TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

AUTOMATIC MODULATION CLASSIFICATION (AMC)

By:

Aasman Bashyal (074BEX402)

Asim Maharjan (074BEX408)

Basanta Rijal (074BEX409)

Saju Khakurel (074BEX438)

THIS PROJECT WAS SUBMITTED TO THE DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENT FOR THE BACHELOR'S DEGREE IN ELECTRONICS AND COMMUNICATION ENGINEERING

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL

April, 2022

TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

AUTOMATIC MODULATION CLASSIFICATION (AMC)

By:

Aasman Bashyal (074BEX402)

Asim Maharjan (074BEX408)

Basanta Rijal (074BEX409)

Saju Khakurel (074BEX438)

DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
LALITPUR, NEPAL

April, 2022

LETTER OF APPROVAL

The undersigned certify that they have read, and recommended to the Institute of Engineering for acceptance, a project report entitled **AUTOMATIC MODULATION CLASSIFICATION (AMC)** submitted by **Aasman Bashyal**, **Asim Maharjan**, **Basanta Rijal** and **Saju Khakurel** in partial fulfillment of the requirements for the Bachelor's Degree in Electronics and Communication Engineering.

Supervisor: **Prof. Dr. Shashidhar Ram Joshi, Dean**
Institute of Engineering, Pulchowk Campus

Internal Examiner: **Sharad Kumar Ghimire, Associate Professor**
Department of Electronics and Computer Engineering
Institute of Engineering, Pulchowk Campus

External Examiner: **Er. Bishnu Ram Neupane, Deputy General Manager**
Nepal Television Corporation
Singha Durbar, Kathmandu

DATE OF APPROVAL _____

COPYRIGHT

ACKNOWLEDGEMENT

We would like to express our deepest gratitude towards our supervisor Prof. Dr. Shashidhar Ram Joshi, for providing his valuable guidance, supervision and encouragement throughout this project. We are really fortunate to have the kind association as well as supervision and encouragement during the project. We are obliged to work under his counseling and even our most profound regards are not enough. We would also like to pay our deep sense of gratitude to the Department of Electronics and Communication for incorporating major project as part of our syllabus for realization of our knowledge in the real world application and giving us a great opportunity like this to carry out the project that will help us shape our career.

And last but not the least, we would like to thank our family and friends who have directly and indirectly supported us throughout this project.

Authors:

Aasman Bashyal

Asim Maharjan

Basanta Rijal

Saju Khakurel

ABSTRACT

Automatic Modulation Classification (AMC) is the classification of signals automatically based on the type of modulation used to generate that signal. Many different approaches for AMC have been proposed which are categorized as 'Decision Theoretic Approach', 'Feature Based Approach' and 'Deep Learning Based Approach'. Here, we have implemented various types of classifiers for AMC which include likelihood based approach, deep learning based approaches (LSTM and BiLSTM networks) and also using a Quantum Neural Network (QNN). The likelihood based classifier and the QNN are restricted to classify only two types of modulations, namely BPSK and QPSK due to various limitation of those approaches. The deep learning based classifiers can classify 11 different modualtion types from their IQ samples in RadioML2016.10A dataset.

Very poor classification accuracy is seen at lower SNR levels for all the implemented models. The unrealistic requirement of perfect Channel State Information(CSI) in likelihood based classifiers and the lack of computational power at present in QNN architectures make them infeasible for real life use cases. The overall accuracy of 37.33%, 46.68%, 51.31%, 54.84% and 56.88% for LSTM, LSTM with attention, RadioML, Modified CNN, and the BiLSTM models respectively were observed. Among all these DL based models, BiLSTM with attention layer achieved the highest accuracy of 84.48% (14dB) while retaining a comparable average prediction time of 0.041 ms for a signal.

Only RNN and CNN based DL architectures were considered in this report for their performance evaluation. Other architectures like vision transformers, encoder-decoder have shown promising improvement in the classification accuracy. Implementation and comparison of such architectures can be one of the next steps. Furthermore, only number of layers and number of nodes were considered for hyper-parameter tuning. More refined hyper-parameter tuning can also be one of the next steps.

Keywords: Automatic Modulation Classification, Adaptive Modulation Scheme, Cognitive Radio, Quantum Neural Network, RadioML

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

ALRT Average Likelihood Ratio Test.

AM Amplitude Modulation.

AM-DSB Amplitude Modulated Double Side-band.

AM-SSB Amplitude Modulated Single Side-band.

AMC Automatic modulation classification.

AWGN Additive White Gaussian Noise.

Bi-LSTM Bidirectional Long Short Term Memory.

BPSK Binary Phase Shift Keying.

CNN Convolutional Neural Network.

CPFSK Continous Phase Frequency Shift Keying.

CSI Chanel State Information.

DL Deep Learning.

DNN Deep Neural Network.

DSB Double Side-band.

FB Feature Based.

FM Frequency Modulation.

GFSK Gaussian Frequency Shift Keying.

IoT Internet of Things.

IQ In-phase and Quadrature.

LA Link Adaptation.

LB Likelihood Based.

LPF Low Pass Filter.

LSTM Long Short Term Memory.

NLP Natural Language Processing.

PAM Pulse Amplitude Modulation.

PQC Parameterized Quantum Circuit.

PSK Phase Shift Keying.

QAM Qudrature Amplitude Modulation.

QNN Quantum Neural Network.

QPSK Qudrature Phase Shift Keying.

ReLU Rectified Linear Unit.

RNN Recurrent Neural Network.

SDR Software-Defined Radio.

SNR Signal to Noise Ratio.

TFQ Tensorflow Quantum.

TFR Time Frequency Representation.

WBFM Wideband Frequency Modulation.

1 INTRODUCTION

1.1. Background

Automatic modulation classification (AMC) in the adaptive modulation system serves the purpose of identifying the modulation scheme mostly according to the channel condition and the other parameters in the channel for the efficient transmission of the signal. The modulation classification problem arose in military applications for identifying adversary transmitting units, preparing jamming signals, and recovering information from the intercepted signals. Before the advent of AMC, the classification was done manually which required domain experts to analyze the received signal. Later with the development in electronics and considerable research works in the field of communications, the manual modulation is being replaced by automatic modulation classification which has now opened it to many other applications. One of which was the Link Adaptation (LA) system which introduced AMC to civilian use cases. LA creates an adaptive modulation scheme given multiple modulation techniques and optimizes the transmission reliability and data rate through the adaptive selection of modulation schemes according to channel conditions. The transmitter changing the modulation, required the receiver to identify the current modulation scheme to effectively get the information. AMC solved this problem meanwhile saving the precious bandwidth in the wireless system by omitting the overhead of transmission of extra modulation information.

In recent years, the field of IoT, smart technology, automated systems, efficient wireless communication and many more are highly evolving which results in a high requirement of advanced communication systems. This demands efficient allocation of network resources using ideas like LA systems, intelligent radio systems including cognitive radio and Software-Defined Radio (SDR); essentially making AMC the Swiss Army Knife of the communication systems in near future.

AMC problem has been approached using various techniques like Likelihood based classifiers and Feature based ones. In recent years, the use of deep learning has gained popularity to solve the problem of AMC, which facilitates the classification process without any manual feature extraction process.

Deep learning techniques, which can automatically extract features, have been looked at as probable approaches in the view of these considerations. AMC has been using these

techniques to automatically select the modulation classification and also to extract the features for further processing for demodulation. To be more specific, Deep neural networks (DNN) like Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN), having multiple layers proved to be efficient in feature extraction. This improvement in performance with DNN, techniques and the growing requirement of robust and highly efficient communication systems is making a high demand in research of AMC.

1.2. Objectives

The general aim of this project is to explore and develop Automatic Modulation Classification approaches and evaluate their performances.The specific objectives are as follows:

1. To evaluate the various approaches towards the AMC problem.
2. To design and implement an Automatic Modulation Classifier.
3. To compare the performances of these implemented approaches.

1.3. Problem Statement

The requirement for a highly efficient, robust, and lightweight Automatic Modulation Classifier has resulted in various approaches to the AMC problem. These various approaches to this problem pose different sorts of benefits and pose various trade-offs. Therefore, in this project, we aim to explore, implement and compare different approaches to this AMC problem.

2 LITERATURE REVIEW

The prospect of increased usage of AMC in the near future has attracted many researchers to this problem. Accuracy, Robustness, Computational Efficiency and Versatility are said to be the essential characteristics of a successful design of AMC by Zhu and Nandi in [1].

Even though AMC is a multidimensional problem, the approaches towards it can be broadly classified into 2 groups; namely - Likelihood Based (LB) Classification and Feature Based (FB) Classification [2]. In case of LB classifiers, classification is done by a likelihood function operating on the received signals. They were first introduced by Polydoros and Kim in [3] to provide the optimal solution to the modulation-classification problem given a good enough signal model and perfect Channel State Information (CSI). Over the years, these classifiers went through different modifications and specifications but still suffer from large computational time.

In contrast to LB classifiers, FB classifiers provide the benefit of reduced time complexity. The FB approach introduces two significant steps: feature extraction and classification. Use of machine learning methodologies for the classification problem has been exploited by many researchers using approaches like K-Nearest Neighbors, Support Vector Machines and Principal Component Analysis [4]. The problem such approaches face is that the significant features are often left out including some time and frequency domain information during feature extraction [5].

Recent approaches to neural networks involve the use of deep learning to learn the features of signals to achieve better accuracy. Deep learning based techniques handle the classification along with the feature extraction by themselves which has led to a significant increase in the study of application of such approaches.

Authors of [6] have proposed and implemented a Convolutional Neural Network architecture for modulation classification. They also compare the the CNN based architecture with various other feature based methods and point out that deep learning based approaches are also a viable approach for modulation classification.

A comprehensive study of various deep learning based approaches to the AMC problem can be found in [7]. The authors provide a detailed description of various approaches to AMC along with their model architectures, reported accuracies and the current trend in ap-

proaches to tackling the AMC problem. Many approaches using simple feed forwards neural networks, convolutional neural networks and recurrent neural networks are highlighted by the authors. Most of the classifiers described were able to achieve high accuracies ($> 90\%$) at certain signal to noise ratios.

A LSTM based network with temporal attention for the AMC problem is described in [8] where the authors classify signals under various noise conditions with Rayleigh fading channels. The performance of the proposed classifier was comparable to that of LB classifiers with perfect channel knowledge.

