with the information is superimposed with a carrier signal, a waveform with constant amplitude, frequency, and phase. Either of these basic waveform properties of the carrier signal is varied in accordance with the signal with information. For example, frequency modulation (FM) is when the frequency of the carrier signal is varied according to the amplitude of the information signal. The value of the original signal can be obtained by demodulating the modulated signal through observation of the frequency. Amplitude modulation (AM) follows the same principle in that the carrier waves amplitude is varied according to the amplitude of the information signal. These two modulation types are analogue, first used in radio applications for transmitting and receiving analogue data like voice and later on used in the 1st generation of mobile networks (1G). However, analogue modulation is very susceptible to noise from the environment, which is why most modulation types are now digital.

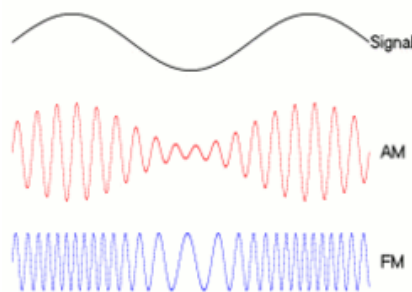


Figure 3: AM vs FM signal modulation [8]

Digital modulation is used to transmit and receive digital (binary) signals rather than analogue signals. The amplitude, frequency or phase of the carrier signal is varied according to the binary data (bits) in the information signal. Various modulation techniques have been developed, attempting to send more and bits in a single waveform. The digital modulation techniques that are used today and are covered in this project are analyzed in table 1.

| Modulation Type | Process | Data Rate | Applications | Constellation Diagram |
|--------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------|
| BPSK – Binary Phase Shift Keying | Two phase offsets, one for logic high, one for logic low. Phase offsets are 180° Two points on constellation | 1 bit per symbol | <ul style="list-style-type: none"> - Robust as binary 1 and 0 separated by 180° of phase. Low Bit Error Rate (BER). Difficult for demodulator to read wrong data entry, even with noise or distortions - Used by most cellular towers for long distance communication - Power efficient - Wireless LANs - Bluetooth | |
| QPSK – Quadrature Phase Shift Keying | Four phase offsets (45°, 135°, 225°, 315°) for four possible two-bit numbers (00, 01, 10, 11) Four points on constellation | 2 bits per symbol | <ul style="list-style-type: none"> - Bandwidth twice as efficient than BPSK - Still resistant to noise and distortion as bits are separated by 90° (still a low BER) - Used for long to medium distance communication - 3G/4G Over very long distances or in noisy conditions | |
| 8PSK – 8 Phase Shift Keying | Eight phase offsets separated by 45° to represent 000, 001, 010, 011, 100, 101, 110, 111. eight points on constellation | 3 bits per symbol | <ul style="list-style-type: none"> - Television and video broadcast - Aircraft and satellite systems - Military operations - 3G/4G LTE Over long to medium distances (rural areas) - Noise tolerance is still low but higher than BPSK and QPSK as bits on constellation diagram are closer | |
| 16QAM – 16 Quadrature Amplitude Modulation | 4 bits are represented by 16 possible waveforms with different phase and amplitude combinations. 16 points on constellation | 4 bits per symbol | <ul style="list-style-type: none"> - Television and video broadcast - Aircraft and satellite systems - 4G/5G in medium to short distances - Used in cities with short proximities to cell towers - BER much higher than PSK modulation, less noise tolerance - Signal to Noise Ratio (SNR) must be high so data is demodulated correctly - More power required for a higher data rate | |
| 64QAM – 64 Quadrature Amplitude Modulation | 6 bits represented by 64 possible waveforms with different amplitude and phase combinations. 64 points on constellation | 6 bits per symbol | <ul style="list-style-type: none"> - Television and video broadcast - Aircraft, satellite, and space technology - 4G/5G in very short distances - Used in city centers, with very short distances to the nearest cell towers - BER even higher than 16QAM, even less noise tolerance as bits are closer together in constellation diagram - Signal often interrupted when behind obstacles or inside buildings - SNR must be high, more power consumed for a higher data rate | |
| PAM4 – Pulse Amplitude Modulation | 4 different amplitude levels in waveform represent four possible two bit numbers (00, 01, 10, 11) 4 points on constellation | 2 bits per symbol | <ul style="list-style-type: none"> - Ethernet, LANs - Digital television - Electronic drivers for LED lighting - Graphics cards - Low BER, due to high noise tolerance as points on constellation are far apart | |

Table 1: Analysis of Digital Modulation Types

Table 1 reveals the different features and applications for each type of digital modulation. A pattern can be seen where the effective communication distance decreases as data rates increase. This is due to the noise tolerance being low for low data rate modulation types while being high for high data rate types. The reason for the decrease in noise tolerance can be seen in the constellation diagrams. [9]

Constellation diagrams are representations of digitally modulated signals on an I/Q diagram (In-phase and Quadrature axis). These axis act as a polar coordinate system with real and complex planes and the signals acting as vectors. Each point is a constellation, representing a symbol (a possible waveform). The angle of the constellation points, measured clockwise from the horizontal axis, represents the phase shift of the signal while the distance from the origin represents the amplitude of the signal. For example, the BPSK constellation diagram has just two constellation points representing two possible symbols, high or low (1 or 0). These points are the same distance from the origin but are on opposite sides, they are 180 degrees apart. This means that the amplitude is constant, and data is represented by changes in phase, which is exactly how BPSK modulation operates, shown in figure 4.

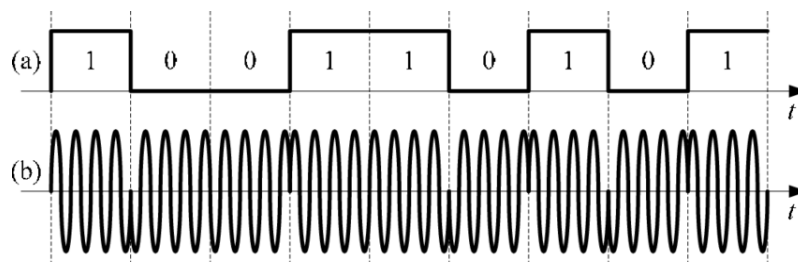


Figure 4: Data signal against BPSK modulated signal [10]

The patterns of constellations for each digital modulation type reveal why the noise tolerance decreases as the data rate increases. The closer the constellations are, the easier it is to mistake one point for another, meaning the symbol measured when we demodulate the signal is wrong. The reasons that the symbols are misread is because of changes in amplitude or phase due to noise and disturbance. The closer the points are, the less noise is needed to cause a misreading. Higher data rate modulation types are used in short proximity to cell towers as there is more noise over long distances, as seen in figure 5.

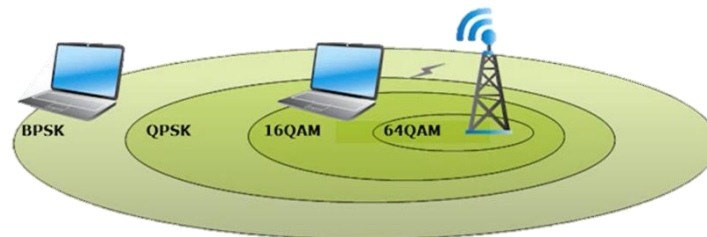


Figure 5: Diagram of modulation types used based on distance from cell tower [11]

2.4 Modulation Classification

Constellation diagrams are helpful tools as they can help diagnose issues with transmitting and receiving signals. For example, we know there is phase error if the constellation points are all rotated relative to their desired positions. Along with waveform feature extraction, they are also tools to help identify and classify the modulation type when receiving a signal from an unknown source. However, it becomes extremely hard to manually identify a modulated signal under significant amounts of noise and interference in a crowded spectrum, as illustrated in figure 6.

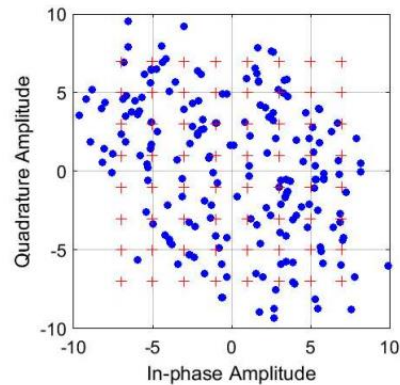


Figure 6: 64QAM Constellation diagram showing transmitted signal (red) and received signal (blue)

Figure 6 shows the constellation diagram from the Simulink model in appendix A. It illustrates how difficult it is to classify a modulated signal under large amounts of noise where the received signal is significantly different from the transmitted signal. For this reason, researchers are using deep learning to identify these signals for us by training neural networks by feeding them thousands of modulated waveforms for classification.

2.5 Deep Learning

Deep learning is a subset of machine learning in artificial intelligence that focuses on models called artificial neural networks, inspired by the function and structure of the human brain. Models are fed large amounts of synthetic data in the form of text, images, or sound, and are trained to learn its features to perform classification tasks with potentially vastly higher accuracy than humans. Based on the data the model is given, it decides for itself what features are relevant to classify an object, whereas machine learning models classify objects based on features that we say are relevant. This allows deep learning models to improve accuracy as more data is provided.

A deep neural network gets its name due to the many processing layers, as illustrated in figure 7. Labelled synthetic data is fed into the input layer which is transferred to the hidden layers via channels that are each assigned a numerical value, known as weight. Each layer consists of neurons (also called nodes) that are assigned a value or bias which is passed through an activation function that determines whether that particular function is activated or not. Through weighted channels, the activated neurons transmit data to the neurons of the next hidden layer. The output layer consists of probabilities and the neuron with the largest value determines the prediction of the neural network.

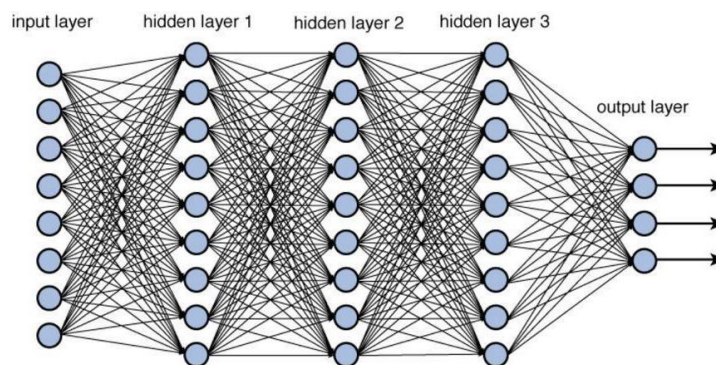


Figure 7: Deep neural network with multiple layers [12]

The type of neural network used in this project is a convolutional neural network (CNN). CNNs are usually involved in image classification tasks where the input layer is a two-dimensional array of neurons representing pixels from an image. The network includes many feature detection layers that perform one of three operations: convolution, pooling, or rectified linear unit (ReLU). Convolution layers activate certain features from the images through filters while pooling simplifies the output by performing nonlinear downsampling, reducing the number of parameters that the network needs to learn about. Rectified linear unit (ReLU) allows for faster and more effective training by mapping negative values to zero and maintaining positive values. [13] Classification layers output the prediction of the image.

Deep learning techniques are great tools as they have unlimited accuracy and features are automatically learned rather than manually assigned in machine learning. However, huge amounts of data are required to train a model, often thousands or millions of images for significant accuracy. This makes deep learning computationally expensive. One way to counter this issue is with transfer learning, a technique that adapts an existing deep neural network for a new application. A pre-trained model performs new classification tasks by modifying the convolution layers and repeating training and assessment until desired results are obtained. Transfer learning requires less synthetic data, less computational power and is much less time consuming than building a new neural network from scratch.

3 Hardware and Software Research

3.1 Analysis of Software Platforms

The first task in this project was to decide the choice of software to use when designing the neural network for our task of modulation classification. After researching all possible platforms, two, in particular, stood out: MATLAB and Python. Python would be a good choice as it is highly efficient and productive in its code while having simple and easy to read syntax. This is why Python is so popular with computer programmers. Furthermore, it integrates very well with other software systems like TensorFlow and PyTorch for example, which are used for machine learning applications and could be used for this project.

MATLAB is the other possible platform and it is very popular among engineers, opposed to Python. This is due to its dedication to mathematical computing allowing easier matrix and vector computations, and engineering graphics or models are easier to generate. Also, Simulink is available through MATLAB, a block diagram environment for model-based design. Simulink is more interactive with users as it is graphical and allows rapid construction of virtual prototypes.

Multiple add-ons and toolboxes are available through MATLAB, particularly the deep learning and communications toolboxes are significantly helpful for this project. Several online resources and examples are also provided including the example “Modulation Classification with Deep Learning” which is extremely helpful and is partially tailored towards this project. Lastly, MATLAB seamlessly integrates with many pieces of hardware through support packages. To test the neural network designed in this project, a software defined radio (SDR) is needed to transmit and receive signals over-the-air. Support packages for various SDRs are provided through MATLAB.

Due to the benefits that come with it, MATLAB was chosen to be the platform to design and train a neural network. Following this, an appropriate SDR was chosen by evaluating those with a support package provided by MATLAB. A support package needed to be provided as it includes additional commands and functions specifically made for use with the SDR, allowing easier integration and communication.

3.2 Analysis of Hardware Options

To test the neural network accuracy, an SDR was needed. SDRs are devices that transmit and/or receive wireless signals with various frequencies. They are used by engineers to perform activities such as field testing live RF signals or rapid prototyping radio functions. They are also used in education to get hands-on learning of radio communication concepts. They usually consist of integrated circuits like programmable System-on-Chips (SoCs) or Field-Programmable-Gate-Arrays (FPGAs) to enable programming. Analog to digital converters (ADCs) and digital to analogue converters (DACs) are used to convert the signals. A lot of the hardware seen in traditional radios like mixers, amplifiers, filters, modulators, demodulators etc. are not seen in software defined radios as most of these processes are done in software or firmware. MATLAB provides support packages for four different SDRs. These were the Xilinx Zynq based radios, the E310, RTL-SDR and the ADALM-Pluto. [14] The choice of SDR depended on a variety of factors.



Figure 8: Xilinx Zynq-7000 SoC ZC706 Evaluation Kit [15]

The Xilinx Zynq based SDRs are those with the programmable Zynq System-on-Chips made by Xilinx. For example, the ADALM-Pluto SDRs among many others use the Xilinx Zynq-7000 SoC. However, Xilinx also provides kits to manually construct an SDR using many individual components of theirs, opposed to single component kits like the other SDRs that MATLAB support. This SDR once constructed, can transmit signals of 6GHz frequency, therefore being excellent for 5G simulations. However, this kit is suitably very expensive at \$2495, unfortunately far above the budget for this project. [15]



Figure 9: USRP E310 Hardware [16]

Similarly, the USRP E310 is an excellent SDR that has a large frequency range at 70 MHz to 6 GHz and would be an excellent choice for the simulation of 5G signals. It provides up to 56 MHz of instantaneous bandwidth and a sample transfer rate of up to 10 mega samples per second (MSPS). In addition, it has many other abilities such as burst modes for rapid signal transmission and it allows for custom algorithms on FPGA fabric and ARM processors using an HDL coder or Embedded coder through Xilinx Vivado

Design Suite. [17] However, these additional functions are not applicable or helpful to this project and much like the Xilinx Zynq kit, this SDR is far above the budget provided by the university at £3160. [16]



Figure 10: RTL-SDR Hardware [18]

On the other hand, a popular and much cheaper option is the RTL-SDR. This is a dongle containing the Realtek RTL2832U chip, which provides I-Q samples through the USB interface. It has a configurable centre frequency in the range of 22 MHz to 2.2 GHz and a configurable sampling rate of 225 – 300 kHz and 900 – 2560 kHz. The SDR is configured using a raspberry pi computer, enabling the process of real-time wireless signals such as FM radio, aeroplane surveillance signals (ADS-B) and signals from smart meters [18] when used alongside MATLAB and Simulink. Although, a disadvantage with this SDR is that wireless signals can only be received and not transmitted which is useful for some but not all applications in this project. Perhaps, for this reason, the RTL-SDR is cheaper than most other SDRs at £21 [19]



Figure 11: ADALM-Pluto SDR Hardware [20]

This leaves the ADALM-Pluto SDR. Like the USRP E310, it can be used to transmit live wireless signals through MATLAB and Simulink. It can also receive and process wireless signals for applications like aeroplane surveillance signals and FM radio like the RTL-SDR. The Pluto uses a Xilinx Zync-7000 System-on-Chip, and The AD9363 RF allows the SDR to operate at a configurable centre frequency of 325 MHz to 3.8 GHz. However, the MATLAB support package reconfigures the SDR to 70 MHz to 6GHz, matching the USRP-E310 and making it possible to simulate 5G signals with the high-frequency range. The sampling rate of this SDR is configurable between 520 kHz and 61.44 MHz. [20] The ADALM-Pluto achieves many of the features of the most expensive SDRs like the USRP-E310 but at a much cheaper price, initially found at £142.70 [21] and later found at £127.80 [22].

When considering all factors, the ADALM-Pluto is an obvious choice for the SDR to be used in this project due to its low cost but high performance.

3.3 ADALM-Pluto SDR

The ADALM-Pluto SDR is capable of both transmitting and receiving, however for the experiments in this project, two SDRs were required, one specifically for transmitting and one for receiving. This made it possible to vary the experiments like changing the distance between the transmitting and receiving antenna, for example, to observe how distance affected the accuracy of the neural network. The total cost of the two kits came to £270.50, one at £142.70 and another was found at a later date with a price of £127.80.

The contents of the ADALM-Pluto SDR kit can be seen in figure 12 and are as follows:

- 1 – ADALM-Pluto SDR active learning module
- 2 – RF antennas (4 cm)
- 1 – SMA RF loopback cable (15cm)
- 1 – Micro USB type B to USB type A cable
- 1 – Micro USB connector
- 2 – Module SMA connector protective caps



Figure 12: Contents of ADALM-Pluto SDR Kit

The ADALM-Pluto SDR active learning module is a plastic case containing the SDR circuit board as seen in figure 12. The Module has two USB ports, one for USB connection with a host computer for example and another for power, although the module can be powered using the voltage supplied from the host computer through the USB connection. The typical DC input ranges from 4.5 V to 5.5 V. There is one transmitter channel and one receiver channel which can both operate on half and full-duplex, enabling simultaneous communication. The circuit board found inside the plastic casing can be seen below in figure 13.