Authors of [9] have prepared a dataset for such modulation classification problem. The dataset, named RadioML, was prepared in GNU Radio software, under various channel conditions for a wide variety of modulation types. They have also highlighted the differences in radio signal processing from other machine learning domains, namely the well structured but very noisy data.

Quantum computers have been gaining popularity in the recent years, mainly due to Shor's Algorithm [10] for factoring numbers and Grover's algorithm for database searching [11]. Similarly, the possibilities of machine learning on quantum computers are also being explored in the present. Various approaches for implementing quantum machine learning are described in [12] and [13]. Authors in [14] have developed a software framework for quantum machine learning which allows the design of hybrid quantum-classical neural networks using Tensorflow. Furthermore, authors in [15] have proposed a way of performing classification on such quantum neural networks. The authors also give an example of performing classification of handwritten digits using the quantum neural network.

3 THEORETICAL BACKGROUND

3.1. In-phase and Quadrature Signals

IQ signals stand for In-phase and Quadrature signal representations of a single signal as two sinusoidal signals with the same frequency and a relative phase shift of 90 degrees. By convention, the I signal is a cosine waveform, and the Q signal is a sine waveform. Any form of modulation can be performed simply by varying the amplitude of I and Q signals and then adding them together. This representation alone does not possess any significance but together can modulate the carrier signals, especially in the quadrature modulation. Quadrature modulation refers to modulation that involves I/Q signals. Quadrature phase-shift keying can be obtained by adding I and Q carriers that have been individually multiplied, following the incoming digital data, by +1 or –1.

3.2. Digital Modulation

In digital communication, the signals are converted into suitable form before transmission where certain parameters like amplitude, phase, and so on are changed as per requirement along with the insertion of additional data like synchronization bit, error correction bits, delay, and so on. Modulation plays an important role in communication since proper modulation is the key to efficient transmission. Modulation can be classified into various types as per the nature of the carrier and message signal. Classification of modulation is shown in Figure 3.1.

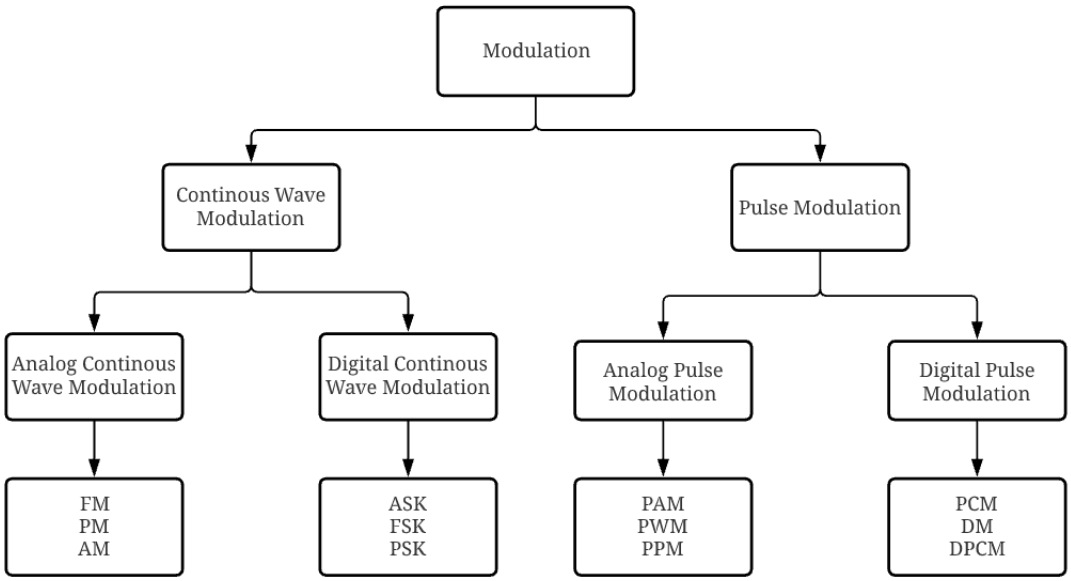


Figure 3.1: Modulation Types

3.2.1. Constellation Diagrams

The graphical representation of a signal modulated by a different digital modulation scheme, such as ASK, PSK, BPSK, QAM is known as a constellation diagram. It is a two-dimensional scatter plot in a complex plane where I represents the real axis of the complex plane and Q represents the imaginary axis. It represents the possible symbols that may be selected by a given modulation scheme as points in the complex plane.

The angle of a point, measured counterclockwise from the horizontal axis, represents the phase shift of the carrier wave from a reference phase. The distance of a point from the origin represents a measure of the amplitude or power of the signal. The constellation plot for 8-ary PSK is shown in Figure 3.2.

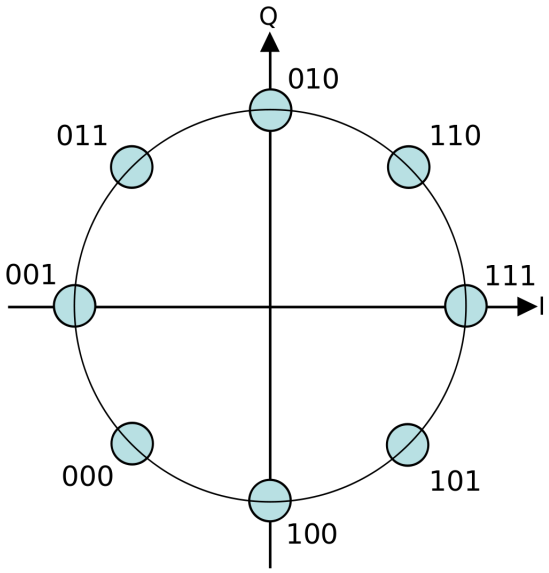


Figure 3.2: Constellation plot for 8-ary PSK.

3.2.2. Phase Shift Keying (PSK)

PSK is a type of digital modulation that sends data by changing the phase of a constant frequency reference signal (the carrier wave). Modulation is achieved by varying the sine and cosine inputs at a specific time. It is the simplest modulation scheme that uses voltage levels of a digital signal to change the phase in the transmitting sinusoidal which is demodulated on the receiver side using a simple phase-locked loop system.

3.2.2.1. Binary Phase Shift Keying (BPSK)

BPSK is a two-phase PSK modulation technique where these two-phase levels are used to represent 1's and 0's in the input signal or the messenger wave. The binary input sequence is multiplied by the high-frequency carrier wave with the help of a Balanced Modulator which outputs the high-frequency carrier wave with alternating phase-shifting(180°) with respect to the input data. Since there are only two levels of phase-shifting BPSK is also known as phase reversal keying or 2PSK.

The pair of signals $s_1(t)$ and $s_2(t)$ are represented using binary symbols 1 and 0 respectively. Where T_b is the bit duration and E_b is the transmitted signal energy per bit. The constellation plot for BPSK is shown in Figure 3.3 [16].

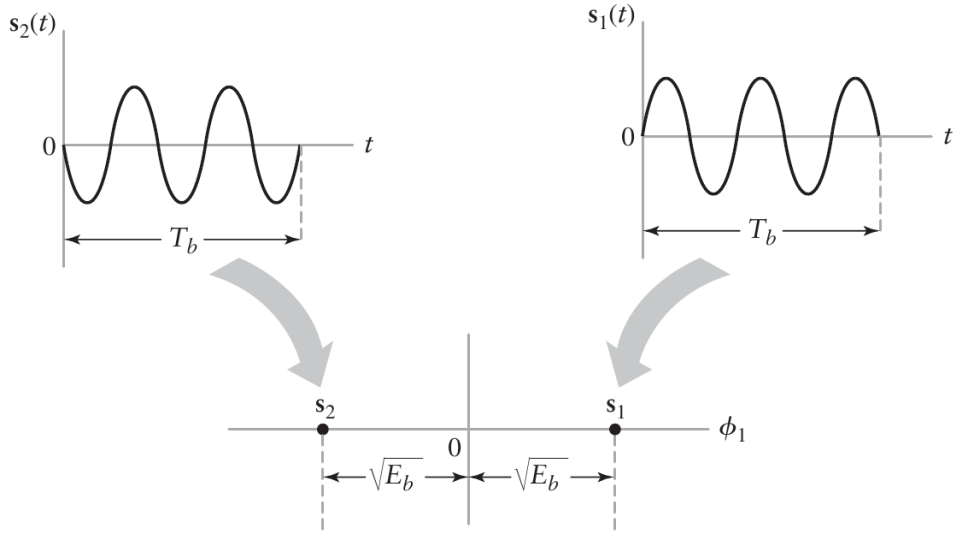


Figure 3.3: Constellation Plot for BPSK

3.2.2.2. Quadrature Phase Shift Keying (QPSK)

QPSK is similar to that of BPSK, but the phase-shifting takes place at 4 levels i.e 90° shifts at a time. Hence the input sequence will have 4 different voltage levels. QPSK is one bandwidth conserving modulation scheme which is well known for the efficient utilization of channel bandwidth. The messaging symbols in QPSK are contained in the carrier phase. The phase of the carrier takes on one of four equally spaced values such as $\pi/4$, $3\pi/4$, $5\pi/4$, and $7\pi/4$. The constellation plot for QPSK is shown in Figure 3.4 [16].

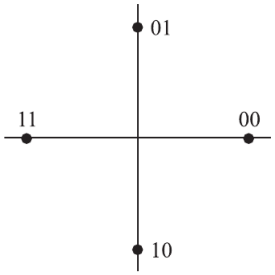


Figure 3.4: Constellation Plot for QPSK

3.2.2.3. 8PSK

8 Phase-Shift Keying is a digital modulation technique that is also similar to BPSK but with 8-level phase changes. To form of phase modulation with eight phase states located at 0, π, +/-(π/4), +/-(π/2) and +/-(3π/4) radians in the IQ plane hence each symbol carries $\log_2(8)$ = 3 bits of information.

3.2.3. Quadrature Amplitude Modulation (QAM)

QAM modulation uses two independently amplitude-modulated carrier signals i.e in-phase carriers and quadrature-phase carriers. The two carrier waves of the same frequency are out of phase with each other by 90°. The signal state in QAM is assigned with in-phase and quadrature carriers thus making it flexible to utilize both amplitude and phase variations. Though its high data capacity of carrying, efficient usage of bandwidth, QAM modulated signals are more susceptible to noise as compared to PSK modulated signals.

3.2.4. Gaussian Frequency Shift Keying (GFSK)

In this modulation, pulse shaping is done prior to the modulation for smooth pulses and limits modulation spectrum width as well as out of band spectrum. The baseband signal is first passed through the Gaussian filter before modulation.

3.3. Analog Modulation

Analog modulation refers to the modulation of high-frequency carrier waves using an analog signal. This modulation is used to transmit low-frequency signals such as TV signals or audio signals. In this type of modulation, a bandpass channel is required where it corresponds to the specified range of frequencies. These frequencies are transmitted over a bandpass filter which allows certain frequencies to pass preventing signals at undesirable frequencies. The type of analog modulation is based on the type of carrier signal property and so there are mainly three kinds of analog modulations and are:

1. Amplitude Modulation
2. Frequency Modulation
3. Phase Modulation

3.3.1. Double Sideband Modulation

Double Sideband Modulation is the modulation in which modulated output signal contains two sidebands of frequencies. A type of DSB, called binary phase-shift keying, is used for digital telemetry. Amplitude modulation (AM) is similar to DSB but DSB has the advantage of permitting a simpler demodulator, the envelope detector. A message signal $x(t)$ can be DSB modulated onto a carrier with simple multiplication and a DSB modulated carrier is normally demodulated with a synchronous detector.

3.3.2. Single side band modulation

Usually, the amplitude modulated signal consists of two reductant sideband signals, but as the name suggests, single-sideband signals use only one of these sidebands. Since the bandwidth of such signals occupies lesser spectrum space than double sideband, more transmission signals are allowed. Transmission power is reduced due to the narrower bandwidth high power signals can be transmitted due to the half uses of the bandwidth of the same AM signal.

3.3.3. Continuous Phase Shift Keying (CPFSK)

CPFSK is a type of traditional frequency shift keyed modulation where the signal is constrained to maintain continuous phase at its symbol time boundaries as its name suggests. This modulation type has noted benefits since this constraint offers better error rate performance as well as signal spectrum containment. Due to its promising modulation format that allows compact spectrum, as well as its receiver sensitivity, which can be improved with a differential detection, it is considered as an important modulation type.

3.3.4. Wideband Frequency Modulation (WBFM)

Widely used for FM broadcasting, Wide-band frequency modulation is preferred when signal quality is required over the spectrum efficiency with the greater expense at spectrum usage. For large values of modulation index m_f , the FM wave ideally contains the carrier and an infinite number of sidebands located symmetrically around the carrier. Such an FM wave has infinite bandwidth. In this modulation technique, the music and speech are transmitted with up to $75kHz$ deviation from the center frequency. It also allows carrying the audio signal up to 20kHz and sub-carriers up to $92kHz$.

3.3.5. Pulse Amplitude Modulation (PAM)

PAM is the simplest analog modulation scheme where the train of pulse carrier is modulated using the analog modulating signal. Here the modulated signal is a train pulse of the carrier signal with modulated amplitude. The natural PAM is susceptible to noise. When the signal is passed through an LPF, it cannot recover the signal without distortion. Hence to avoid this noise, flat-top sampling is preferred. Flat-top sampling is the process in which sampled signal can be represented in pulses for which the amplitude of the signal cannot be changed with respect to the analog signal, to be sampled. The tops of amplitude remain flat.

3.4. Channel Model

Noise is inherently present in real channels due to which a system that does not account for this noise is of little use in the real world. For the design of any part of a communication system, a proper channel model is a must and AMC is no exception to this. Various probabilistic and statistical models have been developed for channel modeling, as described in [17], throughout the years which can be represented as given in figure 3.5:

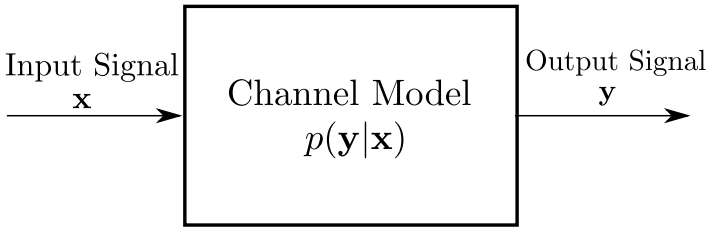


Figure 3.5: A Probabilistic Channel Model

Here, $\mathbf{x}$ is the input to the channel and $\mathbf{y}$ is the output that we observe from the channel. The channel can now be thought of as a system that assigns probabilities to all kinds of output that can be observed by the receiver given a certain input. These probabilities are dependent on the input to the channel. We assume the noise in the channel is to be additive i.e, If $x(t)$ is the input signal to the channel, then the output of the channel $y(t)$ will be given by:

$$y(t) = x(t) + z(t) \tag{3.1}$$

where, $z(t)$ is the additive noise. We also assume that the additive noise $z(t)$ is white noise and has a Gaussian distribution in time. This is commonly known as the Additive White Gaussian Noise (AWGN). The AWGN has zero mean and a power spectral density of $N_0/2$. The AWGN model has been used widely throughout the years because of its simplicity and as a good assumption of noise in a channel. It has provided good performance in modulation classification, especially for various wired communication channels.

To accommodate for wireless channels, we also add the effects of multi-path propagation and signal attenuation. This model is known as the multi-path fading channel. It is generally represented as a linear time-variant system, i.e, the properties of the channel vary with time. Mathematically, the multi-path fading channel can be represented as given in the equation 3.2:

$$y(t) = \alpha(t)e^{j\theta(t)}x(t) + z(t) \tag{3.2}$$

where f_0 is the frequency offset. It is a multiplicative channel model where the time dependent factor $\alpha(t)e^{j\theta(t)}$ represents the time varying channel gain and the phase offset due to the channel. Furthermore, we can also consider the frequency offset of the channel which is caused by the movement between the receiver and the transmitter due to the Doppler effect. The full channel model can now be expressed as given in the equation 3.3:

$$y(t) = \alpha(t)e^{j(2\pi f_0 t + \theta(t))}x(t) + z(t) \tag{3.3}$$

By utilizing these channel models we can make our classifier robust to the various unknown changes that occur in the physical world.

3.5. Spectrogram

A spectrogram is a visual representation of a signal's signal strength across time at various frequencies included in a waveform. It is two dimensional graph with colors acting as the third dimension. Along the horizontal axis, time passes from left to right. The third dimension, color, represents the amplitude of a certain frequency at a given moment. Music, linguistics, sonar, radar, speech processing, seismology, and other fields use spectrograms extensively.

The time representation is the one way to describe a signal and the frequency representation is another significant way to describe a signal. Neither the time representation nor the frequency representation, on the other hand, could tell us when the frequencies occur. As a result, TFR (Time-Frequency Representation) method was developed which is a combination of time and frequency bridging the gap between time and frequency representation. The Short-Time Fourier Transform (STFT) is an intuitive and simple method of TFRs [18].

3.6. Likelihood Ratio Test

The likelihood is the measure of how accurately the given data fit a model with a certain set of fixed parameter values. The likelihood function is a function that is used to calculate the likelihood for a given data. Let $\boldsymbol{X}$ be a random variable that represents the observed data and $\boldsymbol{\theta}$ be the set of parameters used in our model. Then, the likelihood of observing the data $\boldsymbol{x}$ is denoted by $f(\boldsymbol{x};\boldsymbol{\theta})$ and is given by:

$$f(\boldsymbol{x};\boldsymbol{\theta}) = P_{\boldsymbol{\theta}}(\boldsymbol{X} = \boldsymbol{x}) \tag{3.4}$$

where, $P_{\boldsymbol{\theta}}(\boldsymbol{X})$ represents the probability density function parameterized by $\boldsymbol{\theta}$.

The likelihood ratio test is a statistical test for performing hypothesis testing by comparing the likelihood of the data under different hypotheses. Let $f(\boldsymbol{x}|H_0)$ be the likelihood of the observed data $\boldsymbol{x}$ under the hypothesis H_0 and the $f(\boldsymbol{x}|H_1)$ be the likelihood under

hypothesis H_1. Then, the likelihood ratio is given by:

$$\Lambda(\boldsymbol{x}) = \frac{f(\boldsymbol{x}|H_1)}{f(\boldsymbol{x}|H_0)} \tag{3.5}$$

If π_0 and π_1 be the priors of H_0 and H_1 i.e, the probability of observing hypothesis H_0 and H_1 respectively, the ratio of the priors is denoted as:

$$\kappa = \frac{\pi_0}{\pi_1} \tag{3.6}$$

Then, the decision is made as follows:

1. If $\Lambda(\boldsymbol{x}) > \kappa$, then accept H_1.
2. Otherwise, accept H_0.

In the case that both priors are equal, i.e., both hypotheses have an equal chance of appearing, then $\kappa = 1$, and the above decision condition is reduced to the following:

1. if $f(\boldsymbol{x}|H_1) > f(\boldsymbol{x}|H_0)$, accept H_1.
2. Otherwise, accept H_0.

In the average likelihood ratio test, the likelihood under a given hypothesis is averaged over the unknown quantity. It provides maximum classification accuracy when the distribution of the unknown quantity coincides with the actual underlying distribution.

3.7. Deep Learning

Deep learning is a type of machine learning technique or a subset of machine learning that has deeper layers of neural networks. These neural networks attempt to simulate the working of human brains by processing, analyzing, and making cognitive decisions based on the observed input data in such a way that the precision and decision-making ability of these models increase with time.

Deep learning neural networks are the type of network that tends to higher-level features extractions from the raw input. These models are trained by using a large set of labeled data and neural network architectures that contain many layers. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. In today's

world, it is a key technology behind automatic driving cars, enabling them to recognize a stop sign or to distinguish a pedestrian from a lamppost and voice control in consumer devices like phones, tablets, and TVs.

3.7.1. Convolutional Neural Network (CNN)

Convolutional Neural Network (CNN) is a type of Artificial Neural Network architecture that is used to extract spatial information from the given set of inputs. Spatial information is the positional pattern or relationship in the input space. The convolution operation of input with a finite-sized kernel results in the extraction of local spatial information.