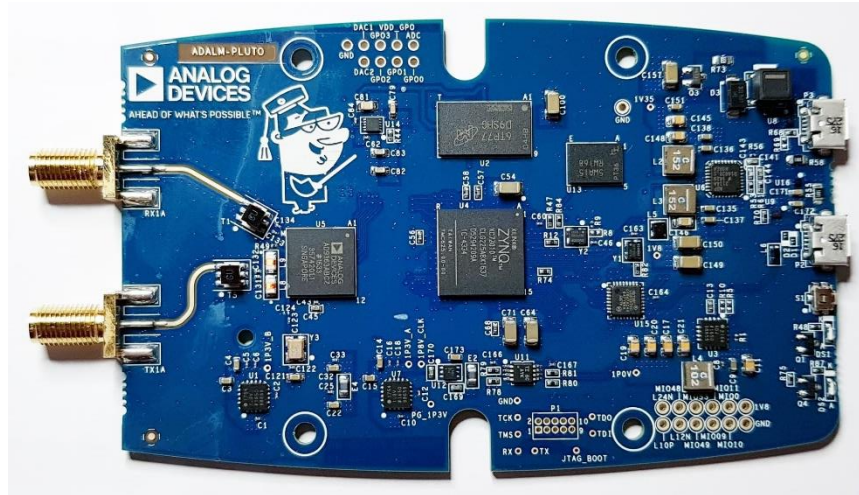


Figure 13: ADALM-Pluto SDR Circuit Board [23]

The main components of the circuit board can be seen in figure 14 below showing a simplified version of the circuit board. As previously discussed, the Xilinx Zynq-7000 all programmable SoC is a programmable logic system and integrated processor that enables communication with the host computer through the USB connection. The Analog Devices AD936x RF Agile Transceiver is an integrated circuit with similarly structured transmitter and receiver subsystems. The transmitter subsystem includes two identical transmit channels containing a 12-bit digital to analogue converter, analogue filters, FIR filters, digital interpolation filters, direct conversion mixers and a power amplifier. The receiver subsystem similarly includes two identical receive channels each containing a 12-bit analogue to digital converter, analogue filters, FIR filters, digital decimation filters, direct conversion mixers and a low noise amplifier. Both the digital-to-analogue and analogue-to-digital converters have configurable sampling rates from 65.2 kilosamples per second to 61.44 Megasamples per second. For memory devices to store information, the circuit board uses the Micron DD3RL for RAM and the Micron QSPI Flash for flash memory. [23]

Simplified Block Diagram

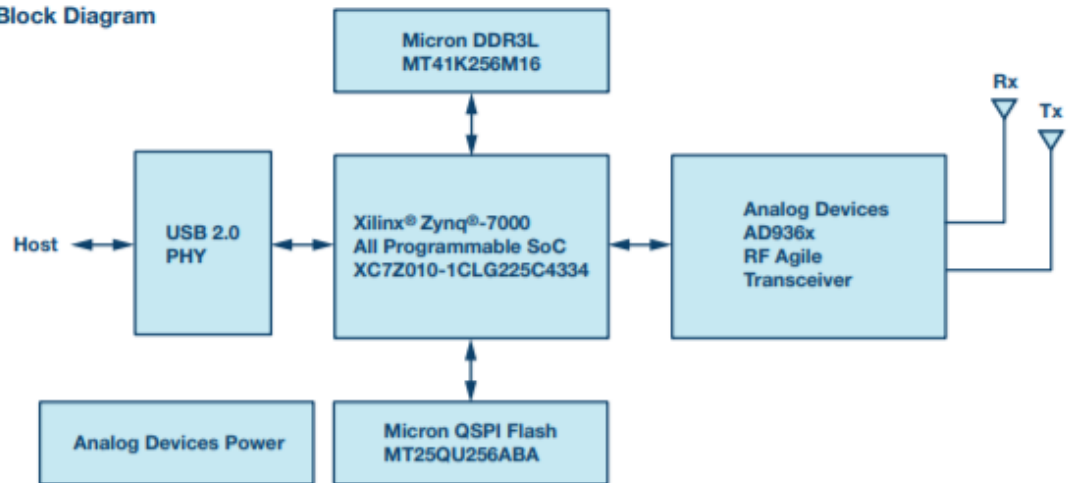


Figure 14: Simplified block diagram of ADALM-Pluto SDR circuit board [24]

4 Neural Network Design and Training

The analysis, design, and implementation of the CNN in this project is inspired by the “Modulation Classification with Deep Learning” example provided by MathWorks. [25]

4.1 Pre-trained Neural Network Analysis

Once sufficient research was put into the software and hardware and a suitable option for each was chosen, a pre-trained CNN designed by MATLAB was selected. This CNN was already able to identify eight different digital modulation types including: Binary Phase Shift Keying (BPSK), Quadrature Phase Shift Keying (QPSK), 8-ary Phase Shift Keying (8PSK), 16-ary Quadrature Amplitude Modulation (16QAM), 64-ary Quadrature Amplitude Modulation (64QAM), 4-ary Pulse Amplitude Modulation (PAM4), Gaussian Frequency Shift Keying (GFSK) and Continuous Phase Frequency Shift Keying (CPFSK). The CNN was also able to recognize analogue modulation of the following three types: Broadcast FM (B-FM), Double Sideband Amplitude Modulation (DSBAM) and Single Sideband Amplitude Modulation (SSB-AM).

For the purpose of this project, the accuracy of the pre-trained CNN was only analyzed for the digital modulation types currently seen in today’s cellular technology. These include BPSK, QPSK, 8PSK, 16QAM, 64QAM and PAM4 modulation types. The MATLAB script seen in appendix B was written to evaluate the performance of the pre-trained CNN.

The accuracy of the pre-trained CNN was evaluated by generating bar charts representing the final value of the neural network output layers, the classification score ranging from 0 to 1. A score of 1 means the model is 100% certain of a prediction whereas a score of 0 means the opposite. The synthetic data fed into the input layers of the pre-trained CNN were random bits between the values of 1 and 0 that were populated into a 1024 x 1 array since the CNN could process 1024 samples at a time. These random bits mimic a signal with information we wish to modulate onto a carrier signal and send to another device. To simulate real life, the random bits were modulated and passed through a square-root raised cosine pulse shaping filter and a test channel with certain features and impairments. The transmitted and received signals are shown in figures 15 to 20 as well as the predictions of each modulation type.

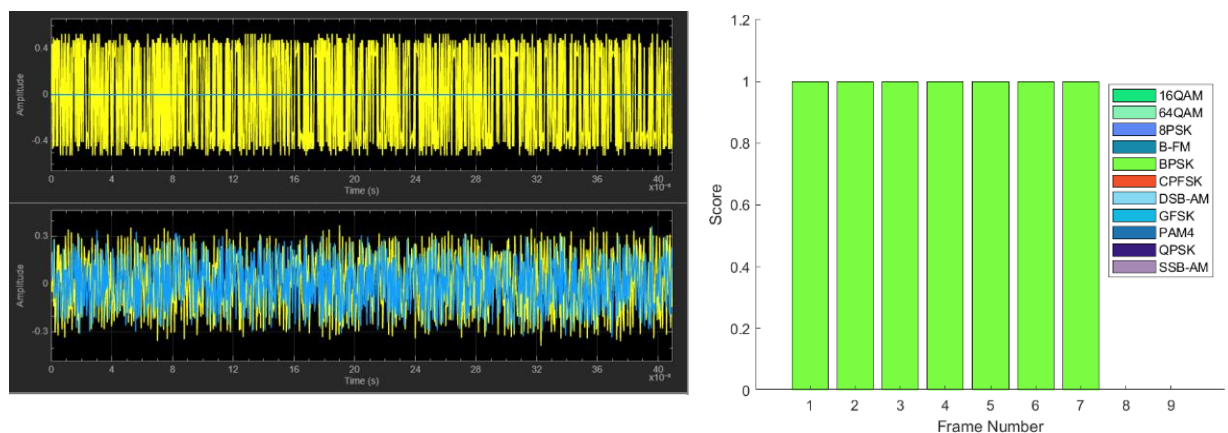


Figure 15: Time scope showing transmitted and received BPSK signals (left), classification scores (right)

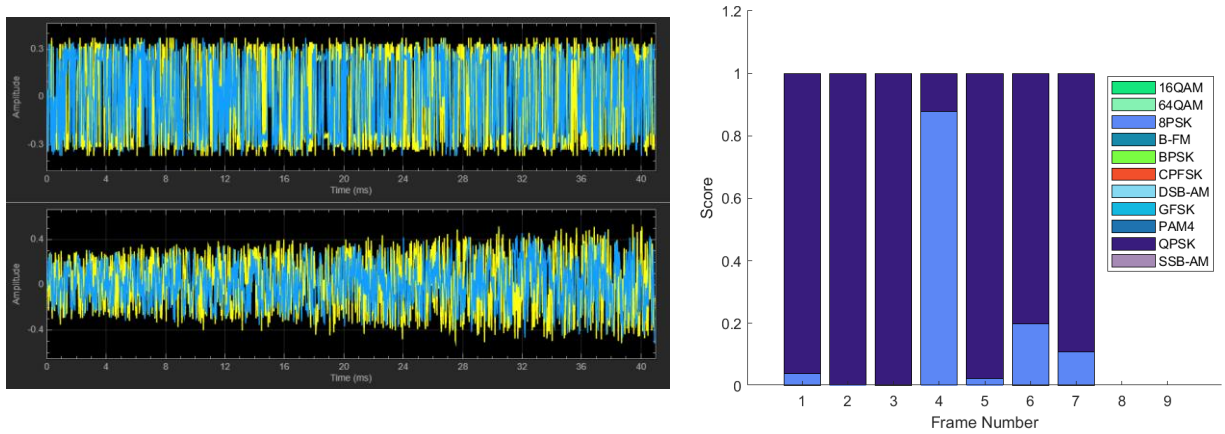


Figure 16: Time scope showing transmitted and received QPSK signals (left), classification scores (right)

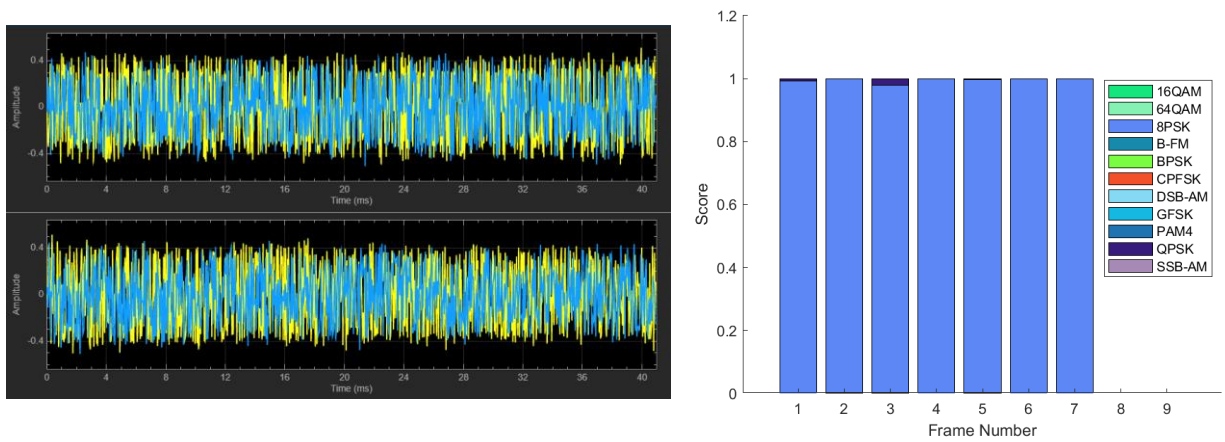


Figure 17: Time scope showing transmitted and received 8PSK signals (left), classification scores (right)

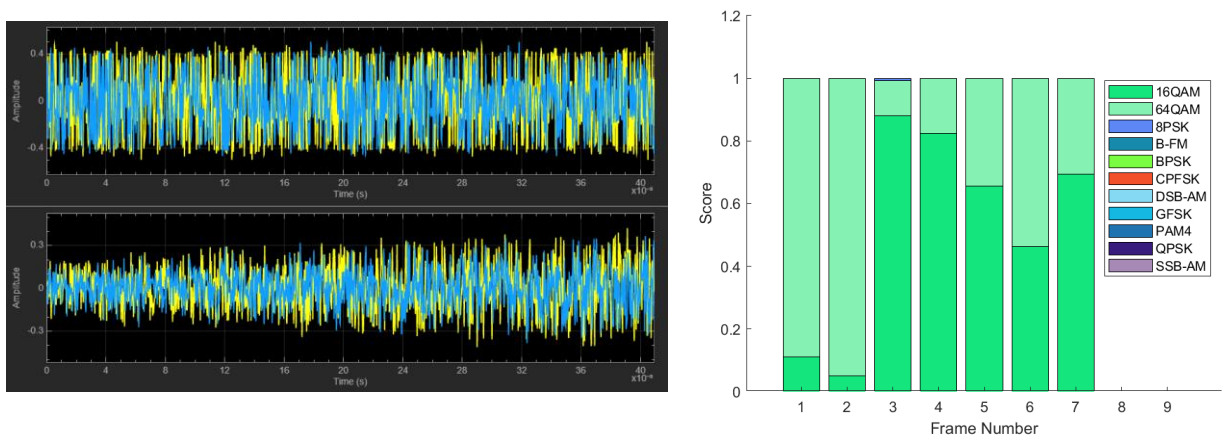


Figure 18: Time scope showing transmitted and received 16QAM signals (left), classification scores (right)

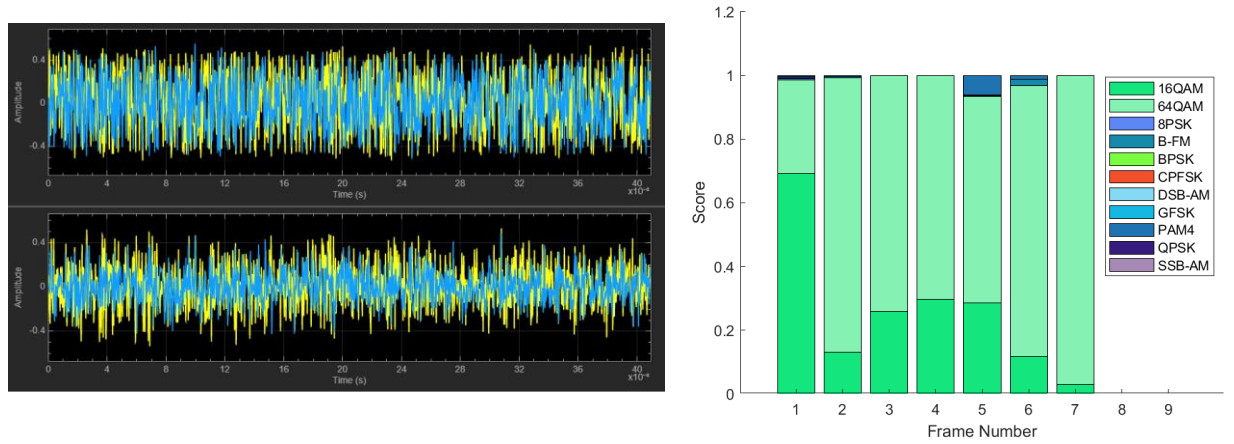


Figure 19: Time scope showing transmitted and received 64QAM signals (left), classification scores (right)

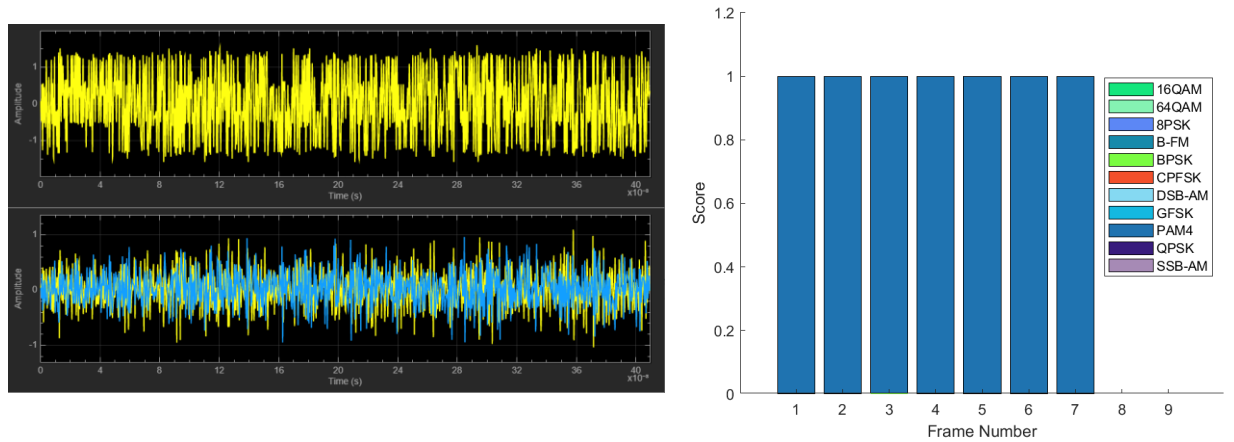


Figure 20: Time scope showing transmitted and received PAM4 signals (left), classification scores (right)

We can observe from figure 15, The model successfully classified the waveform to be BPSK modulated with 100% certainty in all seven of the generated frames. This is also the case for the PAM4 modulated waveform in figure 20 as the model accurately predicted all seven frames with 100% certainty. This was due to the simple structure in the modulation scheme, previously discussed in table 1, in that it takes large amounts of noise or interference to cause errors in classification. There was again a high degree of accuracy in the predictions for the 8PSK modulated signal in figure 16 since there was only incredibly minor confusion between 8PSK and QPSK in three of the seven frames. However, the opposite was the case in the QPSK modulated waveform predictions in figure 17 where frame four of the seven frames was inaccurately predicted to be 8PSK modulated and with a lot of certainty at 88%. This could be caused by the similarities in their modulation scheme as both are determined by phase offsets. Similarly, there was confusion between the 16QAM and 64QAM modulated waveform predictions, perhaps also due to the similarities in their modulation schemes where both phase and amplitude determines symbols. Frames 1, 2 and 6 in the 16QAM predictions inaccurately predicted the waveform to be 64QAM modulated whereas frame 1 in the 64QAM predictions was inaccurately predicted to be 16QAM modulated. There was likely to be less accurate in 16QAM and 64QAM predictions due to the low noise tolerance and the proximity of symbols in the constellation diagrams.