Convolution is a mathematical operation on two functions that gives the amount of overlap as one of the functions is flipped and shifted across the other function. Mathematically speaking, we have, the convolution of two functions $f(t)$ and $g(t)$, denoted by $(f*g)(t)$ is given by:

$$s(t) = (f * g)(t) = \int_{-\infty}^{\infty} f(a)g(t-a)da \tag{3.7}$$

For discrete-time functions and signals we can rewrite same convolution operation as:

$$s[n] = (f * g)[n] = \sum_{k=-\infty}^{k=\infty} f[n]g[n-k] \tag{3.8}$$

In the field of deep learning, the first function of the convolution operation is known as the input whereas the second function is referred to as the kernel, and finally the output as the feature map. The input to a CNN can be a signal of multiple dimensions i.e. audio (1D), image (2D) and medical images (3D). The kernel also can be of multiple dimensions according to input. This is in contrast to the single dimensional convolution that we have seen in equations 3.7 and 3.8.

The typical operation performed between input and kernel in a neural network is not exactly convolution but instead is cross-correlation. Cross-correlation is the measure of similarity between 2 signals and differs from convolution in that there is no signal flipping before the application of multiplication and summation.

In CNN, the neural network is designed to learn the kernels based on the input image and output labels. The kernel learned by a neural network using cross-correlation will just be a relatively flipped version of the kernel learned by a neural network using convolution.

Hence, it does not make much difference if either of the operations is used. Most neural network implementations use cross-correlation as the operation. From hereon the use of the word convolution refers to the one done in CNNs i.e. cross-correlation unless explicitly specified. The overall operation is illustrated in Figure 3.6 [19].

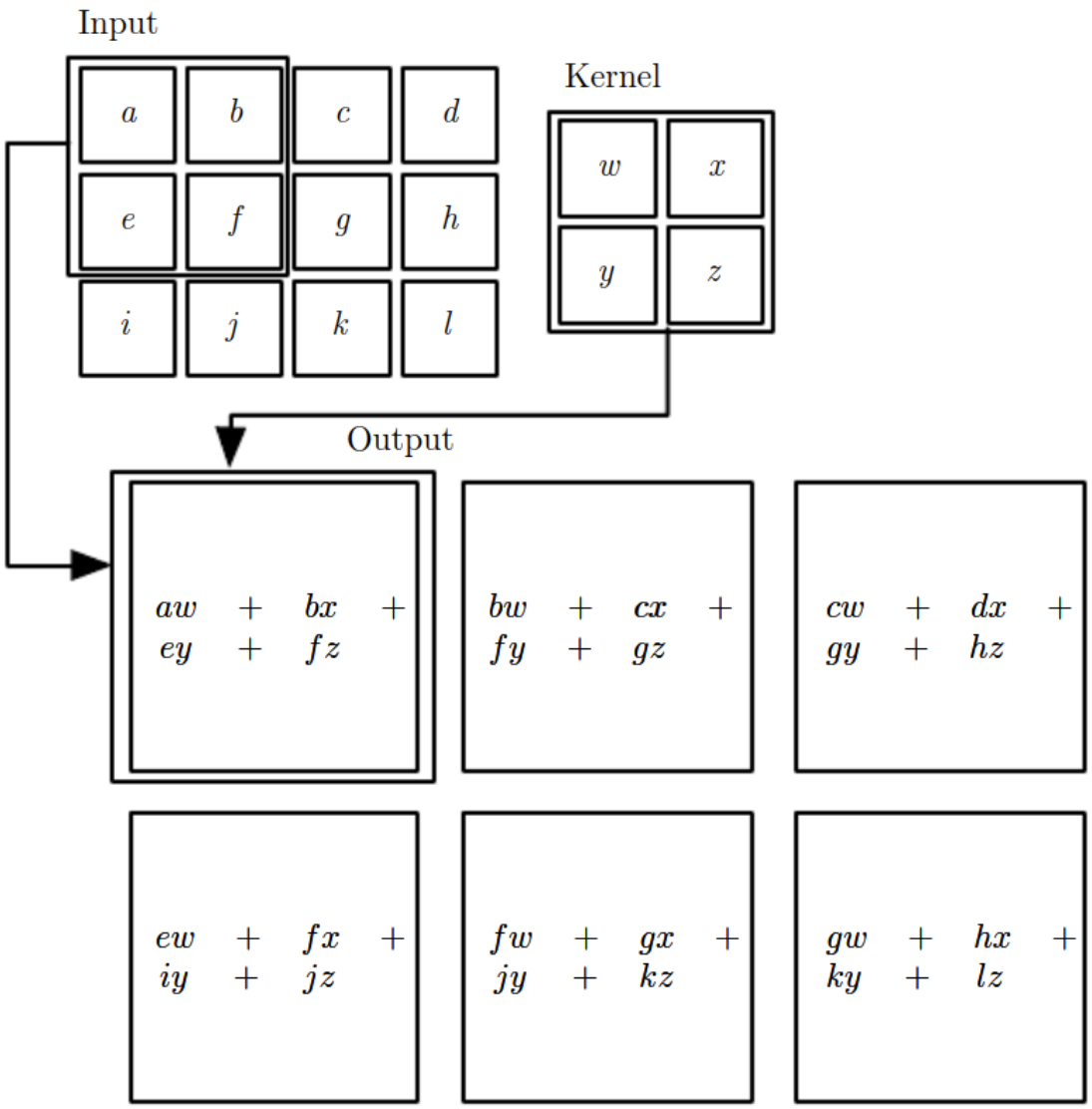


Figure 3.6: Convolution operation

The amount of shift that the kernel goes through before the next multiplication and summation is known as a stride. A larger stride corresponds to lower computational complexity at the cost of the lower resolution of feature map whereas a smaller stride corresponds to greater detail of feature extraction with increased computation complexity.

Since only a few local inputs are matched to a single output sample of the feature map, a CNN is computationally more efficient than the Fully Connected Neural Network. Whilst

saying that, if the depth of the convolutional feature extraction increases, the receptive field of a single output neuron vastly increases. The receptive field of a neuron is the portion of the input from which any amount of information is flowing into the neuron. This is illustrated in Figure 3.7 [19]. So, we can create a feature extracting, computationally better neural network without losing the accuracy with the help of a CNN.

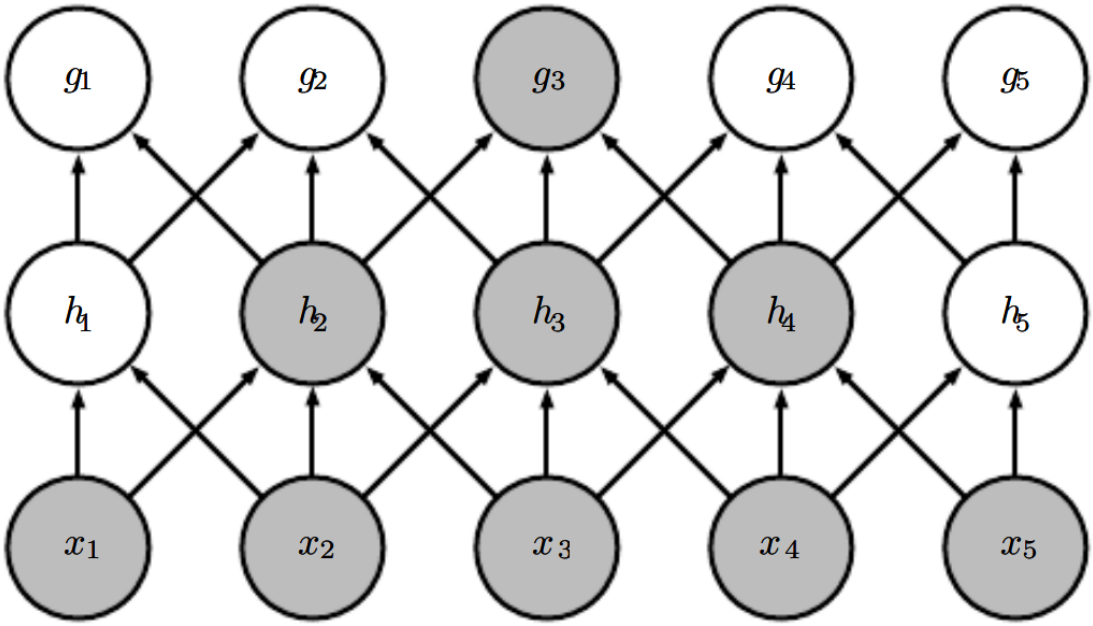


Figure 3.7: Sparse connection and Increasing receptive field with depth.

A typical convolutional layer consists of 3 discrete steps: the convolution step, the detector step, and the pooling step. In the convolution step the input is convoluted with the kernels; in the detector step the output is passed through a non-linear activation function and in the pooling step the dimension of the feature map is reduced as required by replacing a bunch of values with a statistical summary of these values (max, average). The layer is illustrated in Figure 3.8 [20].

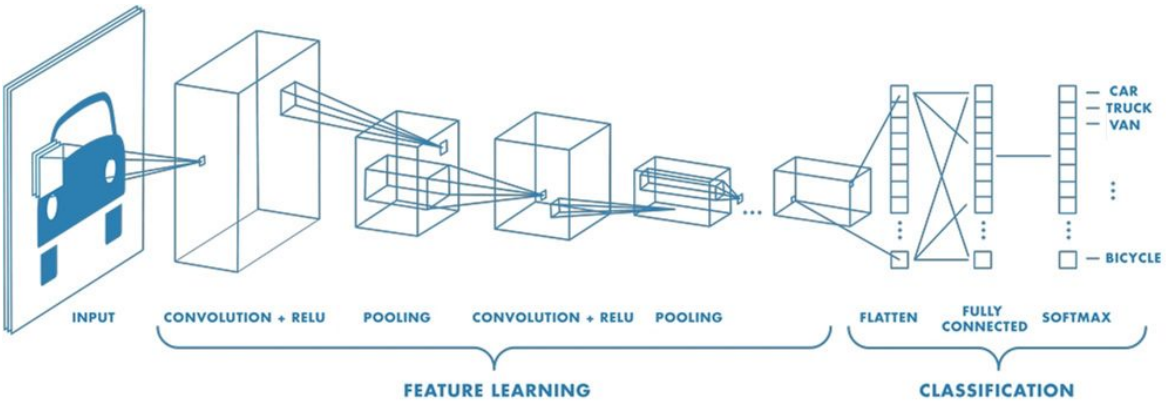


Figure 3.8: Illustration of a convolutional layer

3.7.2. Recurrent Neural Network (RNN)

Unlike most of the deep-learning models, the RNN is a DL model with the concept that not only encapsulates the working mechanisms of other typical DL models but also incorporates the feature of processing the sequential inputs and the recurrent data structures. Instead of connected neurons in traditional DL models, the RNNs have chained memory cells that possess activation functions. Each of these nodes or the memory cells is interlinked with each other such that each one of the cells passes on some weight of current or the previous input to the preceding cell. Hence these idea allows holding the past information for processing the current input and somewhat ensures to capture of the long-range time dependencies of the input data making it extremely useful in areas like NLP, audio/video processing, and so on, that have a sequential flow of data and somehow require the knowledge of prior inputs. This model also benefits from reduced network size and computational cost due to its shared parameters across different time-steps in the architecture.

On the basis of input and output data sequences, the RNNs can be classified into various types as shown in figures 3.10, 3.11, 3.12, and 3.13.

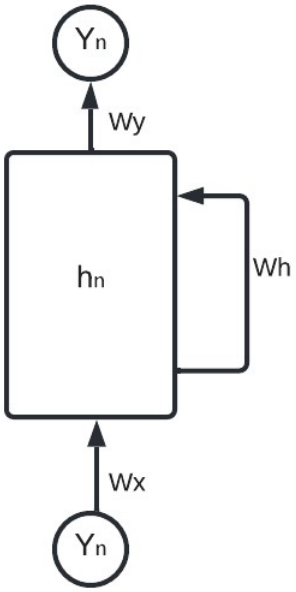


Figure 3.9: RNN Structure

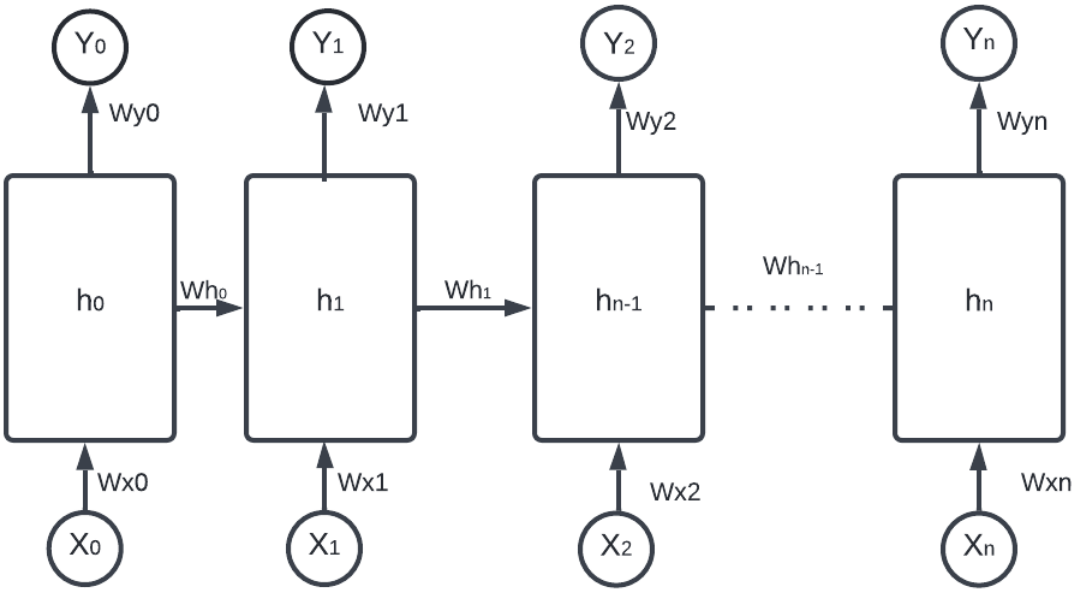


Figure 3.10: Many to many Type I RNN architecture

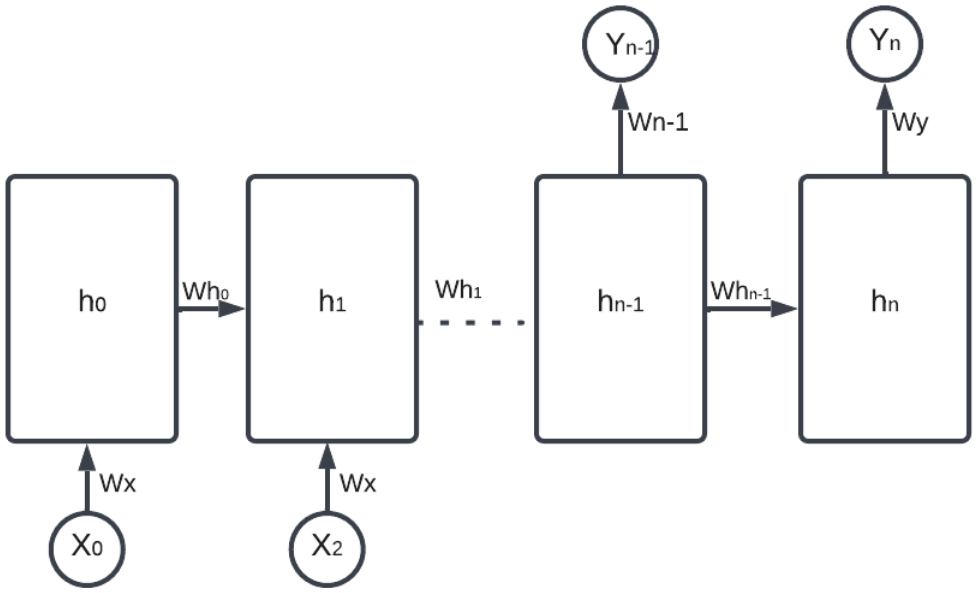


Figure 3.11: Many to many Type II RNN architecture

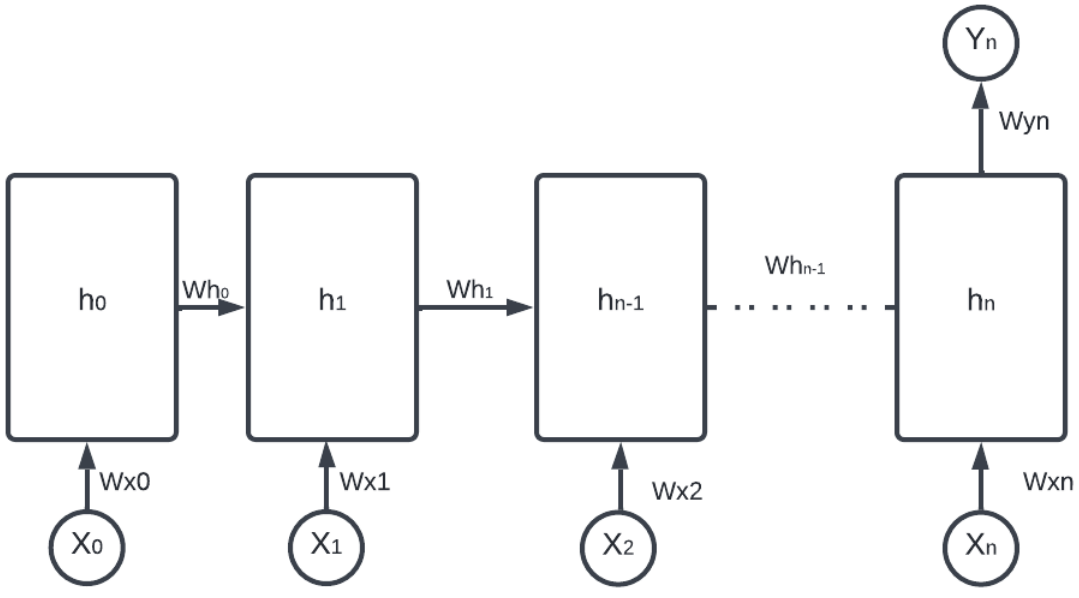


Figure 3.12: Many to one RNN architecture

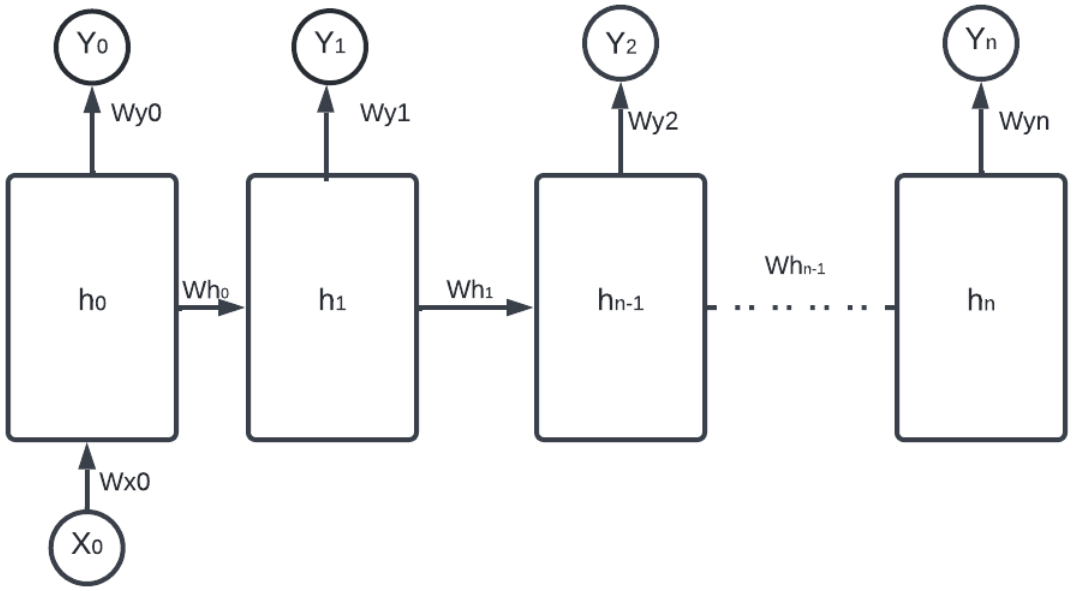


Figure 3.13: One to many RNN architecture

Here, Figure 3.9 depicts the actual architecture of the RNN model where w_x, w_o, and w_h are the input, output, and cell weights respectively. They are also the shared parameters across the model such that each type of weight matrices possesses the same value for each of the cells in the particular model. On unfolding this model we get to look deeper into the linkage of the cells and the type of processing as per the data sequences as shown in Figures 3.10, 3.11, 3.12, and 3.13. RNNs adversely process very long sequences if configuring the activation functions $\tanh$ or rectified linear unit (ReLU) where the function outputs zero with any negative input. A deeper peek into the individual cell of RNN is shown in 3.14.