4.2 Frame Generation and Feature Extraction for Training

Since the pre-trained CNN had been evaluated, the process of transfer learning the CNN to improve its accuracy was started. The first step was to generate frames for each modulation types, used for training, validation, and testing. 10,000 frames were generated with 80% of them being used for training, 10% for validation, and 10% for testing. The training frames are used as synthetic data for the network to learn from, while the validation frames tell the network how well it performed classifying the training frames. The final classification accuracy is obtained using the test frames with never seen before samples. Each frame consisted of 1024 samples as this is the number of samples the network can take in the input layer, and the sample rate for each frame was 200 kHz. The centre frequencies for the received and transmitted signals were 902 MHz for digital types and 100 MHz for analogue types. The MATLAB script for this exercise can be found in appendix C.

To improve the performance of the CNN, the frames were sent through an impaired channel so the neural network can identify and classify modulated signals even under noise or distortion. The channel impairments were Added White Gaussian Noise (AWGN) with a signal to noise ratio (SNR) of 30dB, Rician multipath fading with a K-factor of 4 and a maximum Doppler shift of 4 Hz, and a clock offset causing a centre frequency offset and sampling time drift.

Channel impaired frames were generated and stored in a MAT-file along with the associated label of the frame. By saving the data directly into files, we eliminate the need to generate the data every time this script is run. The data can also be shared more effectively when sending the file.

More effective and common methods for training CNNs are to extract the features of the image and this is no different for this CNN. Time-frequency transforms showing the time-domain representation of each modulation type have been generated, showing the CNN the defining features of waveforms over time and improving the training since almost all signals are non-stationary and change over time. The generated time-domain waveforms are plotted in figure 21.

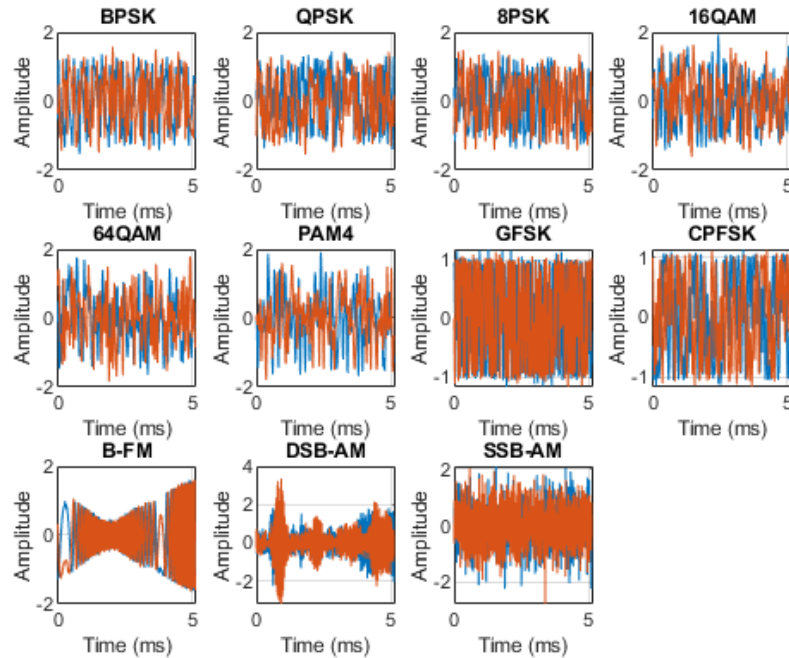


Figure 21: Time-domain representations for each modulation type

Like the transmitted and received waveforms in figures 15 to 20, the time-domain representation of the waveforms in figure 21 shows two colours in the waveform. This is because the signals are complex and have real and imaginary parts.

Another popular technique for training the CNN to classify modulation signals is spectrogram generation. Spectrograms display changes in the frequencies in a signal over time and the amplitude is then represented on a third dimension with variable brightness or colour. The generated spectrograms can be seen below in figure 22.

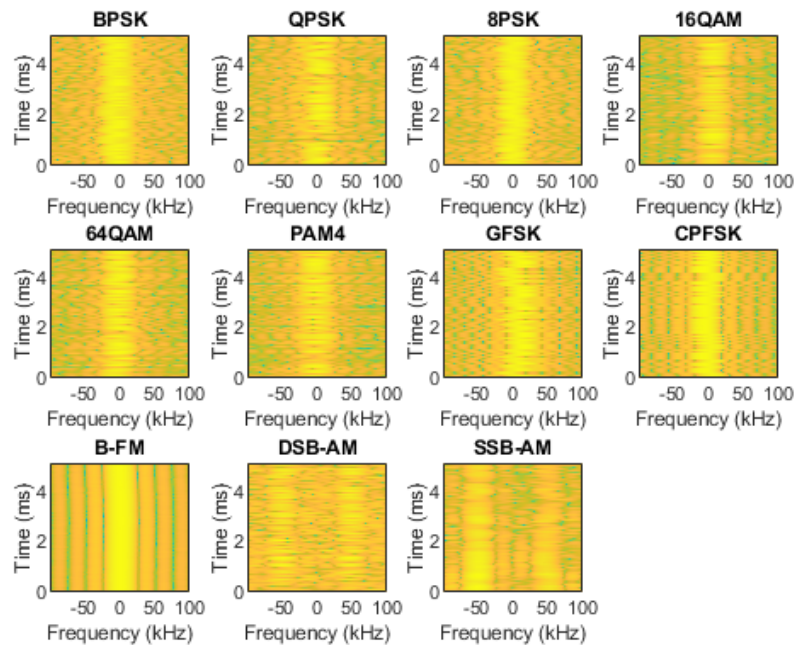


Figure 22: Spectrogram plots for each modulation type

The CNN expects real inputs, opposed to complex baseband samples at the receiver, so the complex signals were transformed into real 4-D arrays to be fed to the input of the CNN. The frames generated for the two types of plots seen in figures 21 and 22 were placed into datastores so they could be imported into the computer memory.

Since neural network training is iterative, the datastore reads data from the MAT files at every iteration and transforms the data from complex to real before updating the network coefficients. However, the repetition of reading from the file and transforming from complex to real data can be eliminated by importing the data from the MAT files into the computer memory. The data is instead read from the files and transformed just once which cuts down the training time from approximately 110 minutes to 50 minutes.

The MATLAB script used for frame generation can be found in appendix C while the script used for importing data into the computer memory can be found in Appendix D.

4.3 Network Design

The neural network that was trained consisted of an input layer, six hidden layers and one fully connected layer. Each of the hidden layers except the last, consist of a 2D convolution layer followed by a rectified linear unit (ReLU) layer, an activation layer, and a max-pooling layer. The last hidden layer replaces the max-pooling layer with an average pooling layer. The final output layer in the network is a softmax activation layer.

Each convolution layer acted as a filter and activated certain features from the waveforms in the images, depending on the modulation type. The ReLU layers allowed for faster and more effective training by mapping negative values to zero and maintaining positive values while the pooling layers simplified the output by performing nonlinear downsampling, reducing the number of parameters that the network needs to learn about. These layers when combined, create the many hidden layers in this network. At the output stage, the fully connected layer output a vector of K dimensions where K is the number of classes that the network will be able to predict and the vector contained the probabilities for each classification. This was fed into the softmax layer which contains a softmax function to provide the classifications. This data was subsequently fed into the classification layer, outputting those classifications with corresponding labels.



Figure 23: Analysis of designed neural network

The network was trained using a variety of training techniques that are commonly used in industry. By using the Stochastic Gradient Descent with Momentum (SGDM) solver, convergence speed was improved and the minibatch size was set to 256 as this suited the size of the CPU while the max epochs were set to 12 as a higher number provides no further training advantage but slows the training speed down. A lower number would improve training speed but hinder the final accuracy.

After implementing these techniques, the trained network had a test accuracy of 92%. The MATLAB script used to train the CNN is found in appendix E.

4.4 Classification Accuracy Evaluation

After training, a confusion matrix was plotted to display the classification accuracy of the CNN, showcased in figure 24.

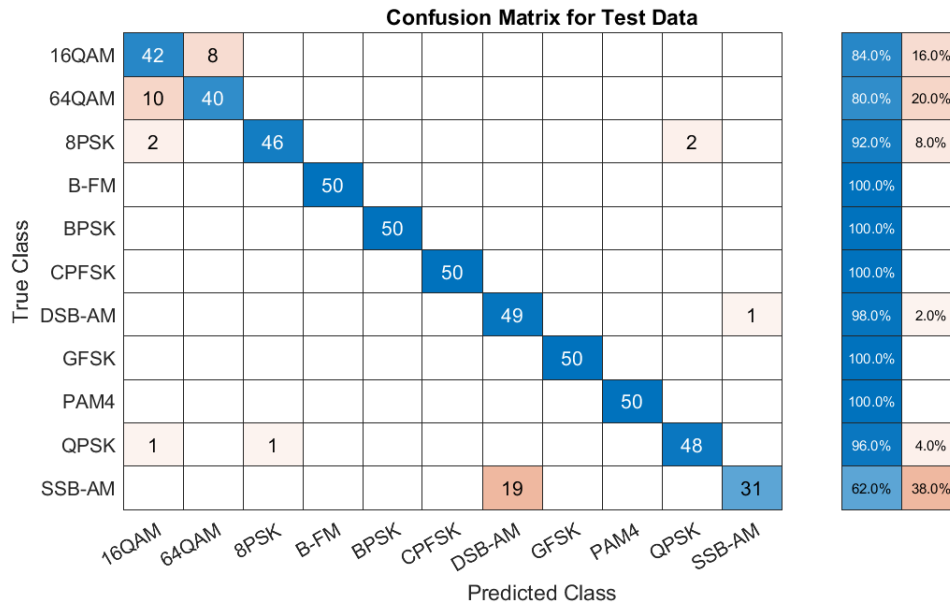


Figure 24: Confusion matrix from synthetic test frames

A confusion matrix is a table displaying the number of correct or incorrect predictions made by the network. The rows represent the true modulation type of the classified signal whereas the columns represent the prediction made by the CNN. A prediction is correct when the column matches the row and the number within that represents the number of correct predictions. Correct predictions are blue whereas incorrect ones are red and the percentage of correct or incorrect predictions, out of 50, determines the shade of the colour. As displayed in the percentages on the right, most of the modulation types classified have 100% accuracy, represented in a deep blue colour.

Like the first experiment, there is still some confusion between the 16QAM and 64QAM modulation types. The accuracy for 16QAM modulation is 84% and for 64QAM just 80% with the other respective 16% and 20% mistaken for the other quadrature amplitude modulation type. This is again due to the similarity in both waveform structure and the constellation diagrams of both modulation types since 16QAM is a subset of 64QAM. There is also confusion between QPSK and 8PSK which is again similar to the first experiment. This is most likely due to the phase rotation from the multipath fading and frequency offset impaired channel that the test frames were put through that makes both modulation types look similar in their constellation diagrams. Furthermore, some confusion between both analogue modulation types, again due to the similarities in their structure from the amplitude modulation.

5 Over-The-Air Testing

5.1 Test Results

Two ADALM-Pluto SDRs were placed 0.5 m apart from each other and connected to the ports usb:0 and usb:1 respectively, as demonstrated in figure 25.

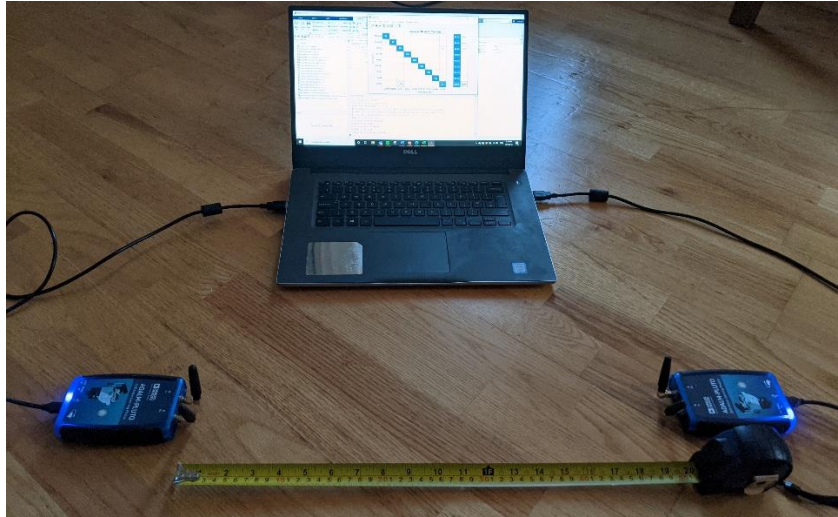


Figure 25: ADALM-Pluto SDRs separated by 0.5 m.

SDRTest, a MATLAB function, was written to transmit and receive over-the-air modulated signals using the two ADALM-Pluto radios. The received signals were generated into test frames for the neural network to classify and the performance of the CNN was evaluated by generating another confusion matrix and overall accuracy was calculated. 100 frames were generated for each modulation type, each with a sample rate of 200 kHz and 1024 samples.

The digitally modulated signals had a centre frequency of 902 MHz while the analogue modulated signals had a centre frequency of 370 MHz. This centre frequency is rather high for analogue signals, however, it is the lowest centre frequency possible with AD9363 RF. The support package available through MATLAB allows the radio to be configured with another firmware, the AD9364 RF, however after repeated efforts, the ADALM-Pluto SDRs could not be reconfigured, and their frequency range was stuck at 370 MHz to 3.8 GHz.

To match the test frames characteristics, the transmitter radio was set to operate with a maximum centre frequency of 902 MHz and both the transmitter and receiver radio were configured with a baseband sample rate of 200 kHz. The receiver radio was set to have an adjustable gain with a minimum of -10 dB and a maximum of 73 dB. The `transmitRepeat()` function from the ADALM-Pluto support package was used to download the waveform signal, data, to the transmitter radio tx, and transmit it repeatedly over the air. The waveform signal was downloaded into the radio hardware memory and transmitted over the air repeatedly, without gaps [26]. The gain of the receiver radio was adjusted before each modulated signal test frames, to avoid signal loss. After each modulated signal was sent and received, another confusion matrix was plotted, detailing the performance of the network, as seen in figure 25. In addition, the overall test accuracies for each modulation type were given, plotted in Table 2. Another MATLAB script was written to test the connection of both radios and if successfully connected, the script calls the SDRTest function and performs the previously mentioned operations.

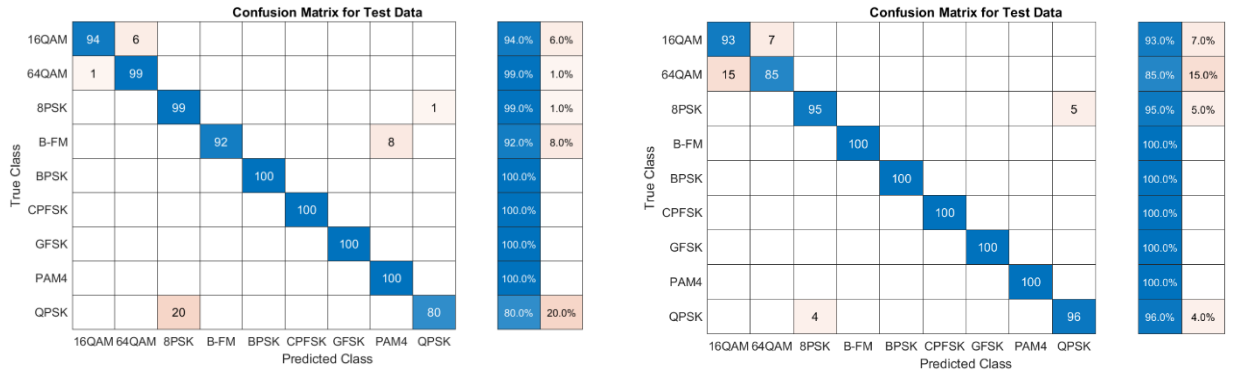


Figure 26: Confusion matrices for over-the-air signals, 0.5 m apart (left), 2.7 m apart (right)

| Modulation Type | Test Accuracy (%) |
|-----------------|-------------------|
| BPSK | 100 |
| QPSK | 98 |
| 8PSK | 97 |
| 16QAM | 96 |
| 64QAM | 93.8 |
| PAM4 | 94.83 |
| GFSK | 95.57 |
| CPFSK | 96.125 |
| B-FM | 96.556 |

Table 2: Modulation type vs accuracy when radios 0.5 m apart

As illustrated in figure 26 and table 2, the network performed better overall for this experiment using live RF signals rather than synthetic signals in the previous experiments, with an overall accuracy of 96.43%. The accuracy of the quadrature amplitude modulated signals improved, however, there was more confusion this time between the QPSK and 8PSK signals. The 16QAM and 64QAM signals may have been classified easier in this example due to there being less noise or interference in the room, compared to the impaired test channel the synthetic signals were sent through. To observe the effect that the distance between the radios had on the accuracy, the script was run again but the radios were at the maximum distance away from each other at 2.7m (the maximum distance was determined by the length of the USB cables). The confusion matrix is shown in figure 26 above and on the right.

The confusion matrix in figure 26 shows the CNN performance with the radios far away from each other and as expected, the 16QAM and 64QAM modulated signals performed worse than before. These signals were expected to perform worse over distance due to their high data rate and low noise tolerance, hence their applications in the real world are specifically used for short-range communication. The overall accuracy this time was 84.7%, significantly lower than the last test with the radios closer together. While both confusion matrices have some differences and the overall accuracy dropped in the second test, there was not enough evidence to suggest that the neural network performance was significantly affected by the change in distance between the two radios.