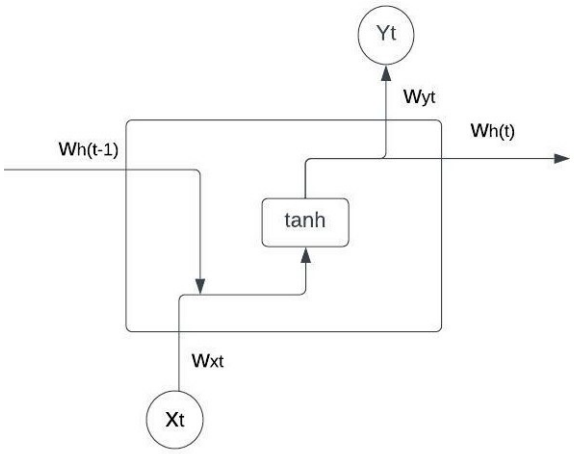


Figure 3.14: RNN cell architecture

As mentioned above, RNN is a derivation of feed-forward networks that are especially focused on modeling sequential data. For the recurrent sequence modeling, prior knowledge of the input stream is required. Thus each cell in this model acts as a memory cell where the state of the previous state $\mathbf{h_{t-1}}$ is fed to the next cell along with the input $\mathbf{x}_t$ which produces the output for the corresponding cell as $\mathbf{y}_t$ along with the state of the cell $\mathbf{h}_t$, that will be fed to the preceding cell along with the input. Since these memory cells are not backed up and are limited to only a short period RNNs have certain criteria:

1. Incompatible with the variable length sequence
2. Long term dependencies
3. Requires maintenance info about the order
4. Shared parameters across the sequence

Along with these criteria, RNNs also face rapid fluctuations in the gradients. These fluctuations in the gradients are either exploding gradient problems or vanishing gradient problems. To overcome and compensate for these conditions, various techniques can be applied such as changing activation functions, random initialization of weight matrices, or change in the architecture of the cells.

3.7.2.1. Long Short Term Memory (LSTM)

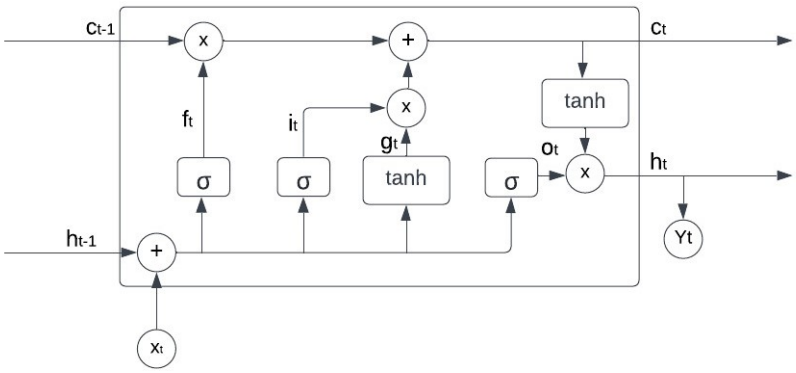


Figure 3.15: LSTM cell architecture

LSTM is an architecture, of the RNN model that allows long-term dependency tracking. This approach in RNN is the third technique i.e change in the architecture of the cells that offer effective modeling of sequential data, omit vanishing gradient problems, and maintain long-term dependency. Few changes are made in the existing architecture of the cell by designing them as complex recurrent units.

LSTM is also known as a gated cell since few gates are added to the existing model of the RNN cell. These added gates put on four new features to the RNN's cell i.e STORE, FORGET, UPDATE, and, OUTPUT. The STORE gate only stores the relevant information to the cell state whereas FORGET gate erases the irrelevant information from the past. The UPDATE gate maintains the separate value of the cell state and updates accordingly. Finally, the OUTPUT gate passes the inputs along with prior cell states and biases through the activation function. For these features, four new gates are added which are: forget gate(ft), input gate(it), cell candidate(gt) and, output gate(ot).

The architecture of LSTM regulates the information flow in the storage of the cell along with a better grip for longer-term dependencies. Since separate cell states are maintained and gate control mechanisms are implemented, this allowed the back propagation through time with uninterrupted gradient flow. Thus, the vanishing gradient problem is overcome due to the uninterrupted gradient flow by maintaining the cell state.

3.7.2.2. Bidirectional Long Short Term Memory (Bi-LSTM)

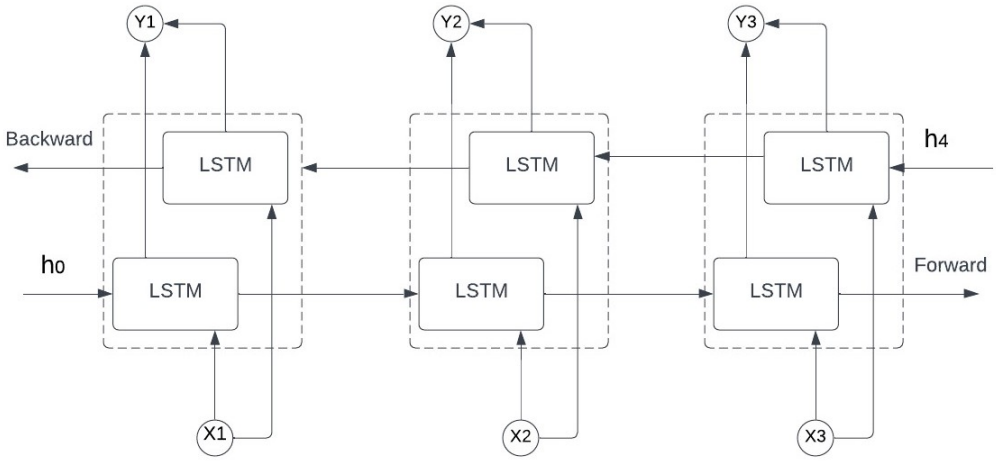


Figure 3.16: Bi-LSTM architecture

Bi-LSTM is an extension of traditional LSTM where two blocks of LSTM are placed parallel forming a single cell of Bi-LSTM such that one LSTM cell takes the input in a forwarding direction and the other one takes in the backward direction. This model processes the sequence input by enabling additional training by traversing the input twice which gives deep features from the lowest level to the highest level from large a data-set.

3.7.2.3. Attention Layer

The attention layer simply implements the idea of the word “attention”. This layer implements the basic idea of attention i.e selectively focusing on a few specific features while ignoring other information or concentrating on relevant information. Primarily the thought of attention layer was introduced to overcome the initial encoder and decoder-based neural machine translation system in NLP where the bottleneck between the two parts i.e, encoder and decoder side faced a lot of data loss as well as increased complexity and error for variable or long sequence data.

Attention is proposed as a method to both align and translate. Alignment is the problem in machine translation that identifies which parts of the input sequence are relevant to each

word in the output, whereas translation is the process of using the relevant information to select the appropriate output. In this layer, instead of encoding the input sequence into a single fixed context vector, the attention model develops a context vector that is filtered specifically for each output time step thus eradicating the need for long term dependencies and focusing only on the specific input features without having to track it backward.

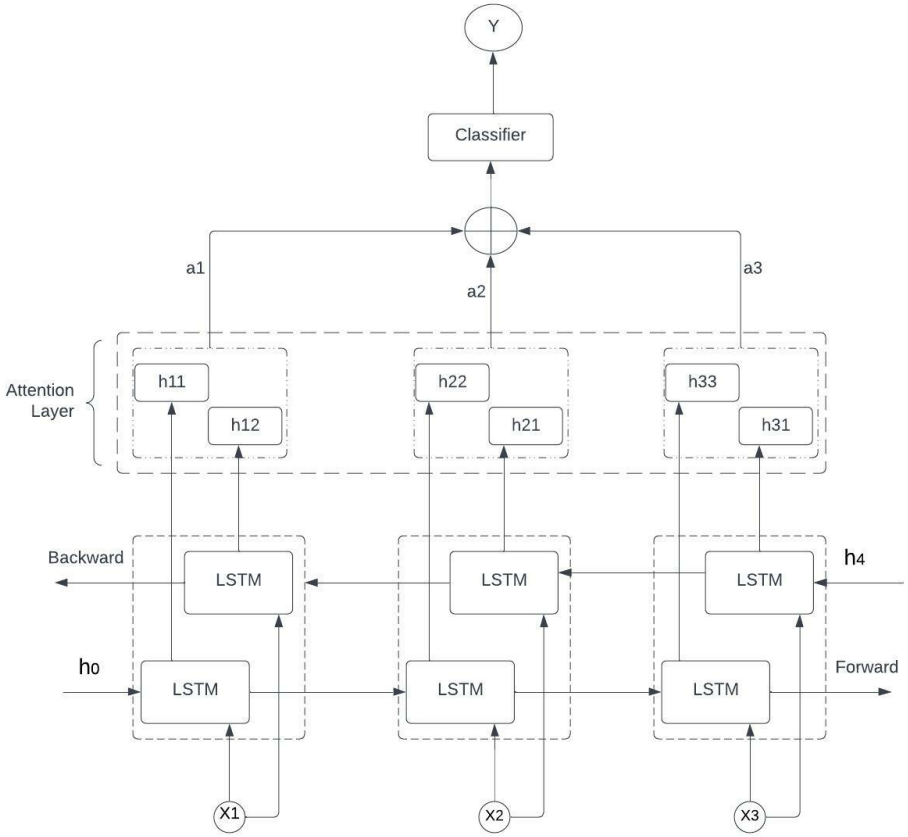


Figure 3.17: Attention layer architecture in Bi-LSTM

3.8. Quantum Neural Networks

Quantum computing is the implementation of working of quantum physics to replicate similar phenomena in ways of computing. This type of computation encapsulates the collective properties of quantum states, such as superposition, interference, and entanglement. Quantum computers have qubits instead of bits to run multidimensional quantum algorithms. Unlike the classical bit which can possess either 0 or 1 value at a time, the qubits can have any state in between these values. They have the ability to be put into a superposition and share entanglement with one another. Due to these features, quantum computers can perform

quantum operations that are difficult to emulate at scale with classical computers. Due to the ability to compute the complex algorithm with extremely fast speed quantum computers can help achieve results that are not possible to chive with classical computers.

Quantum neural networks are computational neural network models influenced by the phenomena observed in quantum mechanics. Typical research in quantum neural networks involve combining classical artificial neural network models with the advantages of quantum information in order to develop more efficient algorithms.

For the realization of these algorithms, Qiskit or Cirq are used. Qiskit is open-source software development kit for working with quantum computers at the level of pulses, circuits, and application modules. Qiskit includes a comprehensive set of quantum gates and a variety of pre-built circuits such that users at all levels can use Qiskit for research and application development. On the other hand, Cirq is a Python software library by Google, for writing, manipulating, and optimizing quantum circuits, and then running them on quantum computers and quantum simulators. Cirq provides useful abstractions for dealing with today's noisy intermediate-scale quantum computers, where details of the hardware are vital to achieving state-of-the-art results.

3.9. GNU Radio

GNU Radio is a free software development toolkit that contains signal processing blocks for creating software-defined radios and signal processing systems. Which can be used to create software-defined radios with external RF hardware, or it can be used in a simulation-like environment without hardware. It's widely used to support both wireless communications research and real-world radio systems in hobbyist, academic, and commercial environments.

The GNU Radio software provides a framework and tools for developing and running software radio applications, as well as general signal-processing applications. GNU applications are known as flowgraphs, which are a series of signal processing blocks. Different blocks are available which can be used to generate the modulated data. Combination of different noise blocks, fading blocks, and so on are suitable for creating the simulated environment whereas the different modulation blocks are used to generate modulated data.

4 METHODOLOGY

4.1. System Block Diagram

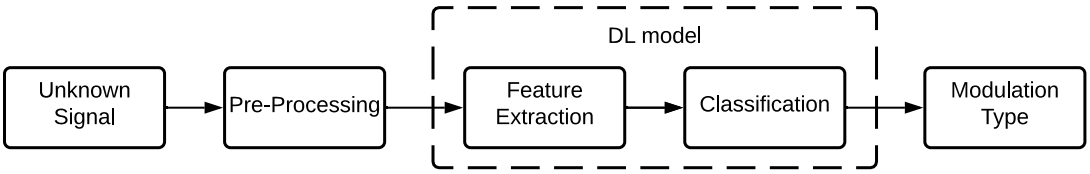


Figure 4.1: AMC System Block Diagram

4.2. Likelihood Based Classifier

4.2.1. Signal and Channel Model

We have used an AWGN channel with phase distortion in our model for the likelihood ratio test as given in [21]. The model is given by equation 4.1.

$$\boldsymbol{r}^{(m)} = \alpha e^{j\psi} \boldsymbol{s}^{(m)} + \boldsymbol{w} \tag{4.1}$$

where,

$\boldsymbol{r}^{(m)}$ = Received signal vector corresponding to modulation m

$\boldsymbol{s}^{(m)}$ = Transmitted symbol vector corresponding to modulation m

ψ = Phase distortion factor

$\boldsymbol{w}$ = Gaussian Noise with zero mean and variance $N/2$

$\alpha = \sqrt{\text{SNR} \times N}$ = Signal amplitude for the given SNR level and noise variance

Here, $\boldsymbol{r}$ and $\boldsymbol{s}$ are complex vectors, where each component, $r_k^{(m)}$ and $s_k^{(m)}$ represents the IQ representation of the signal sample k of modulation m.

4.2.2. Signal Generation and Visualization

The algorithm GENERATE_SIGNALis used generate the dataset to test our likelihood based classifier. It generates a vector of received symbols that has passed through a AWGN channel as specified by equation 4.1. First, the algorithm generates a set of all possible symbols belonging to the modulation m and a random phase angle that will be applied to all the symbols. It then picks a random signal from the constellation set and applies the channel model given in equation 4.1 to the symbol. This process is carried out for the given number of signal length n. The procedure is outlined in algorithm 4.1.

Algorithm 4.1 Signal Generation Algorithm

Input: Modulation type m, the number of signal points K and the SNR γ in decibels.

Output: Received signal $\boldsymbol{r}^{(m)}$ and the phase noise parameter ψ

procedure GENERATE_SIGNAL(m,K,γ)
  $\boldsymbol{M} =$ CONSTELLATION(m)
  $\alpha = 10^{\gamma/10}$
  $\psi = \mathcal{N}(0, 1)$
  for $k = 1$ to K **do**
    $s_k^{(m)} =$ Random constellation point from the constellation set $\boldsymbol{M}$
    $r_k^{(m)} = \alpha e^{j\psi} s_k +$ COMPLEX_NORMAL$(0, N/2)$
  end for
  return $\boldsymbol{r}^{(m)}$, ψ
end procedure

Here, CONSTELLATION is a procedure which returns a set of all the symbols that belong to the modulation m. Also, COMPLEX_NORMAL is a procedure which returns a complex number which follows the complex Gaussian distribution which is given in equation 4.2.

$$\text{COMPLEX\_NORMAL}(\mu, \sigma^2) = \frac{1}{\sqrt{2}} \left(\mathcal{N}(\mu, \sigma^2) + j\mathcal{N}(\mu, \sigma^2) \right) \tag{4.2}$$

Here, $\mathcal{N}(\mu, \sigma^2)$ represents a Gaussian random variable with mean μ and variance σ^2.

4.2.3. ALRT of the data

The likelihood function used for our classifier, as given in [21], is shown in equation 4.3.

$$f^{(m)}(\boldsymbol{r}|\boldsymbol{\Theta}) = \prod_{k=1}^{K} \frac{1}{M^{(m)}} \left(\sum_{l=1}^{M^{(m)}} \frac{1}{\pi N} \exp\left(\frac{-|r_k - \alpha e^{j\psi} s_l^{(m)}|^2}{N} \right) \right) \tag{4.3}$$

Here, m represents the modulation, $\boldsymbol{r}$ represents the received signal vector of length K and $\boldsymbol{\Theta}$ is the parameter vector given by:

$$\boldsymbol{\Theta} = \begin{bmatrix} \alpha & \psi & N \end{bmatrix}^T \tag{4.4}$$

Thus, the likelihood function $f^{(m)}(\boldsymbol{r}|\boldsymbol{\Theta})$ calculates the probability that the received signal $\boldsymbol{r}$ belongs to modulation m. Also, $M^{(m)}$ is the number of symbols present in modulation m and $s_l^{(m)}$ represents a constellation point of the constellation set $\boldsymbol{M}$ and $|z|$ represents the magnitude of the complex number z. The likelihood function $f^{(m)}(\boldsymbol{r}|\boldsymbol{\Theta})$ for BPSK and QPSK are as shown below:

$$f^{\text{BPSK}}(\boldsymbol{r}|\boldsymbol{\Theta}) = \prod_{k=1}^{K} \frac{1}{2} \left(\sum_{l=1}^{2} \frac{1}{\pi N} \exp\left(\frac{-|r_k - \alpha e^{j\psi} s_l^{\text{BPSK}}|^2}{N} \right) \right) \tag{4.5}$$

$$f^{\text{QPSK}}(\boldsymbol{r}|\boldsymbol{\Theta}) = \prod_{k=1}^{K} \frac{1}{4} \left(\sum_{l=1}^{4} \frac{1}{\pi N} \exp\left(\frac{-|r_k - \alpha e^{j\psi} s_l^{\text{QPSK}}|^2}{N} \right) \right) \tag{4.6}$$

We assume that the received signal $\boldsymbol{r}$ has equal chances of either being a BPSK or QPSK signal. Then, from the equal prior case of section 3.6., we say that the modulation of the received signal $\boldsymbol{r}$ is BPSK if $f^{\text{BPSK}}(\boldsymbol{r}|\boldsymbol{\Theta}) > f^{\text{QPSK}}(\boldsymbol{r}|\boldsymbol{\Theta})$ and is QPSK modulated otherwise.

Since the parameter vector $\boldsymbol{\Theta}$ is already known to us, they don't have to be estimated from the received signal.

4.3. Deep learning based classifiers

4.3.1. Dataset

The RadioML 2016.10A was selected as the standard database for training and evaluating the implemented AMC classifiers. The RadioML is an open-source, synthetically generated dataset as a part of the GNURadio Extended Universe which has been used in various research works over the years. The RadioML 2016.10A, in particular, was selected here for including 11 different modulation techniques while retaining a lighter file size, allowing for rapid development, sharing, and evaluation of different AMC classifiers. The dataset contains labeled received signal samples IQ samples which are synthetically created using GNURadio after pushing the modulated signal through different mathematical channel models. The dataset consists of 11 modulations (8 digital and 3 analog) at varying signal-to-noise ratios. The modulations contained in the data set are 8PSK, AM-DSB, AM-SSB, BPSK, CPFSK, GFSK, PAM4, QAM16, QAM64, QPSK and WBFM. For each modulation scheme, the signals vary in SNR levels from -20 dB to 20dB with the step size of 2db. And each of the SNRs in a particular modulation scheme consists of 1000 signals; each one comprising 128 complex floating-point time samples stored in form of Inphase and Quadrature phase samples. This results in a matrix of (2×128) samples for each signal and a total of $(11 \times 21 \times 1000)$ i.e. 231000 signals. The entire dataset is 640.92 MB in total.

4.3.2. Classification using CNN networks

CNN based classifier for Automatic Modulation Classification was first proposed by O'Shea et al. in [6]. CNN-based classifiers exploit the spatial relations in the constellation diagrams of the input data for pattern recognition. The same model from [6] is implemented here as the baseline CNN model for performance evaluation. They use 2 convolutional layers for feature extraction and 2 dense layers for classification. The Radio-ML architecture of [6] is illustrated in Figure 4.2 with a dropout rate of 0.6 and a $60 - 40$ train-test split of the dataset.