5.2 Distance vs Accuracy

To test if there is any proportionality between the classification accuracy and the distance between the two radios, more comprehensive testing was required. The MATLAB script for testing the network performance with over-the-air signals was run at varying distances, starting from 0.25 m, and increasing 0.25 m for each test up to 2.5 m. At first, tests were performed just once for each measurement, however, after observing the variation in classification accuracy, it was determined that there should be 3 tests per measurement so that an average can be calculated across the three tests. This led to a lot more time being

spent on testing due to each test taking roughly 5 minutes, but the quality and consistency of the experiments improved. Table 3 shows the average accuracy across 3 repeated tests for each modulation type, measured against the distance between the two radios. The full table with all repeated tests can be seen in appendix H.

| Mod Type | Distance (m) | | | | | | | | | |
|----------------|--------------|--------------|--------------|--------------|--------------|-------------|--------------|--------------|--------------|--------------|
| | 0.25 | 0.5 | 0.75 | 1 | 1.25 | 1.5 | 1.75 | 2 | 2.25 | 2.5 |
| BPSK | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 99.67 | 100.00 | 100.00 | 100.00 |
| QPSK | 95.00 | 92.75 | 92.30 | 87.00 | 90.60 | 91.00 | 91.50 | 91.77 | 93.50 | 93.16 |
| 8PSK | 96.55 | 93.66 | 94.22 | 89.56 | 91.99 | 92.67 | 94.00 | 94.32 | 95.10 | 94.76 |
| 16QAM | 96.75 | 94.58 | 92.83 | 89.58 | 93.08 | 93.42 | 86.83 | 93.60 | 89.50 | 94.30 |
| 64QAM | 95.80 | 92.13 | 92.26 | 89.93 | 91.60 | 92.20 | 86.87 | 93.73 | 90.80 | 92.60 |
| PAM4 | 96.50 | 93.43 | 93.50 | 91.78 | 93.05 | 93.50 | 89.05 | 94.76 | 92.34 | 93.83 |
| GFSK | 97.01 | 94.05 | 94.48 | 92.81 | 94.01 | 94.40 | 90.62 | 95.52 | 93.43 | 94.71 |
| CPFSK | 97.38 | 95.08 | 95.16 | 93.83 | 94.79 | 95.13 | 91.75 | 96.08 | 94.25 | 95.38 |
| B-FM | 92.41 | 93.70 | 94.08 | 94.08 | 94.59 | 95.49 | 91.83 | 96.69 | 94.66 | 95.88 |
| Overall | 96.38 | 94.38 | 94.33 | 91.88 | 93.76 | 94.2 | 91.35 | 95.18 | 93.73 | 94.96 |

Table 3: Distance vs average classification accuracy per modulation type

From the results of these tests, it is clear that the initial hypothesis was incorrect, the performance of the network did not get worse as the distance between the two radios increased. The previous results seemed to show some correlation between accuracy and distance, with the accuracy being significantly lower for the test when the radios were far apart. However, after completing over 30 separate tests, it was noticed that there are many fluctuations in accuracies between tests of the same distance and modulation test. The previously mentioned confusion matrices may represent this fluctuation.

The results of the tests were plotted on graphs in MATLAB shown in figures 27 and 28. The corresponding MATLAB scripts written to plot these graphs are displayed in appendices I and J.

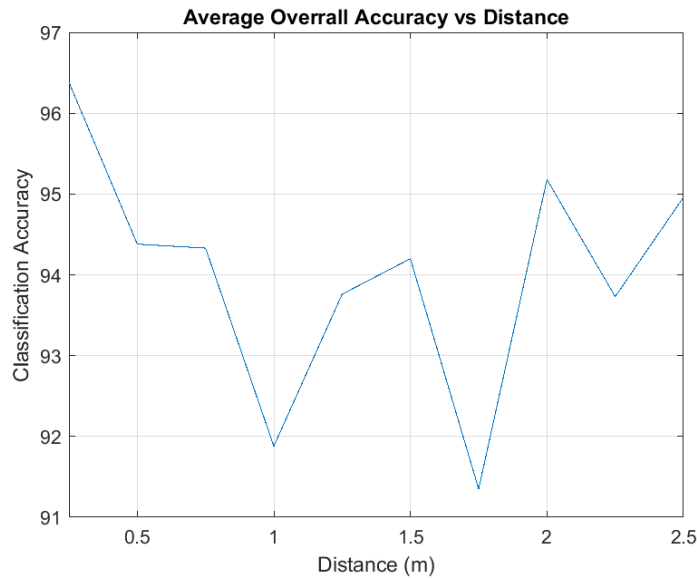


Figure 27: MATLAB graph showing average overall accuracy against distance between radios

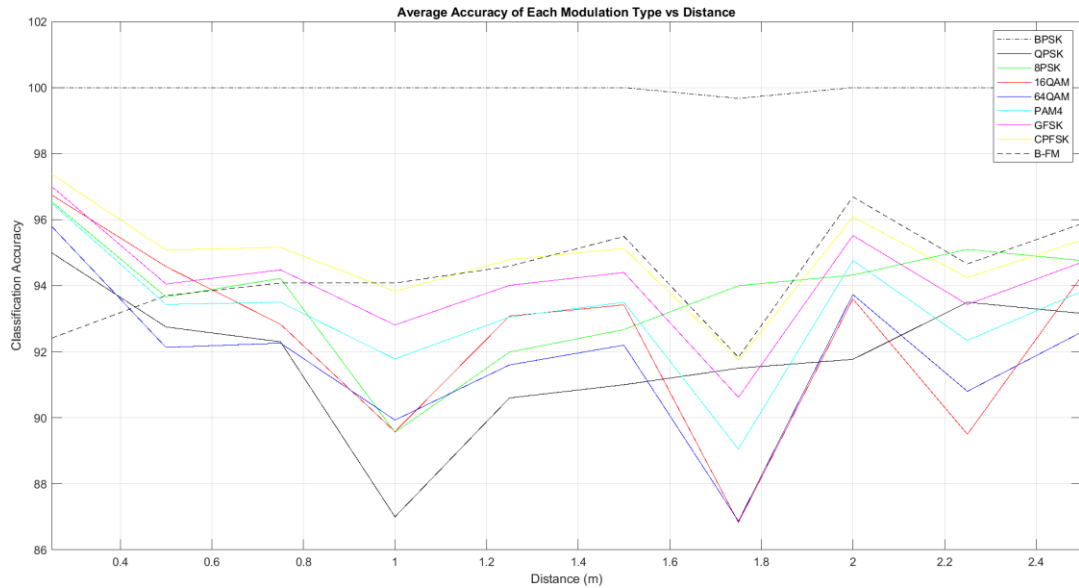


Figure 28: MATLAB graph showing average accuracy of each modulation type against distance between radios

By observing the table and both graphs, it is clear that there is no proportionality between the radio distance and the accuracy of the network, for all modulation types. The overall test accuracy varies between around 91 - 96% and fluctuates between both different distances and repeated tests with the same distance. This may be due to the uncontrolled environment since the experiments were conducted in a household rather than a controlled space like a laboratory. Because of this, there might have been inconsistent noise or interference that attenuated the radios at different times in different capacities. This would explain why the accuracies vary between distances, but the ranking of modulation types stays mostly constant throughout since they were all affected by the same attenuation, shown most clearly in figure 28. While there is no linear pattern based on the distance between the radios, the different lines representing the accuracy for each modulation type all follow a similar shape. For example, the CPFSK modulated signals have consistently higher classification accuracies than 64QAM modulated signals, however they both increase and decrease by roughly the same amount at each change in distance.

As expected, the BPSK signals had the highest accuracies, maintaining 100% for all but one test, that one scoring 99%. This was expected due to its high noise tolerance and simple constellation diagram as previously discussed. Also as expected, there was confusion between the QPSK and 8PSK signals, as well as the 16QAM and 64QAM signals. Again, as previously discussed, this is due to their similar waveform and constellation structures. Although, some unexpected results were the accuracies of the 16QAM and 64QAM modulation types over longer distances. Due to their higher data rate and lower noise tolerance, it was expected there would be a drop off in accuracy as the transmission distance increased. Despite the transmission distance being relatively very low where signals are sent over thousands of kilometres in the real world, the low power of the SDR should make up for this since the signals sent and received are of very low power and therefore low range. It is impressive that the neural network classifies them with such accuracy over longer distances.

The GFSK and CPFSK modulated signals consistently had the highest accuracy (apart from BPSK modulated signals) which may be due to their unique waveform structure since they are both frequency shift keyed compared to phase shift keyed like the most used digital modulation types. Furthermore, the B-FM modulation type had high accuracy scores, from around 92 – 96%. This was surprising due to how the AD9364 firmware could not be configured on the radios and the lowest frequency possible was set

at 370 MHz. Since it is a high frequency for an analogue waveform, it should be expected that the accuracy would decrease over long distance transmissions as high-frequency waveforms do not travel as far.

Overall, the trained neural network has performed relatively well, and its performance did not suffer under shorter or longer transmission distances. This would be a crucial factor if this performance was used for real-world applications and solve issues in spectrum sharing as the network will need to classify signals that have travelled thousands of kilometres. Another important factor would be whether the network can classify signals when obstructed by obstacles. The next test was to see how the classification accuracy was affected when objects of different materials obstructed either radio.

5.3 Obstacle Tests

For this experiment, the radios were placed 0.5 m apart from each other, as seen in figure 25. The radios stayed in those positions while various objects were placed over the transmitter radio to simulate signal transmission through real-world obstacles like buildings or other structures. Obstacles between transmitting and receiving stations can often have large effects on the attenuation of a signal, often causing phase or frequency offsets as well as the scattering or reflection of waveforms. Therefore, the network needs to maintain accuracy in these experiments for it to be used in real-world applications.

The first material to be placed over the transmitter was metal. Due to the nature of working from home because of the COVID-19 pandemic, many household items were used for materials and in this case, a saucepan was used, as seen in figure 29. Like the distance experiment, tests were repeated 3 times for each material. All confusion matrices generated for this exercise can be found in Appendix K.

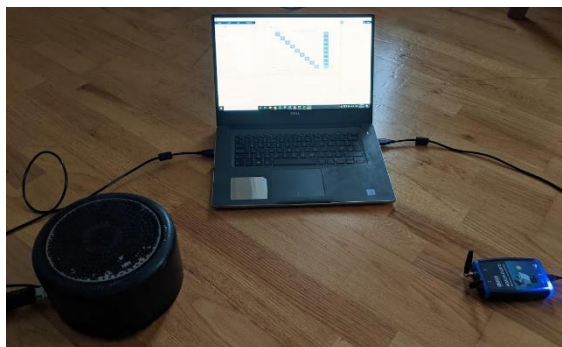


Figure 29: Metal pot covering radio transmitter

| | 16QAM | 64QAM | 8PSK | B-FM | BPSK | CPFSK | GFSK | PAM4 | QPSK | |
|-------|-------|-------|------|------|------|-------|------|------|------|--------|
| 16QAM | 83 | 7 | | | | | | | | 83.0% |
| 64QAM | 25 | 75 | | | | | | | | 75.0% |
| 8PSK | | | 98 | | | | | | 2 | 98.0% |
| B-FM | 19 | 11 | | 70 | | | | | | 70.0% |
| BPSK | | | | | 100 | | | | | 100.0% |
| CPFSK | | | | | | 100 | | | | 100.0% |
| GFSK | | | | | | | 100 | | | 100.0% |
| PAM4 | | | | | | | | 100 | | 100.0% |
| QPSK | | | 30 | | | | | | 70 | 70.0% |
| | 16QAM | 64QAM | 8PSK | B-FM | BPSK | CPFSK | GFSK | PAM4 | QPSK | |

Figure 30: Confusion matrix from metal obstacle test

Figure 30 shows a confusion matrix generated from one of the three tests and there is again lots of fluctuation in the accuracies. The signals with the least accuracy were the analogue B-FM modulated signals, most likely due to the high frequency configured. High-frequency signals do not travel as far and cannot travel through objects as well as low-frequency signals. The 16QAM and 64QAM signals were often mistaken for each other as well as the QPSK and 8PSK signals.

The next material to be used as an obstacle was wood. Similarly, not many materials were available due to the location of the testing, therefore a wooden drawer was used as seen in figure 31. Figure 32 shows the confusion matrix where this time, it shows improvements in the B-FM modulation accuracy. Since 16QAM and 64QAM as well as QPSK and 8PSK are still getting mixed when classified like in the metal test, this might prove that high-frequency analogue signals penetrate wood easier than they penetrate metal.

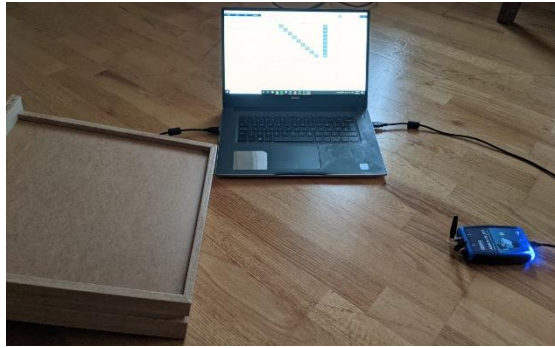


Figure 31: Wooden drawer covering radio transmitter

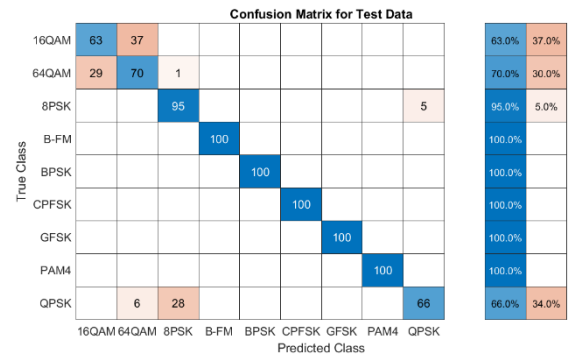


Figure 32: Confusion matrix from wood obstacle test

Figure 33 below shows a glass bowl covering the transmitter radio while figure 34 displays the accuracy of the network from this setup. The obstacle in this experiment seems to have the opposite effect on the neural network accuracy compared to the wooden obstacle. The B-FM modulated signals were all wrongly predicted as 8PSK modulated signals. This result is unexpected as glass is known to be transparent to radio waves.

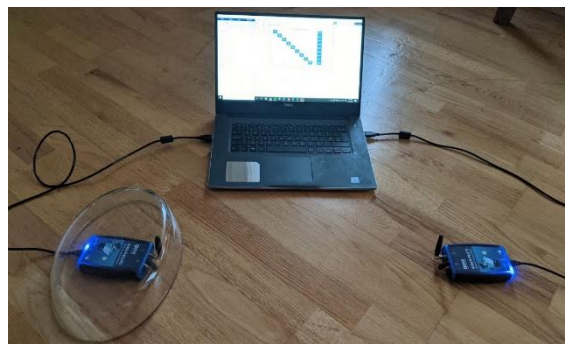


Figure 33: Glass bowl covering radio transmitter

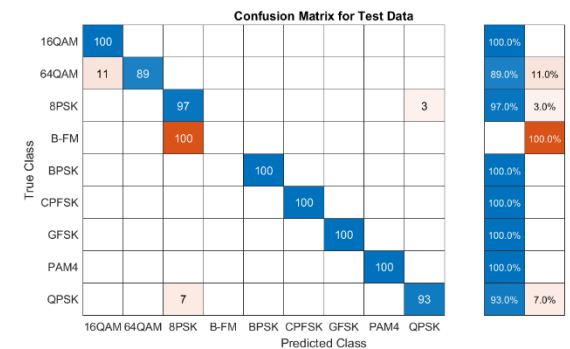


Figure 34: Confusion matrix from glass obstacle test

The next material to act as an obstacle was plastic, in this case, a plastic bowl shown in figure 35. In figure 36, we see the B-FM modulation accuracy has increased, but still has a low accuracy with only 66 out of 100 frames predicted correctly. Other predictions show high degrees of accuracy, suggesting that most of the signals pass through a material like plastic easier than they do metal.

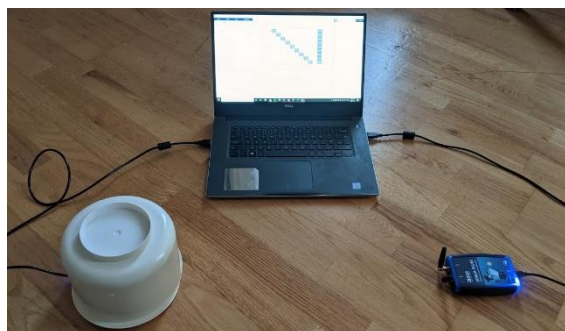


Figure 35: Plastic bowl covering radio transmitter

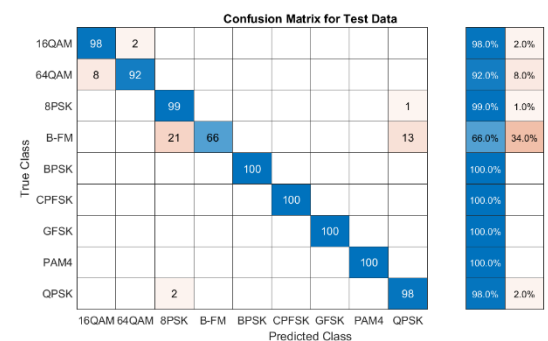


Figure 36: Confusion matrix from plastic obstacle test



Figure 37: Ceramic bowl covering transmitter radio

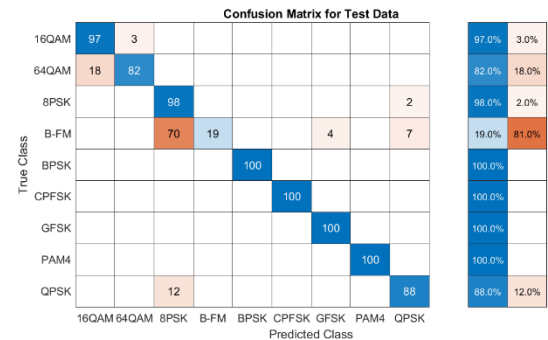


Figure 38: Confusion matrix from ceramic obstacle test

Figure 37 shows the transmitter radio being covered by a bowl made from ceramics and figure 38 shows the corresponding confusion matrix. With the exception of the wooden obstacle test, all the tests with different materials used, follow similar patterns with some small amount of confusion between the 16QAM and 64QAM, and the QPSK and 8PSK modulated signals, but then a large amount of confusion for the B-FM signals due to the high frequency and low penetration. This is the exact same for the next material type used as an obstruction, seen in figures 39 and 40.

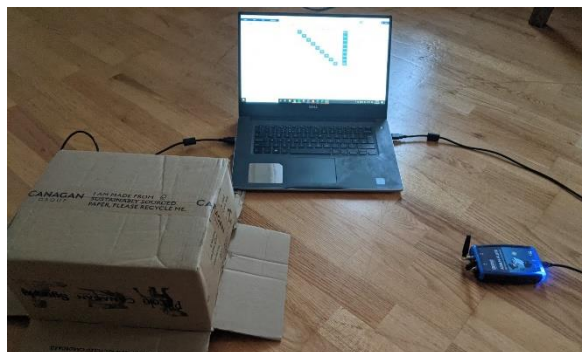


Figure 39: Cardboard box covering transmitter radio

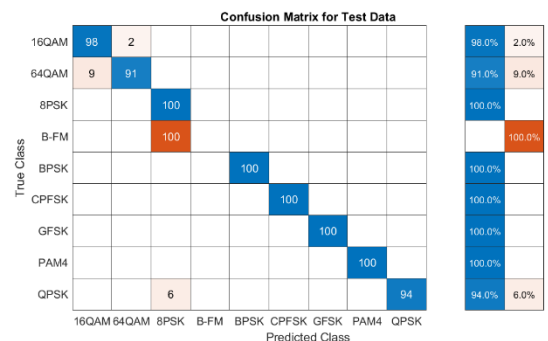


Figure 40: Confusion matrix from cardboard obstacle test

The next material type for this experiment was cardboard and despite being incredibly different on a structural and molecular level than glass, for example, the modulation accuracy is affected very similarly. There is again minor confusion between the quadrature amplitude modulated and phase shift keyed signals, and major confusion for the B-FM signals.

For some materials used as obstacles such as metal and plastic, there were fluctuations between each test, particularly in the B-FM modulated signals. However, in the tests for other materials, there were very few fluctuations in the confusion matrix outputs, showing some consistency which would be preferred in a real-world setting. While many of the materials used as obstacles in this experiment would be found in the real world, the obstacle would usually come in the form of a combination of these materials. A building isn't only made of metal, for example, there may also be ceramics, glass, and other composite materials. It is also worth noting that there are many different types of a particular material so just because the high-frequency B-FM signals could be correctly classified with a wooden drawer between the radios, this does not mean that B-FM signals will be correctly classified with all types of wood acting as obstacles. The next test aims to address these issues where there is no doubt of its application in the real world. The radios were placed on either side of a thick wall of width 0.2 m, as seen in figure 41. The script was again run three times and confusion matrices were generated, one of them shown in figure 42.

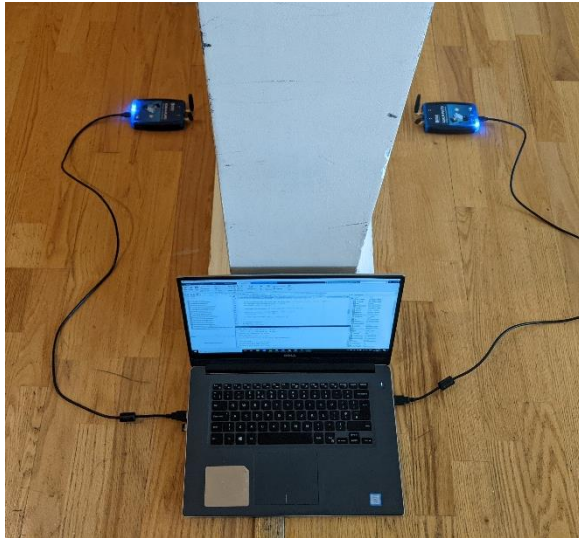


Figure 41: Wall between transmitter and receiver radios

The results of this test would give more conclusive results as to whether the neural network designed in this project, is capable of modulation classification in the real world as the wall would effectively simulate an obstacle like a building since it is one itself. The wall is made from multiple materials such as wood, plaster (sand and water), and perhaps insulation like glass wool.

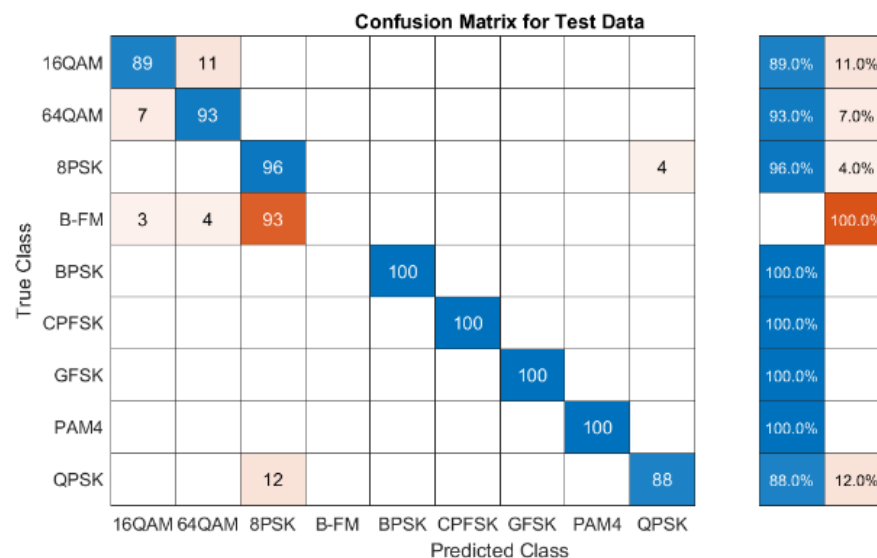


Figure 42: Confusion matrix when the wall is between transmitter and receiver radios

Rather surprisingly, the results of this test resemble those previously with just one material blocking the transmitter radio. There is again some confusion between the QAM signals and the PSK signals, and there is again major confusion in the B-FM modulated signal, most likely due to the thickness of the wall that the high-frequency signal cannot penetrate. Since many of the confusion matrices showed similar results, the accuracies generated in the MATLAB command window were plotted for each modulation type and each repetition, similar to the distance vs accuracy test. Table 4 below shows the average accuracies taken across the three tests for each modulation type and each material. The full table with every result can be found in Appendix L

| Mod Type | Type of Material | | | | | | |
|----------------|------------------|--------------|--------------|--------------|--------------|--------------|--------------|
| | Metal | Wood | Glass | Plastic | Ceramic | Cardboard | Between Wall |
| BPSK | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 |
| QPSK | 90.83 | 87.00 | 94.00 | 97.67 | 94.83 | 95.83 | 94.33 |
| 8PSK | 93.21 | 89.67 | 94.78 | 97.89 | 95.44 | 96.33 | 94.89 |
| 16QAM | 91.67 | 87.25 | 95.50 | 97.67 | 95.83 | 96.25 | 94.17 |
| 64QAM | 90.87 | 86.47 | 93.87 | 95.53 | 93.80 | 95.07 | 93.80 |
| PAM4 | 92.38 | 88.72 | 94.88 | 96.28 | 94.83 | 95.89 | 94.83 |
| GFSK | 93.48 | 90.24 | 95.62 | 96.81 | 95.57 | 96.47 | 95.57 |
| CPFSK | 94.29 | 91.54 | 96.17 | 97.21 | 96.13 | 96.92 | 96.13 |
| B-FM | 91.73 | 90.74 | 85.48 | 92.56 | 86.26 | 86.15 | 85.44 |
| Overall | 93.16 | 90.18 | 94.48 | 96.85 | 94.74 | 95.44 | 94.35 |

Table 4: Classification accuracy for each modulation type against the material used as an obstacle

From this set of results, patterns can be observed, and it can be determined what materials acted as the best or worst obstacles. The material with the worst overall accuracy with just 90.18% was wood. The wooden obstacle achieved the lowest accuracy across most modulation types with the exception of B-FM modulated signals where wood scored amongst the highest. With these results, it can be argued that wood is better at blocking signals than other materials.

The material with the highest associated accuracy was plastic with consistently high accuracies for each modulation type. It would be sensible to assume that the plastic material is the easiest material out of these options to penetrate and it is the worst at blocking signals, even the B-FM signals had high classification accuracies at 92.56%.

BPSK modulated signals again stayed constant at 100% accuracy while accuracies for other modulated signals varied, perhaps based on the ability to penetrate the material or the thickness or shape of the obstacle. In the case of the effect that the shape of the obstacle has on the accuracy, the wood and cardboard obstacles were cube-shaped whereas the metal, glass, plastic, and ceramic were round. To notice the effect the shape has, there would need to be some correlation between the wood and cardboard obstacles, as well as all the round obstacles. However, there is no pattern in this regard since the classification accuracies of the network when there is a wood obstacle are in no way similar to that of the cardboard obstacle. Furthermore, there is no correlation between the thickness of the material and the classification accuracy since all materials are of similar thickness.

The modulation types most crucial to the development of 5G communications out of these are the 16QAM and 64QAM signals. Worryingly, these modulated signals were classified better among obstacles that are less common in environments like cardboard, plastic, and ceramic, whereas the accuracy is lower when more common obstacles are introduced such as metal and wood. This gives an insight into the difficulties working with higher data rate signals like these and when paired with high-frequency carrier signals, a lot of power is needed when transmitting to provide a suitable signal to noise ratio.

As previously discussed, these results may not reflect all types of their respective materials as the different origin and manufacturing process might change the network accuracy for that type of obstacle. For example, multiple types of wood could have been explored, however, due to the nature of working from home, household items were used instead of perhaps more effective obstacles.

The neural network in this experiment performed relatively well by maintaining network accuracy in the region of 90 – 96% depending on the obstacle used. It is assumed that the variations in accuracy are due to the type of material alone and no other factor like the shape, size, or thickness of each material.

6 Project Plan

The project plan and schedule is outlined in the Gantt chart shown below.

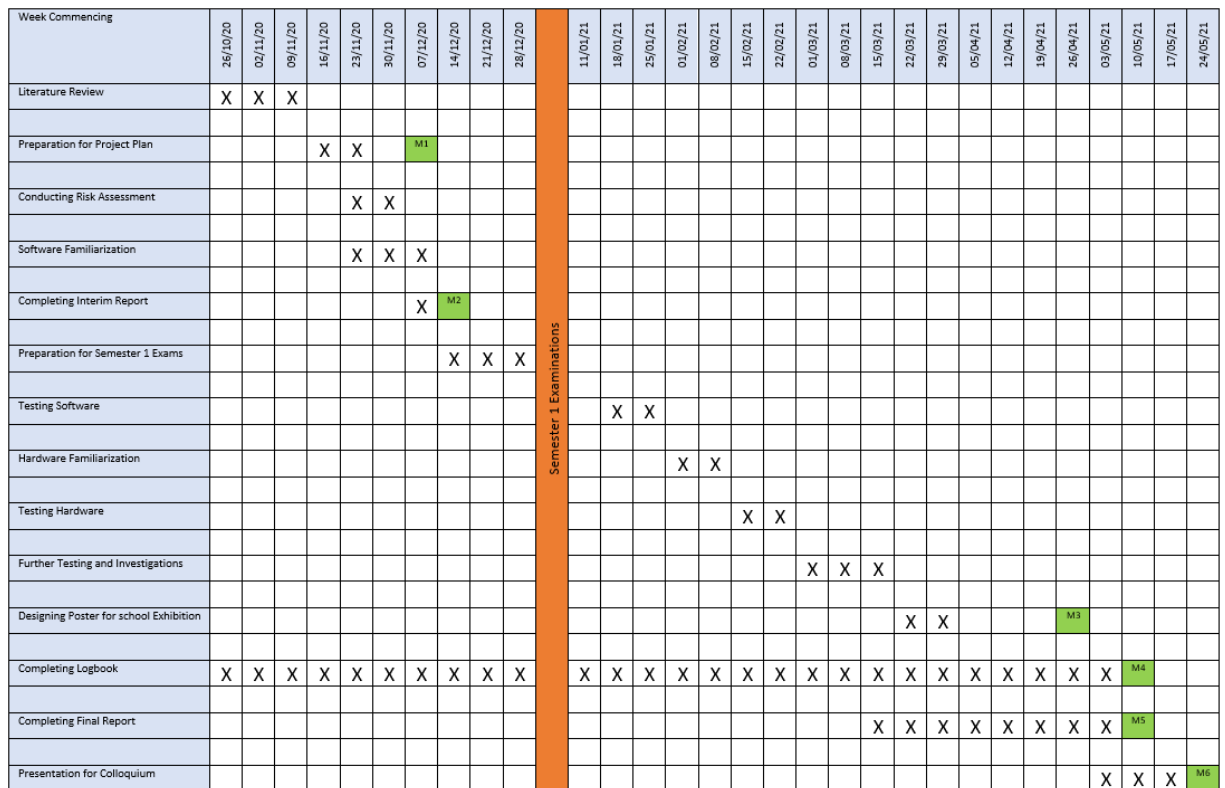


Figure 43: Gantt chart showing project plan

M1 – Submission of Project Plan: Milestone met

M2 – Submission deadline for interim report: Milestone met

M3 – Submission deadline for school exhibition poster: Milestone met

M4 – Submission deadline for logbook entries: Milestone not met (testing not finished)

M5 – Submission deadline for Final Report: Milestone not met (submission 1 week late)

M6 – Presentation for Colloquium: N/A

The Gantt chart above shows that my project started on 26th October 2020, when I took my first step towards creating the project, to 28th May 2021, the date that the final piece of work must be submitted. Throughout this timeline, the project mostly stayed on track with the on-time submission of multiple deadlines. The literature review was conducted early on in the project including research on subjects relevant to the objectives set out at the start, and also research on hardware and software options.

MATLAB was chosen due to the numerous relevant toolboxes and tutorials, and the proficiency of the student when compared to alternatives like Python which both reduced the possible delays in schedule. The ADALM-Pluto SDR was chosen for hardware due to the cost and useful features of the device. A project plan was submitted on the entry M1 and a health and safety review was carried out in conjunction with the seminar provided. Following the MATLAB tutorial “Modulation Classification With Deep Learning”, I explored the abilities and functions of the pre-trained convolutional neural network designed by MATLAB and worked on improving it in a span of 3 weeks. This led to the on-time submission of the interim report on the 17th of December, explaining my work thus far.

Following the winter break and examination period, I continued working on my project and carried on designing, training, and testing the neural network through software alone. Eventually, on the 11th of February, I ordered an ADALM-Pluto SDR to begin testing with live over-the-air signals. Through rigorous and extensive testing, it was realised that many tests could not be performed as expected with just one radio. Therefore, thanks to the generosity of the university, they allowed me to go over budget and order another radio. Through many struggles of configuring both radios to operate on one computer, diverse testing was carried out including transmission over distance and through obstacles.

Throughout the testing stages, the poster for the school exhibition was being created, showcasing the project in a brief but informative nature. Unfortunately, due to the COVID-19 pandemic, there was no in-campus exhibition event and posters were submitted and available to view online. The poster was submitted on the 26th of April. The project was worked on up to the 11th of May but due to many unforeseen setbacks in testing and the hectic university schedule, both the logbook and final dissertation were submitted a week late on the 18th of May.

7 Ethical Review

After discussions with my supervisor, it was decided that an ethical review was not required as there were no human participants involved in this project. Therefore, the completion of an ethical compliance form was not necessary.

8 Health and Safety

To ensure the safety of all university campus users as well as all individuals when working from home, a three-hour health and safety seminar was held by the School Health & Safety Co-Ordinator in Semester 1. This session involved an overview of health and safety and the legal practices surrounding it. We were also informed on how we could identify hazards in the work environment and how we could eliminate these hazards. We were told each student is personally responsible for their own health and safety, and others who could be potentially impacted by the project underhand.

Therefore, we had to conduct a risk assessment to assess any risks in the environments where the project is conducted. This involved identifying any hazards in our surroundings, who could be harmed in this situation and in what matter. After identifying all risks, control measures were identified to try and reduce the chances of a potential hazard taking place, or to mitigate the effects of this hazard. This involved rating each hazard by the likelihood of it occurring along with the severity of that hazard of impact. If this rating was too high, further control measures were to be implemented and again rated on their risk. These further control measures have an associated due date. These findings were recorded on a risk assessment form.

Risk Assessment Record: University of Sussex / Software Defined Radio, MATLAB / Year 3 Individual Project

| | | | |
|----------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|-----------------------|---------------------------------------------------------------------------------------------------------------------|
| Location / Area: | Richmond 3B3 & 4B9/10, Chichester Labs 1, 2 & Future Technology Lab (FTL), 81 Trinity Square, Margate, Kent, United Kingdom | | |
| Dept/School: | Eng. | Building manager: | School Health & Safety Co-ordinators (Margarita Steinberg), The Quality, Safety, Health & Environment (QSHE) Office |
| Assessor: | Harry Bradford | Position: | Student |
| Date of assessment: | 28/09/2020 | Colleagues consulted: | Professor Maziar Nekovee (Project Supervisor) |
| Valid until: | 28/05/2019 | Due for review (Y/N) | Y |
| Summary of project or activity | An electronics project that entails the integration of deep learning techniques into 5G applications for performance optimization | | |
| Overview of main risks involved: | Risks associated with long periods of computer use and exposure to electronic devices | | |
| Overall Risk Rating: | Low Risk | | |

| Hazards Identified *** | Who may be harmed? | Existing Risk Controls * | Likelihood (L) * | Severity (S) * | Risk Rating (LxS) * | Further Risk Controls required (if Risk Rating is 5 or above) * | Residual Likelihood (RL) | Residual Severity (RS) | Residual Risk Rating (RL x RS) | Responsible for implementing further Risk Control measures | Due Date ** | Date Completed |
|--------------------------------------------|--------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|----------------|---------------------|-----------------------------------------------------------------|--------------------------|------------------------|--------------------------------|------------------------------------------------------------|-------------|----------------|
| | | | | | | | N/A | N/A | N/A | | | |
| Electrical Shocks/Burns | Myself | All PCs and electronic equipment used in accordance with the manufacturer's instructions. Defective plugs, cables and equipment are removed. Sufficient power sockets provided to reduce need for extension cables. | 1 | 2 | 2 | N/A | N/A | N/A | N/A | N/A | N/A | N/A |
| Slips/Trips/Falls | Myself | Organize room so there is space to move. Trailing cables positioned neatly away from doors and walkways | 2 | 1 | 2 | N/A | N/A | N/A | N/A | N/A | N/A | N/A |
| Headache/ Eyestrain from computer use | Myself | Screen tilted slightly to avoid reflections or glare. Screen positioned at a suitable distance and at eye level. Regular breaks. Adequate lighting provided and maintained. | 1 | 1 | 1 | N/A | N/A | N/A | N/A | N/A | N/A | N/A |
| Repetitive Strain Injury from computer use | Myself | Keyboard and Mouse positioned correctly. Chair is adjusted to be stable and in a comfortable position. Regular breaks are taken. | 1 | 1 | 1 | N/A | N/A | N/A | N/A | N/A | N/A | N/A |

* Risk Rating:
This form assumes 1-5 scoring system for Likelihood & Severity, and 1-25 scoring for Risk Rating. (see [Guidance Note on Risk Assessments \(RAs\)](#) for more details)
Within 1-25 Risk Rating scoring, Risk Rating score of 1-4 is considered Acceptable, 5-9 Adequate.
Risk Rating of 10 and above indicates that further risk control measures must be introduced until the Residual Risk Rating has been reduced to at least below 10.

** Due Date records the date by which all the identified risk control measures need to be implemented.



*** For software-based projects, only DSE, and trips/slips/falls apply
Lone work is never allowed for practical work in final year projects.

Contingency plan of action

| In the event of | Take this action | Involve |
|-----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------|
| Fire | If a fire is observed: 1) Raise any fire alarm in the vicinity 2) If an exit is clear and if safe to do so, use a nearby fire extinguisher to tackle the fire 3) Escape through the nearest fire escape door 4) Call emergency services | Fire and Rescue Services All others in the building in danger of fire |

Safety Protocols
N/A

Approval for student projects

| Academic supervisor / Course convenor | | | |
|---------------------------------------|----------------|--------------------------------------------------------------------------------------|------------|
| Name | | Signature | Date |
| Prof. Maziar Nekovee | |  | 16/12/2020 |
| Personnel Involved | | | |
| Role | Print name | Signature | Date |
| Student | Harry Bradford |  | 12/11/20 |

9 Conclusions and Recommendations

At the start of this project, I set out to design a neural network for the task of modulation classification. This was achieved by, first of all, conducting a literature review to study the theories around communication technologies as well as deep learning, neural networks and how they all tie together. After reviewing hundreds of articles and publications on the subjects, I had sufficient knowledge to start conducting experiments. Following this, the different options for both software and hardware were analysed where MATLAB and the ADALM-Pluto SDR were respectively chosen.

The pre-trained convolutional neural network designed by MATLAB was examined and it was decided to adopt transfer learning as a technique to design a CNN. Numerous frames of various modulated signals were generated and their features extracted for training, validation and testing the pre-trained CNN. The CNN structure was modified and using the generated frames, the CNN was trained and had an improved classification accuracy of 92%. Testing frames were used to test the CNN and a confusion matrix was plotted showing the network's predictions of the synthetic modulated signals.

The performance of the CNN was then analysed with real over-the-air signals transmitted and received by the ADALM-Pluto SDRs, rather than synthetic waveforms generated in MATLAB. This performance had improved the classification of the synthetic data signals with an overall accuracy of 96.43%. To observe the network's abilities in real-world applications, it was decided to see how the network changed under various factors such as transmission distance and placement of obstacles.

Even during these experiments, the performance of the CNN maintained a high level, never falling below 90% accuracy. These results are promising for the future of communications technology as the CNN was able to classify modulated signals with high degrees of accuracy even under conditions designed to attenuate and distort the signals. Up to 2.5 m, there was no significant loss of accuracy as the transmission distance between the two radios increased, and there was no significant loss of accuracy when various materials were placed between the radios. However, it was discovered that signals with higher frequencies like the B-FM modulated signal, had a harder time penetrating the materials and accuracy lowered because of this.

On the other hand, the quality of experiments could have been improved throughout this project. Due to the COVID-19 pandemic, students have had to conduct experiments in their home which is not a controlled environment like a laboratory. Many possible extraneous variables out of my control could have had an impact on the experiments, perhaps causing the fluctuations in the distance against the accuracy experiment. Furthermore, the materials used as obstacles were household objects, not designed for experimentation and therefore, the quality of the results may have been impacted by this. In a more controlled experiment, all obstacles would have been of the same size, shape and thickness which may have increased or decreased the network performance. Although, these changes should have only caused marginal changes and should not take away from the network performance.

Due to the success of these experiments, it could be claimed that neural networks are ready to have a part to play in communications technology. Despite the low power output of the SDRs and the small amount of data used to feed the neural network, the classification accuracy consistently maintained a high level. This means deep learning has a role in spectrum sharing applications and can help solve the dilemma of spectrum resource management in the future. Accuracies will only improve with thousands or millions of waveform samples fed into a more complex network in a larger-scale project. Other advances in technology such as data capacities and cloud storage will only boost the implementation of this kind of project.

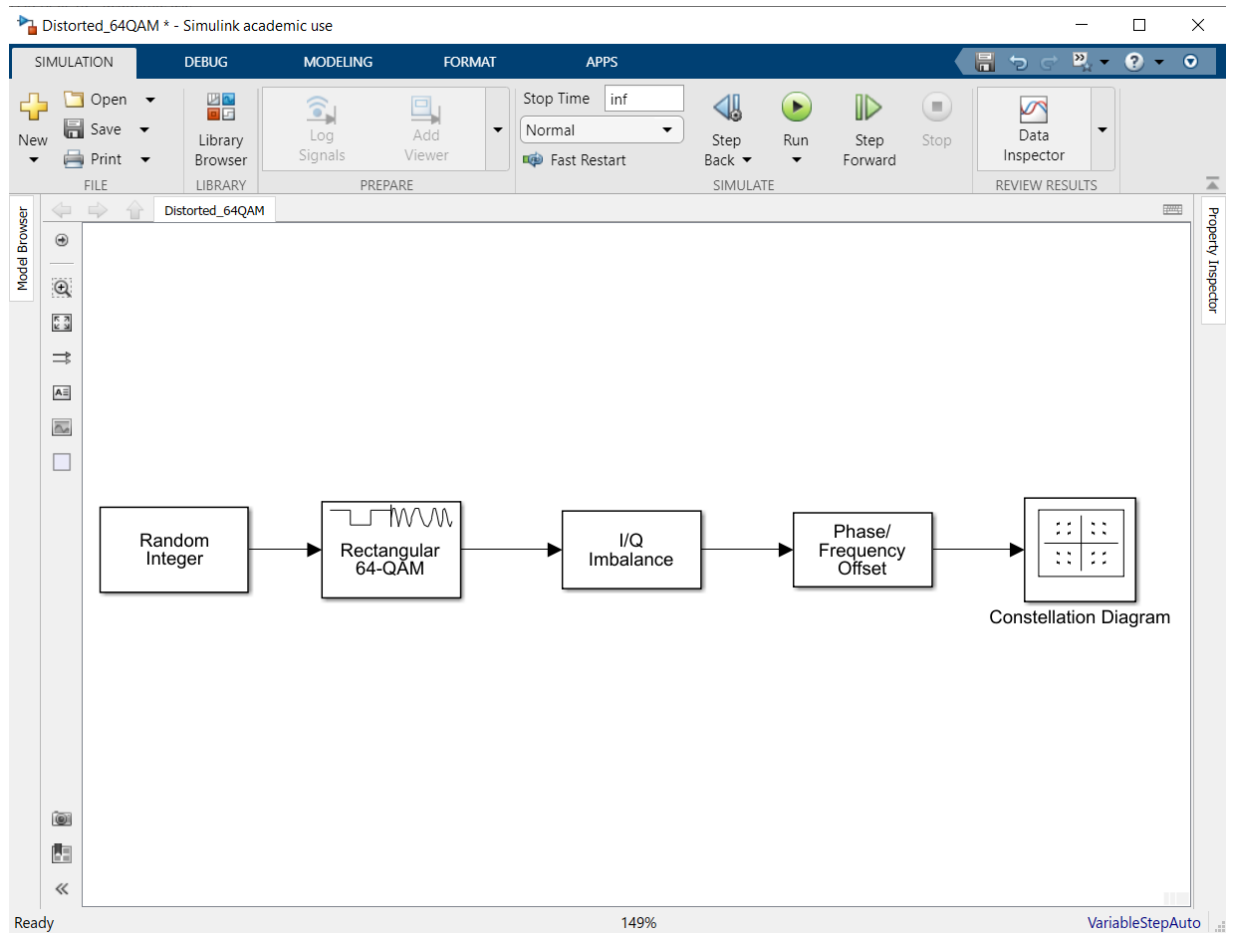
10 References

- [1] Vodafone, “How Fast Is 5G”, 2020. [online] Available from: <https://www.vodafone.co.uk/network/5g#:~:text=5G%20is%20around%2010%20times,over%2015%20minutes%20on%204G> [accessed 05/05/21]
- [2] Thales, “5G vs 4G, What’s the Difference?”, 08/10/2020. [online] Available from: <https://www.thalesgroup.com/en/worldwide-digital-identity-and-security/mobile/magazine/5g-vs-4g-whats-difference#:~:text=5G%20up%20to%20100%20times,for%20an%20increasingly%20connected%20society>. [accessed 05/05/21]
- [3] James Rogerson, 5g.co.uk, “What is 5G?”, 21/10/20. [online] Available from: <https://5g.co.uk/guides/what-is-5g/#5G%20latency> [accessed 05/05/21]
- [4] Connor Craven, sdxcentral, “What Is the 5G Spectrum? Definition”, 18/01/20 [online] Available from: <https://www.sdxcentral.com/5g/definitions/what-is-5g-spectrum/#:~:text=The%205G%20spectrum%20is%20a,stations%20to%20the%20data's%20endpoint>. [accessed 05/05/21]
- [5] Britannica, “Radio-Frequency Spectrum”, 2013 [online] available from: <https://www.britannica.com/science/radio-frequency-spectrum> [accessed 07/05/21]
- [6] Samsung Newsroom, “Samsung Brings 5G Indoors with New Commercial 5G mmWave Small Cell for In-building Use”, 24/09/20 [online] Available from: <https://news.samsung.com/us/5g-indoor-mmwave-small-cell/> [accessed 14/05/21]
- [7] NIST, “NIST Helps Facilitate First-Ever Spectrum Sharing Between Military and Commercial Wireless Users”, 22/03/18 [online] Available from: <https://www.nist.gov/news-events/news/2018/03/nist-helps-facilitate-first-ever-spectrum-sharing-between-military-and> [accessed 14/05/21]
- [8] Charlotte Yates, flypaper, “What Actually IS the Difference Between AM and FM Radio”, 20/03/19 [online] Available from: <https://flypaper.soundfly.com/discover/what-actually-is-the-difference-between-am-and-fm-radio/> [accessed 06/05/21]
- [9] Shree Prakash Singh, S. Sengar, Rochak Bajpai, Sridhar Iyer, “Next-Generation Variable-Line-Rate Optical WDM Networks: Issues and Challenges”, March 2014, pg 338, Journal of Optical Communications.
- [10] Ghafour Amouzad Mahdiraji, Ahmad Fauzi Abas, “Advanced Modulation Formats and Multiplexing Techniques for Optical Telecommunication Systems”, March 2010, pg 16, UCSI University & Universiti Putra Malaysia.
- [11] Tutorangel. Video Tutorial “3 Modulation”, 29/01/14 [online] Available from: https://www.youtube.com/watch?v=mfSWO3TpGq4&list=PLwor3jxjzv8wAUqZezHguZtmSZJO_3j1R&index=3&t=598s [accessed 14/05/21]
- [12] Ravindra Parmar, Towards Data Science, “Training Deep Neural Networks”, 11/09/18 [online] Available from: <https://towardsdatascience.com/training-deep-neural-networks-9fdb1964b964> [accessed 14/05/21]

- [13] MathWorks, eBook, “*Introducing Deep Learning with MATLAB*”, 2018 [online] Available from: [Introducing Deep Learning with MATLAB \(mathworks.com\)](https://uk.mathworks.com/help/deeplearning/ug/introducing-deep-learning-with-matlab.html) [accessed 06/05/21]
- [14] MathWorks, “*Supported Hardware – Software Defined Radio*”, [online] Available from: <https://uk.mathworks.com/help/comm/supported-hardware-software-defined-radio.html> [accessed 06/05/21]
- [15] Xilinx, “*Xilinx Zynq-7000 SoC ZC706 Evaluation Kit*”, [online] Available from: <https://www.xilinx.com/products/boards-and-kits/ek-z7-zc706-g.html> [accessed 06/05/21]
- [16] Ettus, “*USRP E310*”, [online] ettus.com/all-products/e310/ [accessed 06/05/21]
- [17] MathWorks, “*USRP E310 Support from Communications Toolbox*”, [online] available from: <https://uk.mathworks.com/hardware-support/usrp-e310.html> [accessed 06/05/21]
- [18] MathWorks, “*RTL-SDR Support from Communications Toolbox*”, [Online] Available from: <https://uk.mathworks.com/hardware-support/rtl-sdr.html> [Accessed 06/05/21].
- [19] The Pi Hut, “*Software Defined Radio Receiver USB Stick - RTL2832 w/R820T*”, [online] Available from: https://thepihut.com/products/software-defined-radio-receiver-usb-stick-rtl2832-w-r820t?variant=27739418385¤cy=GBP&utm_medium=product_sync&utm_source=google&utm_content=sag_organic&utm_campaign=sag_organic&gclid=CjwKCAjwnPOEBhA0EiwA609ReYsm8okaz5tqlKeYIPLcbO_8O0aUuUxSFgO03H-luGnzeirIKfx2SRoCafUQAvD_BwE [accessed 14/05/21]
- [20] MathWorks, “*ADALM-PLUTO Radio Support from Communications Toolbox*”, [online] Available from: <https://uk.mathworks.com/hardware-support/adalm-pluto-radio.html> [accessed 14/05/21]
- [21] Mouser Electronics, “*Analog Devices ADALM-Pluto*”, [online] Available from: <https://www.mouser.co.uk/ProductDetail/Analog-Devices/ADALM-PLUTO?qs=xbccQsLEe0ffoUoi%2FjfiWA%3D%3D> [accessed 14/05/21]
- [22] Digi-Key, “*ADALM-Pluto*”, [online] available from: https://www.digikey.co.uk/product-detail/en/analog-devices-inc/ADALM-PLUTO/ADALM-PLUTO-ND/6624230?gclid=CjwKCAjwnPOEBhA0EiwA609ReZOsKnYvkEz4B-qJwyhQ4NswsV_YA9MF7z_PTVL7vICcZ0asNhcfIxoCyfEQAvD_BwE [accessed 14/05/21]
- [23] RTL-SDR.com, “*ADALM-PLUTO SDR: UNBOXING AND INITIAL TESTING*”, 16/08/17 [online] Available from: <https://www.rtl-sdr.com/adalm-pluto-sdr-unboxing-and-initial-testing/> [accessed 14/05/21]
- [24] Analog Devices, “*ADALM-Pluto SDR Active Learning Module*”, 2017, pg 2.
- [25] MathWorks, “*Modulation Classification with Deep Learning*”, 2021 [online] available from: <https://uk.mathworks.com/help/deeplearning/ug/modulation-classification-with-deep-learning.html> [accessed 12/05/21]
- [26] MathWorks, “*transmitRepeat*”, 2021 [online] available from: <https://uk.mathworks.com/help/supportpkg/plutoradio/ref/comm.sdrtxpluto.transmitrepeat.html> [accessed 17/05/21]

11 Appendices

Appendix A: Simulink Model for Simulated 64QAM Modulated Signals



Appendix B: MATLAB Script for Untrained CNN Predictions for Relevant Digital Modulation Types

```
modulationTypes = categorical(["BPSK", "QPSK", "8PSK", ...  
    "16QAM", "64QAM", "PAM4", "GFSK", "CPFSK", "B-FM", ...  
    "DSB-AM", "SSB-AM"]);  
  
load trainedModulationClassificationNetwork  
openvar('trainedNet')  
  
rng(123456)  
  
testChannel = helperModClassTestChannel(...  
    'SampleRate', 200e3, ...  
    'PathDelays', [0 1.8 3.4] / 200e3, ...  
    'AveragePathGains', [0 -2 -10], ...  
    'KFactor', 4, ...  
    'MaximumDopplerShift', 4, ...  
    'SNR', 30, ...  
    'MaximumClockOffset', 5, ...  
    'CenterFrequency', 902e6);  
  
%BPSK  
randomBits = randi([0 1], 1024, 1);  
BPSKmod = pskmod(randomBits,2);  
  
filterCoeffs = rcosdesign(0.35, 4, 8);  
BPSK_tx = filter(filterCoeffs,1,upsample(BPSKmod,8));  
BPSK_rx = testChannel(BPSK_tx);  
  
scope = timescope(2, 200e3, 'YLimits', [-1 1], 'ShowGrid', true, ...  
    'LayoutDimensions', [2 1], 'TimeSpan', 41e-3);  
scope(BPSK_tx, BPSK_rx)  
  
unknownFrames = helperModClassGetNNFrames(BPSK_rx);  
[BPSKprediction,BPSKscore] = classify(trainedNet,unknownFrames);  
openvar('BPSKprediction')  
helperModClassPlotScores(BPSKscore,modulationTypes)  
  
% PAM4  
randomBits = randi([0 3], 1024, 1);  
mod_PAM4 = pammod(randomBits, 4);  
  
filterCoeffs = rcosdesign(0.35, 4, 8);  
  
tx_PAM4 = filter(filterCoeffs,1,upsample(mod_PAM4,8));  
rx_PAM4 = testChannel(tx_PAM4);  
  
scope = timescope(2, 200e3, 'YLimits', [-2 2], 'ShowGrid', true, ...  
    'LayoutDimensions', [2 1], 'TimeSpan', 41e-3);  
scope(tx_PAM4, rx_PAM4)
```

```

unknownFrames = helperModClassGetNNFrames(rx_PAM4);
[prediction_PAM4,score_PAM4] = classify(trainedNet,unknownFrames);
openvar('prediction_PAM4')
helperModClassPlotScores(score_PAM4,modulationTypes)

%16QAM
randomBits = randi([0 15], 1024, 1);
mod_16QAM = qammod(randomBits, 16, 'UnitAveragePower', true);

filterCoeffs = rcosdesign(0.35, 4, 8);

tx_16QAM = filter(filterCoeffs,1,upsample(mod_16QAM,8));
rx_16QAM = testChannel(tx_16QAM);

scope = timescope(2, 200e3, 'YLimits', [-1 1], 'ShowGrid', true, ...
    'LayoutDimensions', [2 1], 'TimeSpan', 41e-3);
scope(tx_16QAM, rx_16QAM)

unknownFrames = helperModClassGetNNFrames(rx_16QAM);
[prediction_16QAM,score_16QAM] = classify(trainedNet,unknownFrames);
openvar('prediction_16QAM')
helperModClassPlotScores(score_16QAM,modulationTypes)

%64QAM
randomBits = randi([0 63], 1024, 1);
mod_64QAM = qammod(randomBits, 64, 'UnitAveragePower', true);

filterCoeffs = rcosdesign(0.35, 4, 8);

tx_64QAM = filter(filterCoeffs,1,upsample(mod_64QAM,8));
rx_64QAM = testChannel(tx_64QAM);

scope = timescope(2, 200e3, 'YLimits', [-1 1], 'ShowGrid', true, ...
    'LayoutDimensions', [2 1], 'TimeSpan', 41e-3);
scope(tx_64QAM, rx_64QAM)

unknownFrames = helperModClassGetNNFrames(rx_64QAM);
[prediction_64QAM,score_64QAM] = classify(trainedNet,unknownFrames);
openvar('prediction_64QAM')
helperModClassPlotScores(score_64QAM,modulationTypes)

% QPSK
randomBits = randi([0 3], 1024, 1);
QPSKmod = pskmod(randomBits, 4, pi/4);

filterCoeffs = rcosdesign(0.35, 4, 8);

QPSK_tx = filter(filterCoeffs,1,upsample(QPSKmod,8));
QPSK_rx = testChannel(QPSK_tx);

scope = timescope(2, 200e3, 'YLimits', [-1 1], 'ShowGrid', true, ...
    'LayoutDimensions', [2 1], 'TimeSpan', 41e-3);

```

```

scope(QPSK_tx, QPSK_rx)

unknownFrames = helperModClassGetNNFrames(QPSK_rx);
[QPSKprediction,QPSKscore] = classify(trainedNet,unknownFrames);
openvar('QPSKprediction')
helperModClassPlotScores(QPSKscore,modulationTypes)

%8PSK
randomBits = randi([0 7], 1024, 1);
mod_8PSK = pskmod(randomBits, 8);

filterCoeffs = rcosdesign(0.35, 4, 8);

tx_8PSK = filter(filterCoeffs,1,upsample(mod_8PSK,8));
rx_8PSK = testChannel(tx_8PSK);

scope = timescope(2, 200e3, 'YLimits', [-1 1], 'ShowGrid', true, ...
    'LayoutDimensions', [2 1], 'TimeSpan', 41e-3);
scope(tx_8PSK, rx_8PSK)

unknownFrames = helperModClassGetNNFrames(rx_8PSK);
[prediction_8PSK,score_8PSK] = classify(trainedNet,unknownFrames);
openvar('prediction_8PSK')
helperModClassPlotScores(score_8PSK,modulationTypes)

```

Appendix C: MATLAB Script for Frame Generation and Feature Extraction

```
%Generating frames for each mod type

trainNow = false;
if trainNow == true
    framesPerModType = 10000;
else
    framesPerModType = 500;
end

trainingSamples = 80;    % 80% of samples used for training

validationSamples = 10; % 10% of samples used for validation

testSamples = 10;    % 10% of samples used for testing

sps = 8;    % 8 samples per symbol
spf = 1024; % 1024 samples per frame
symbolsPerFrame = 128; % symbols per frame = spf/sps = 1024/8
fs = 200e3;    % Sample rate
fc = [902e6 100e6]; % Centre Frequencies
SNR = 30;    % Signal to noise ratio = 30 dB

std = sqrt(10.^(-SNR/10)); % Noise standard deviation

awgnChannel = comm.AWGNChannel('NoiseMethod', 'Signal to noise ratio
(SNR)', 'SignalPower', 1, 'SNR', SNR);

multipathChannel = comm.RicianChannel('SampleRate', fs, 'PathDelays',
[0 1.8 3.4]/fs, 'AveragePathGains', [0 -2 -10], 'KFactor', 4,
'MaximumDopplerShift', 4);

maxDeltaOff = 5;
deltaOff = (rand()*2*maxDeltaOff) - maxDeltaOff;
C = 1 + (deltaOff/1e6);

offset = -(C-1)*fc(1);
frequencyShift = comm.PhaseFrequencyOffset('SampleRate', fs,
'FrequencyOffset', offset);

testChannel = helperModClassTestChannel('SampleRate', fs, 'SNR', SNR,
'PathDelays', [0 1.8 3.4]/fs, 'AveragePathGains', [0 -2 -10],
'KFactor', 4, 'MaximumDopplerShift', 4, 'MaximumClockOffset', 5,
'CenterFrequency', 902e6);

rng(1235)
tic

numModulationTypes = length(modulationTypes);

transDelay = 50;
dataDirectory = fullfile(tempdir, "ModClassDataFiles");
```



```

disp("Data file directory is " + dataDirectory)

fileNameRoot = "frame";

% Check if data files exist
dataFilesExist = false;
if exist(dataDirectory, 'dir')
    files = dir(fullfile(dataDirectory, sprintf("%s*", fileNameRoot)));
    if length(files) == numModulationTypes*framesPerModType
        dataFilesExist = true;
    end
end

if ~dataFilesExist
    disp("Generating data and saving in data files...")
    [success, msg, msgID] = mkdir(dataDirectory);
    if ~success
        error(msgID, msg)
    end
    for modType = 1:numModulationTypes
        fprintf('%s - Generating %s frames\n',
            datestr(toc/86400, 'HH:MM:SS'), modulationTypes(modType))

        label = modulationTypes(modType);
        numSymbols = (framesPerModType / sps);
        dataSrc = helperModClassGetSource(modulationTypes(modType), sps,
            2*spf, fs);
        modulator = helperModClassGetModulator(modulationTypes(modType),
            sps, fs);
        if contains(char(modulationTypes(modType)), {'B-FM', 'DSB-
AM', 'SSB-AM'})
            % Analog modulation types use a center frequency of 100 MHz
            channel.CenterFrequency = 100e6;
        else
            % Digital modulation types use a center frequency of 902 MHz
            channel.CenterFrequency = 902e6;
        end

        for p=1:framesPerModType

            x = dataSrc();    % Generate random data
            y = modulator(x); % Modulate
            rxSamples = testChannel(y);    % Pass through independent
channels

            % Remove transients from the beginning, trim to size, and
normalize
            frame = helperModClassFrameGenerator(rxSamples, spf, spf,
transDelay, sps);

            % Save data file
            fileName = fullfile(dataDirectory, ...

```

```
        sprintf("%s%s%03d", fileNameRoot, modulationTypes(modType), p));  
        save(fileName, "frame", "label")  
    end  
end  
else  
    disp("Data files exist. Skip data generation.")  
end  
  
helperModClassPlotTimeDomain(dataDirectory, modulationTypes, fs)  
helperModClassPlotSpectrogram(dataDirectory, modulationTypes, fs, sps)
```

Appendix D: MATLAB Script for Storing Data MAT Files into Computer Memory

```
% Storing data and importing into memory

frameDS =
signalDatastore(dataDirectory, 'SignalVariableNames', ["frame", "label"]
);

frameDSTrans = transform(frameDS, @helperModClassIQAsPages);

splitPercentages = [trainingSamples, validationSamples, testSamples];
[trainDSTrans, validDSTrans, testDSTrans] =
helperModClassSplitData(frameDSTrans, splitPercentages);

% Gather the training and validation frames into the memory
trainFramesTall = tall(transform(trainDSTrans,
@helperModClassReadFrame));
rxTrainFrames = gather(trainFramesTall);
rxTrainFrames = cat(4, rxTrainFrames{:});
validFramesTall = tall(transform(validDSTrans,
@helperModClassReadFrame));
rxValidFrames = gather(validFramesTall);
rxValidFrames = cat(4, rxValidFrames{:});

% Gather the training and validation labels into the memory
trainLabelsTall = tall(transform(trainDSTrans,
@helperModClassReadLabel));
rxTrainLabels = gather(trainLabelsTall);
validLabelsTall = tall(transform(validDSTrans,
@helperModClassReadLabel));
rxValidLabels = gather(validLabelsTall);
```

Appendix E: MATLAB Script for Designing and Training CNN

```
% Train CNN

numModTypes = numel(modulationTypes);
netWidth = 1;
filterSize = [1 sps];
poolSize = [1 2];
modClassNet = [
    imageInputLayer([2 spf 1], 'Normalization', 'none', 'Name', 'Input
Layer')

    convolution2dLayer(filterSize, 16*netWidth, 'Padding', 'same',
'Name', 'CNN1')
    batchNormalizationLayer('Name', 'BN1')
    reluLayer('Name', 'ReLU1')
    maxPooling2dLayer(poolSize, 'Stride', [1 2], 'Name', 'MaxPool1')

    convolution2dLayer(filterSize, 24*netWidth, 'Padding', 'same',
'Name', 'CNN2')
    batchNormalizationLayer('Name', 'BN2')
    reluLayer('Name', 'ReLU2')
    maxPooling2dLayer(poolSize, 'Stride', [1 2], 'Name', 'MaxPool2')

    convolution2dLayer(filterSize, 32*netWidth, 'Padding', 'same',
'Name', 'CNN3')
    batchNormalizationLayer('Name', 'BN3')
    reluLayer('Name', 'ReLU3')
    maxPooling2dLayer(poolSize, 'Stride', [1 2], 'Name', 'MaxPool3')

    convolution2dLayer(filterSize, 48*netWidth, 'Padding', 'same',
'Name', 'CNN4')
    batchNormalizationLayer('Name', 'BN4')
    reluLayer('Name', 'ReLU4')
    maxPooling2dLayer(poolSize, 'Stride', [1 2], 'Name', 'MaxPool4')

    convolution2dLayer(filterSize, 64*netWidth, 'Padding', 'same',
'Name', 'CNN5')
    batchNormalizationLayer('Name', 'BN5')
    reluLayer('Name', 'ReLU5')
    maxPooling2dLayer(poolSize, 'Stride', [1 2], 'Name', 'MaxPool5')

    convolution2dLayer(filterSize, 96*netWidth, 'Padding', 'same',
'Name', 'CNN6')
    batchNormalizationLayer('Name', 'BN6')
    reluLayer('Name', 'ReLU6')

    averagePooling2dLayer([1 ceil(spf/32)], 'Name', 'AP1')

    fullyConnectedLayer(numModTypes, 'Name', 'FC1')
    softmaxLayer('Name', 'SoftMax')

    classificationLayer('Name', 'Output') ];
```

```

% Training Options
maxEpochs = 12;
miniBatchSize = 256;

validationFrequency = floor(numel(rxTrainLabels)/miniBatchSize);
options = trainingOptions('sgdm', 'InitialLearnRate', 2e-2, ...
    'MaxEpochs', maxEpochs, 'MiniBatchSize', miniBatchSize, ...
    'shuffle', 'every-epoch', 'Plots', 'training-progress', ...
    'Verbose', false, 'ValidationData', {rxValidFrames,
rxValidLabels}, ...
    'ValidationFrequency', validationFrequency, 'LearnRateSchedule',
...
    'piecewise', 'LearnRateDropPeriod', 9, 'LearnRateDropFactor',
0.1, ...
    'ExecutionEnvironment', 'cpu');

if trainNow == true
    fprintf('%s - Training the network\n',
datestr(toc/86400, 'HH:MM:SS'))
    trainedNet =
trainNetwork(rxTrainFrames, rxTrainLabels, modClassNet, options);
else
    load trainedModulationClassificationNetwork
end

fprintf('%s - Classifying test frames\n',
datestr(toc/86400, 'HH:MM:SS'))
% Gather the test frames into the memory
testFramesTall = tall(transform(testDSTrans,
@helperModClassReadFrame));
rxTestFrames = gather(testFramesTall);
rxTestFrames = cat(4, rxTestFrames{:});

% Gather the test labels into the memory
testLabelsTall = tall(transform(testDSTrans,
@helperModClassReadLabel));
rxTestLabels = gather(testLabelsTall);

rxTestPred = classify(trainedNet, rxTestFrames);
testAccuracy = mean(rxTestPred == rxTestLabels);
disp("Test accuracy: " + testAccuracy*100 + "%")

figure
cm = confusionchart(rxTestLabels, rxTestPred);
cm.Title = 'Confusion Matrix for Test Data';
cm.RowSummary = 'row-normalized';
cm.Parent.Position = [cm.Parent.Position(1:2) 740 424];

```

Appendix F: MATLAB Function Script for Testing Over-the-air Signals

```
function testAcc = SDRTest(radios)

modulationTypes = categorical(["BPSK", "QPSK", "8PSK", ...
    "16QAM", "64QAM", "PAM4", "GFSK", "CPFSK", "B-FM", ...
    "DSB-AM", "SSB-AM"]);
load trainedModulationClassificationNetwork
trainedNet

framesPerModType = 100;
sps = 8;
spf = 1024;
fs = 200e3;

%Transmitter
txPluto = sdrtx('Pluto');
txPluto.RadioID = 'usb:0';
txPluto.CenterFrequency = 902e6;
txPluto.BasebandSampleRate = 200e3;

%Receiver
rxPluto = sdrrx('Pluto');
rxPluto.RadioID = 'usb:1';
rxPluto.BasebandSampleRate = 200e3;
rxPluto.ShowAdvancedProperties = true;
rxPluto.EnableQuadratureCorrection = false;
rxPluto.GainSource = "Manual";
rxRadio.SamplesPerFrame = 1024;
rxRadio.EnableBurstMode = true;
rxRadio.NumFramesInBurst = 1;
rxRadio.OutputDataType = 'single';
maximumGain = 73;
minimumGain = -10;

rng(12345)
tic

numModTypes = length(modulationTypes);
txModType = repmat(modulationTypes(1), numModTypes*framesPerModType,
1);
estModType = repmat(modulationTypes(1), numModTypes*framesPerModType,
1);
frameCount = 1;

for modType = 1:numModTypes
    fprintf('%s - Testing %s Frames \n', datestr(toc/86400,
'HH:MM:SS'), modulationTypes(modType))
    dataSrc =
helperModClassGetSource(modulationTypes(modType), sps, 2*spf, fs);
    modulator =
helperModClassGetModulator(modulationTypes(modType), sps, fs);
```



```

        if contains(char(modulationTypes(modType)), {'B-FM'}) &&
(radioPlatform == "PlutoSDR")
            txRadio.CenterFrequency = 370e6;
            rxRadio.CenterFrequency = 370e6;
        else
            txRadio.CenterFrequency = 902e6;
            rxRadio.CenterFrequency = 902e6;
        end

disp('Starting Transmitter')
x = dataSrc();
y = modulator(x);
y = y(4*sps+1:end,1);
maxVal=max(max(abs(real(y))), max(abs(imag(y))));
y = y*0.8/maxVal;
transmitRepeat(txRadio, complex(y));

disp('Adjusting Receiver Gain')
rxRadio.Gain = maximumGain;
gainAdjusted = false;
while ~gainAdjusted

    for p=1:20
        rx=rxRadio();
    end

    maxAmplitude=max([abs(real(rx)); abs(imag(rx))]);
    if (maxAmplitude > 0.8) || (rxRadio.Gain <= minimumGain)
        gainAdjusted = true;
    else
        rxRadio.Gain = rxRadio.Gain - 3;
    end
end

disp('Starting Receiver and Classification')
for p=1:framesPerModType
    rx = rxRadio();

    frameEnergy = sum(abs(rx).^2);
    rx = rx/sqrt(frameEnergy);
    reshapedRx(1,:,1,1) = real(rx);
    reshapedRx(1,:,2,1) = imag(rx);

    txModType(frameCount) = modulationTypes(modType);
    estModType(frameCount) = classify(trainedNet, reshapedRx);

    frameCount = frameCount + 1;
    pause(0.1)
end

disp('Releasing Rx radio')
release(txRadio);

```

```

testAcc = mean(txModType(1:frameCount-1) == estModType(1:frameCount-1));
disp("Test Accuracy: " + testAcc*100 + "%")
end
release(rxRadio);
testAcc = mean(txModType == estModType);
disp("Final Test Accuracy: " + testAcc + "%")

figure
cm = confusionchart(txModType, estModType);
cm.Title = 'Confusion Matrix for Test Data';
cm.RowSummary = 'row-normalized';
cm.Parent.Position = [cm.Parent.Position(1:2) 740 424];
end

```

Appendix G: MATLAB Script Testing Radio Connection and Calling the Testing Function

```
if helperIsPlutoSDRInstalled() == true
    radios = findPlutoRadio();
    if length(radios) >= 2
        SDRTest(radios);
    else
        disp('ADALM Pluto Radios Not Found')
    end
end
```

Appendix H: Full Table of Distance vs Accuracy Results

| Distance (m) | 0.25 | | | 0.5 | | | 0.75 | | | 1 | | | 1.25 | | | 1.5 | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| Modulation Type | | | | | | | | | | | | | | | | | | |
| BPSK | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 |
| QPSK | 93.50 | 95.50 | 96.00 | 96.25 | 91.00 | 91.00 | 88.50 | 94.00 | 94.50 | 85.50 | 86.50 | 89.00 | 95.50 | 85.50 | 91.00 | 89.50 | 93.00 | 90.50 |
| 8PSK | 95.67 | 96.67 | 97.30 | 95.30 | 93.00 | 92.67 | 91.67 | 95.00 | 96.00 | 88.67 | 89.00 | 91.00 | 95.67 | 87.30 | 93.00 | 92.00 | 93.00 | 93.00 |
| 16QAM | 96.50 | 97.50 | 96.25 | 96.00 | 94.00 | 93.75 | 90.75 | 95.00 | 92.75 | 90.25 | 90.00 | 88.50 | 94.75 | 90.25 | 94.25 | 93.50 | 93.00 | 93.75 |
| 64QAM | 95.00 | 97.00 | 95.40 | 94.20 | 91.40 | 90.80 | 91.40 | 93.00 | 92.40 | 90.20 | 90.20 | 89.40 | 93.80 | 88.80 | 92.40 | 91.40 | 92.40 | 92.80 |
| PAM4 | 95.83 | 97.50 | 96.17 | 95.17 | 92.83 | 92.30 | 92.83 | 93.86 | 92.43 | 91.83 | 91.83 | 91.67 | 94.83 | 90.67 | 93.67 | 92.83 | 93.67 | 94.00 |
| GFSK | 96.43 | 97.86 | 96.71 | 95.86 | 93.86 | 92.43 | 93.86 | 95.00 | 94.57 | 93.00 | 93.00 | 92.43 | 95.52 | 92.00 | 94.57 | 93.86 | 94.52 | 94.86 |
| CPFSK | 96.88 | 98.13 | 97.13 | 96.38 | 94.63 | 94.25 | 94.63 | 95.63 | 95.25 | 93.88 | 93.88 | 93.75 | 96.13 | 93.00 | 95.25 | 94.63 | 95.25 | 95.50 |
| B-FM | 87.11 | 95.40 | 94.73 | 94.30 | 95.60 | 91.20 | 92.00 | 93.50 | 96.75 | 94.00 | 91.00 | 92.40 | 95.14 | 93.84 | 94.80 | 94.21 | 95.72 | 96.54 |
| Overall | 95.21 | 97.28 | 96.63 | 95.94 | 94.03 | 93.16 | 92.85 | 95.03 | 95.10 | 91.93 | 91.71 | 92.02 | 95.70 | 91.26 | 94.33 | 93.55 | 94.51 | 94.55 |
| Average | 96.38 | | | 94.38 | | | 94.33 | | | 91.88 | | | 93.76 | | | 94.20 | | |

| Distance (m) | 1.75 | | | 2 | | | 2.25 | | | 2.5 | | |
|-----------------|--------|--------|-------|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| Modulation Type | | | | | | | | | | | | |
| BPSK | 100.00 | 100.00 | 99.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 |
| QPSK | 91.50 | 93.00 | 90.00 | 91.00 | 90.00 | 94.50 | 94.50 | 90.50 | 95.50 | 93.00 | 91.00 | 95.50 |
| 8PSK | 94.00 | 95.00 | 93.00 | 93.67 | 93.00 | 96.30 | 96.00 | 93.30 | 96.00 | 94.30 | 94.00 | 96.00 |
| 16QAM | 95.50 | 93.25 | 71.75 | 90.75 | 93.25 | 97.00 | 77.25 | 94.25 | 97.00 | 91.00 | 95.25 | 96.75 |
| 64QAM | 92.40 | 93.20 | 75.00 | 90.80 | 94.40 | 96.00 | 80.40 | 94.80 | 97.20 | 91.40 | 92.60 | 93.80 |
| PAM4 | 93.67 | 94.30 | 79.17 | 92.30 | 95.30 | 96.67 | 83.67 | 95.67 | 97.67 | 92.83 | 93.83 | 94.83 |
| GFSK | 94.57 | 95.14 | 82.14 | 93.43 | 96.00 | 97.14 | 86.00 | 96.29 | 98.00 | 93.86 | 94.71 | 95.57 |
| CPFSK | 95.25 | 95.63 | 84.38 | 94.25 | 96.50 | 97.50 | 87.75 | 96.75 | 98.25 | 94.63 | 95.38 | 96.13 |
| B-FM | 95.64 | 94.73 | 85.13 | 95.43 | 96.85 | 97.80 | 88.23 | 97.00 | 98.74 | 95.20 | 95.89 | 96.56 |
| Overall | 94.73 | 94.92 | 84.40 | 93.51 | 95.03 | 96.99 | 88.20 | 95.40 | 97.60 | 94.02 | 94.74 | 96.13 |
| Average | 91.35 | | | 95.18 | | | 93.73 | | | 94.96 | | |

Appendix I: MATLAB Script for Plotting Graph of Test Accuracy Against Distance Between Radios

```
%Plot graph of the average overall accuracy against the distance  
between two radios
```

```
distance = [0.25 0.5 0.75 1 1.25 1.5 1.75 2 2.25 2.5];  
avgOverallAcc = [96.38 94.38 94.33 91.88 93.76 94.20 91.35 95.18  
93.73 94.96];
```

```
plot(distance, avgOverallAcc);  
grid;  
xlabel('Distance (m)');  
ylabel('Classification Accuracy');  
title('Average Overall Accuracy vs Distance');  
xlim([0.25 2.5]);
```

Appendix J: MATLAB Script for Plotting Graph of Test Accuracy Against Distance for Each Modulation Type

```
%Plot Graph of average accuracy for each modulation type
against distance

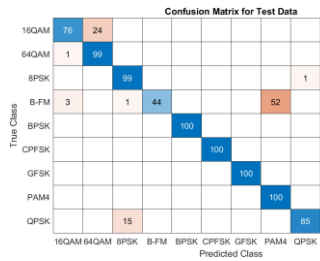
distance = [0.25 0.5 0.75 1 1.25 1.5 1.75 2 2.25 2.5];
bpskAcc = [100.00 100.00 100.00 100.00 100.00 100.00 100.00 99.67
100.00 100.00 100.00];
qpskAcc = [95.00 92.75 92.30 87.00 90.60 91.00 91.50 91.77
93.50 93.16];
acc_8psk = [96.55, 93.66, 94.22, 89.56, 91.99, 92.67,
94.00, 94.32, 95.10, 94.76];
acc_16qam = [96.75 94.58 92.83 89.58 93.08 93.42 86.83
93.60 89.50 94.30];
acc_64qam = [95.80 92.13 92.26 89.93 91.60 92.20 86.87
93.73 90.80 92.60];
pam4Acc = [96.50 93.43 93.50 91.78 93.05 93.50 89.05 94.76
92.34 93.83];
gfskAcc = [97.01 94.05 94.48 92.81 94.01 94.40 90.62 95.52
93.43 94.71];
cpfskAcc = [97.38 95.08 95.16 93.83 94.79 95.13 91.75
96.08 94.25 95.38];
bfmAcc = [92.41 93.70 94.08 94.08 94.59 95.49 91.83 96.69
94.66 95.88];

plot(distance, bpskAcc, 'k-.', distance, qpskAcc, 'k-',
distance, acc_8psk, 'g-', distance, acc_16qam, 'r-
',distance, acc_64qam, 'b-', distance, pam4Acc, 'c-
',distance, gfskAcc, 'm-', distance, cpfskAcc, 'y-',
distance, bfmAcc, 'k--');
grid;
xlabel('Distance (m)');
ylabel('Classification Accuracy');

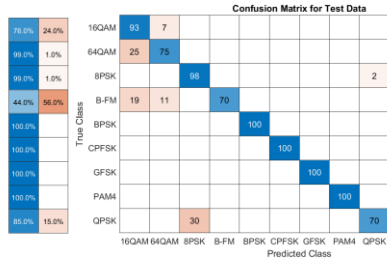
title('Average Accuracy of Each Modulation Type vs
Distance');
xlim([0.25 2.5]);
ylim([86 102]);
```


Appendix K: Confusion Matrices generated from obstacle experiments

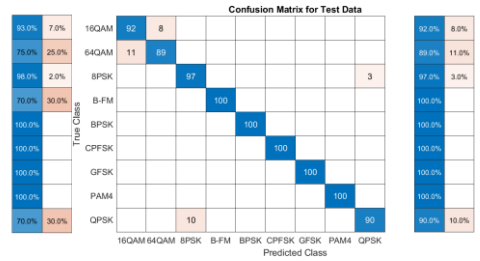
Metal Test 1:



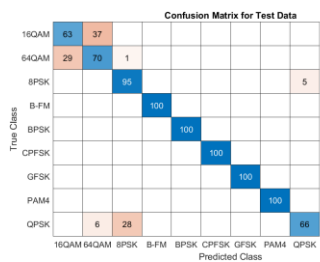
Metal Test 2:



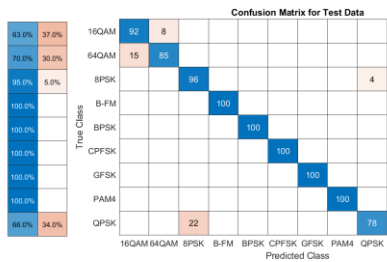
Metal Test 3:



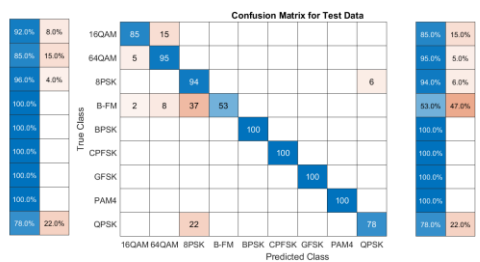
Wood Test 1:



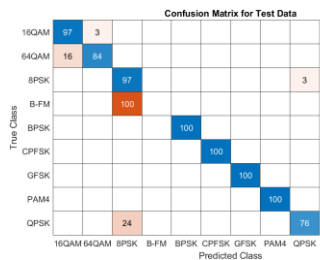
Wood Test 2:



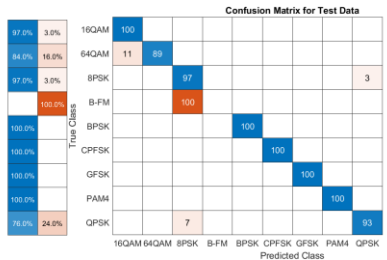
Wood Test 3:



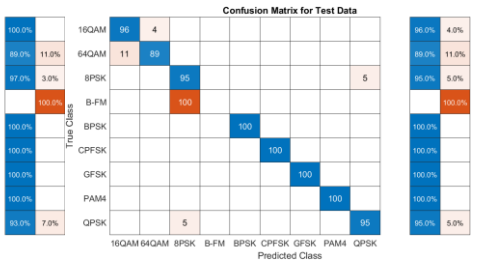
Glass Test 1:



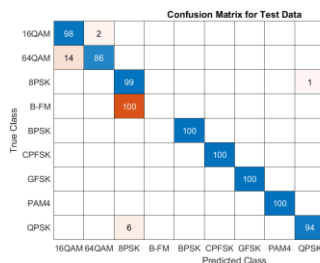
Glass Test 2:



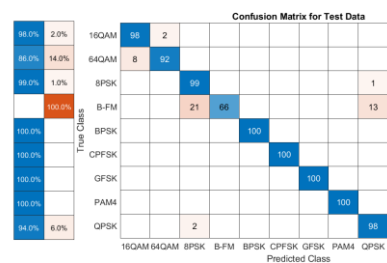
Glass Test 3:



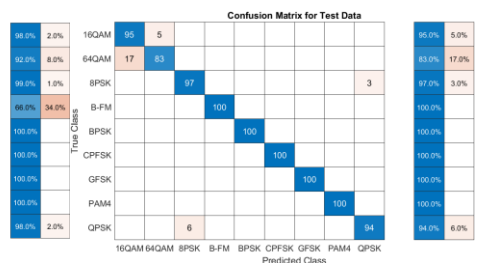
Plastic Test 1:



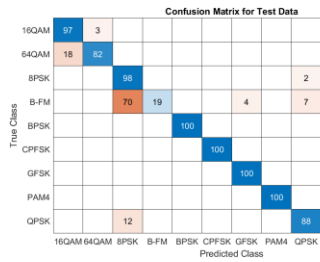
Plastic Test 2:



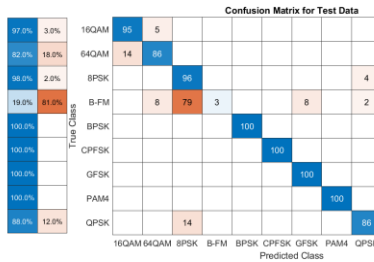
Plastic Test 3:



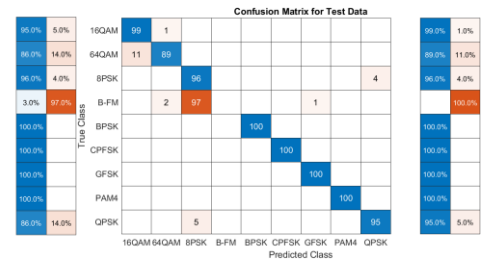
Ceramic Test 1:



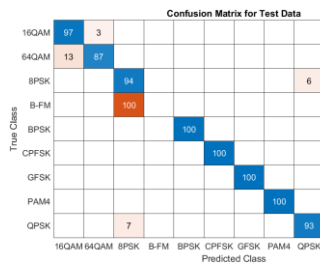
Ceramic Test 2:



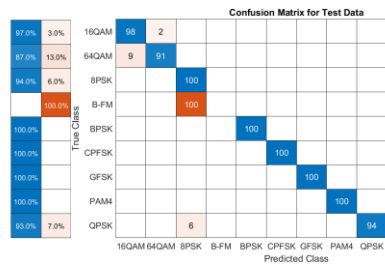
Ceramic Test 3:



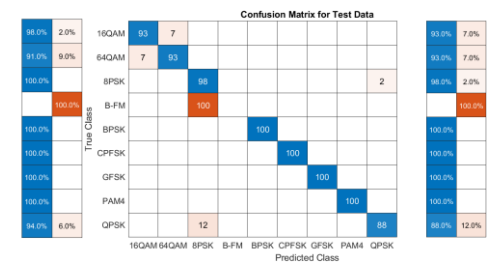
Cardboard Test 1:



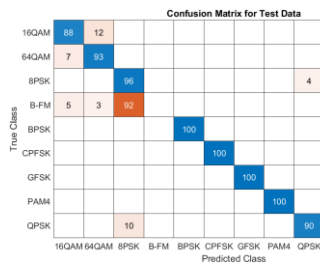
Cardboard Test 2:



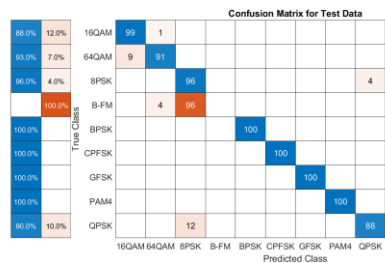
Cardboard Test 3:



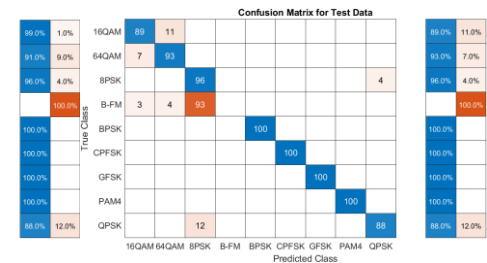
Between Wall Test 1:



Between Wall Test 2:



Between Wall Test 3:



Appendix L: Full Table of Material vs Accuracy Results

| Material | Metal | | | Wood | | | Glass | | | Plastic | | |
|-----------------|--------|--------|--------|--------|--------|--------|--------|--------|--------|---------|--------|--------|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| Modulation Type | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| BPSK | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 |
| QPSK | 92.50 | 85.00 | 95.00 | 83.00 | 89.00 | 89.00 | 88.00 | 96.50 | 97.50 | 97.00 | 99.00 | 97.00 |
| 8PSK | 94.67 | 89.30 | 95.67 | 87.00 | 91.33 | 90.67 | 91.00 | 96.67 | 96.67 | 97.67 | 99.00 | 97.00 |
| 16QAM | 90.00 | 90.25 | 94.75 | 81.00 | 91.50 | 89.25 | 92.50 | 97.50 | 96.50 | 97.75 | 98.75 | 96.50 |
| 64QAM | 91.80 | 87.20 | 93.60 | 78.80 | 90.20 | 90.40 | 90.80 | 95.80 | 95.00 | 95.40 | 97.40 | 93.80 |
| PAM4 | 93.17 | 89.30 | 94.67 | 82.33 | 91.83 | 92.00 | 92.30 | 96.50 | 95.83 | 96.17 | 97.83 | 94.83 |
| GFSK | 94.14 | 90.86 | 95.43 | 84.57 | 93.00 | 93.14 | 93.43 | 97.00 | 96.43 | 96.71 | 98.14 | 95.57 |
| CPFSK | 94.88 | 92.00 | 96.00 | 86.75 | 93.88 | 94.00 | 94.25 | 97.38 | 96.88 | 97.13 | 98.38 | 96.13 |
| B-FM | 89.22 | 89.56 | 96.40 | 88.22 | 94.56 | 89.44 | 83.78 | 86.56 | 86.11 | 86.33 | 94.78 | 96.56 |
| Overall | 93.38 | 90.39 | 95.72 | 85.74 | 92.81 | 91.99 | 91.78 | 95.99 | 95.66 | 96.02 | 98.14 | 96.38 |
| Average | 93.16 | | | 90.18 | | | 94.48 | | | 96.85 | | |

| Material | Ceramic | | | Cardboard | | | Between Wall | | |
|-----------------|---------|--------|--------|-----------|--------|--------|--------------|--------|--------|
| | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| Modulation Type | 1 | 2 | 3 | 1 | 2 | 3 | 1 | 2 | 3 |
| BPSK | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 | 100.00 |
| QPSK | 94.00 | 93.00 | 97.50 | 96.50 | 97.00 | 94.00 | 95.00 | 94.00 | 94.00 |
| 8PSK | 95.33 | 94.00 | 97.00 | 95.67 | 98.00 | 95.33 | 95.33 | 94.67 | 94.67 |
| 16QAM | 95.75 | 94.25 | 97.50 | 96.00 | 98.00 | 94.75 | 93.50 | 95.75 | 93.25 |
| 64QAM | 93.00 | 92.60 | 95.80 | 94.20 | 96.60 | 94.40 | 93.40 | 94.80 | 93.20 |
| PAM4 | 94.17 | 93.83 | 96.50 | 95.17 | 97.17 | 95.33 | 94.50 | 95.67 | 94.33 |
| GFSK | 95.00 | 94.71 | 97.00 | 95.85 | 97.57 | 96.00 | 95.29 | 96.29 | 95.14 |
| CPFSK | 95.63 | 95.38 | 97.38 | 96.38 | 97.88 | 96.50 | 95.88 | 96.75 | 95.75 |
| B-FM | 87.11 | 85.11 | 86.56 | 85.67 | 87.00 | 85.78 | 85.22 | 86.00 | 85.11 |
| Overall | 94.44 | 93.65 | 96.14 | 95.05 | 96.58 | 94.68 | 94.24 | 94.88 | 93.94 |
| Average | 94.74 | | | 95.44 | | | 94.35 | | |