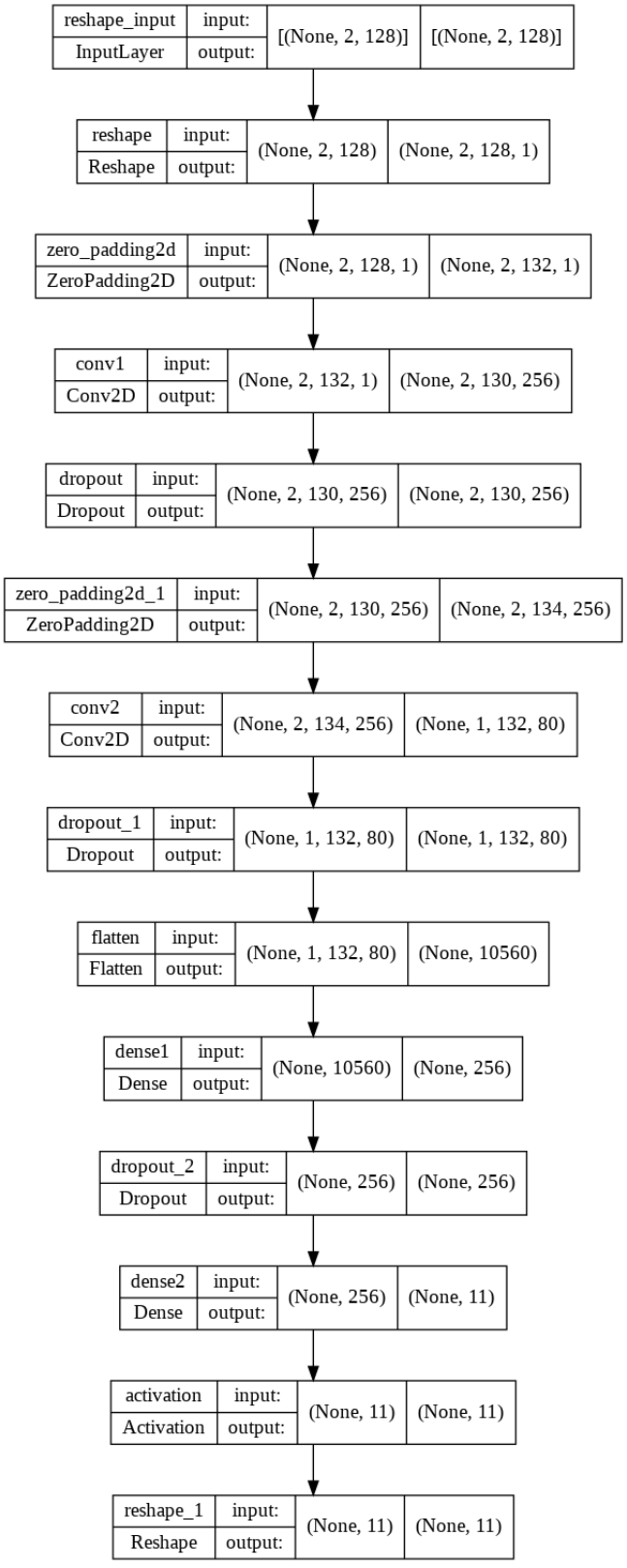


Figure 4.2: Network architecture for Convnet.

To improve upon the classification accuracy of the CNN model, we modified the Convnet model. We added a batch normalization after each convolution operation, to improve

the generalization of the model and to reduce over-fitting in the training data.This way the model can be further trained whilst generalizing to the testing data. Also, during training, we removed the softmax activation at the output layer to increase the resolution of the output for faster convergence of the model.

To summarize, we have a model with 2 convolution layers: first with 256 filters and the later with 80 filters and both having ReLU as the activation function. Each convolution layer is followed by a batch normalization layer and a dropout layer with dropout rate of 0.6 and preceded by a zero padding layer. After the feature extraction by convolution layer, the feature map is flattened and passed to a dense layer with 256 hidden units and ReLU as the activation function. It is also followed by a dropout layer with dropout rate of 0.6. Finally the output layer is a dense layer with 11 nodes and no activation function. The architecture is illustrated in Figure 4.3 and the summary is given in Table 4.1

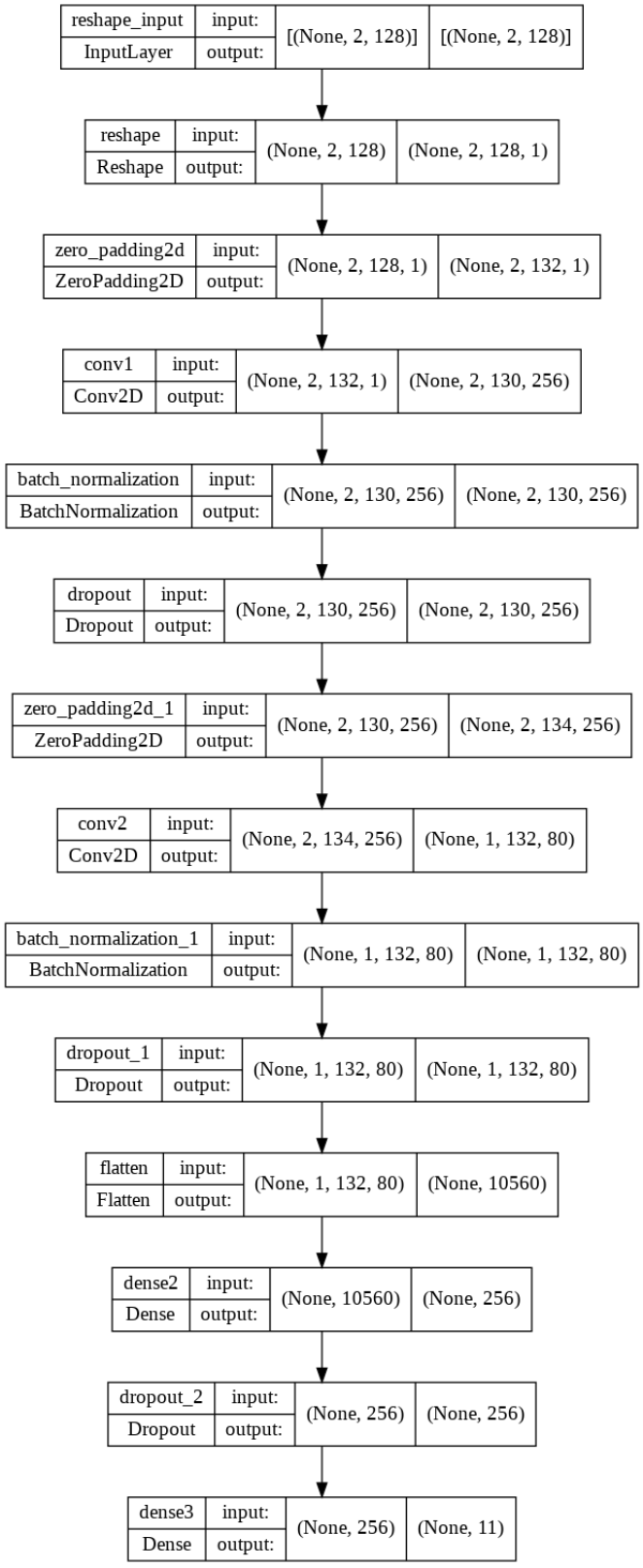


Figure 4.3: Network architecture for Modified RadioML.

Table 4.1: Summary of the modified CNN network

Input Dimension (for a single sample)	128×2
Number of Layers	14
Number of epochs	100
Batch size	1024
Optimizer	Adam
Loss function	Categorical Cross Entropy

These architectures were trained on the entire RadioML-2016.10A. The input data to the network are in the form of constellation diagram of dimension 128×2. We used a $60-40\%$ split between training and testing data for training these networks. These networks were trained for 100 epochs with a batch size of 1024 with early stopping enabled. The Adam optimization algorithm was used along with the categorical cross-entropy as the loss function for these networks.

4.3.3. Classification using LSTM and Bi-LSTM networks

The next approach to the AMC problem is to exploit the temporal relations in the time sequence signal samples using RNN based classifiers. One of the implementations of the RNN classifier is LSTM with an attention layer given by S. Hu et al.(2018) in [8]. To look at the importance of the attention layer, we removed it from the architecture in [8]. Hence the architecture consists of four stacked LSTM layers, and three fully connected (Dense) layers with ReLU activation and a final dense layer with softmax activation for classification as shown in Figure 4.4. All of the hidden states of the first two LSTM networks are passed to the next corresponding LSTM layer. The final LSTM layer outputs only the final state of the layer. This output is then operated upon by the dense layers and then finally classified using the softmax activation layer. A summary of the architecture is given in Table 4.2.

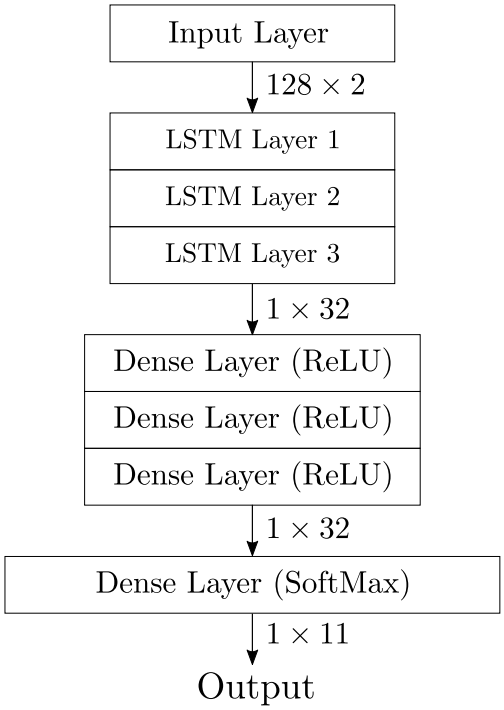


Figure 4.4: Network architecture for classification using LSTM.

Table 4.2: Summary of the used LSTM network

Input Dimension (for a single sample)	128×2
Number of Units in LSTM Layer	32
Number of Layers	7 (3 LSTM + 4 Dense Layer)
Output dimensions of LSTM Layers 1 and 2	128×32
Output dimension of LSTM Layer 3	1×32
Output dimension of Dense Layer (ReLU)	1×32
Number of epochs	100
Batch size	1024
Optimizer	Adam
Loss function	Categorical Cross Entropy

The original architecture from [8] uses an additional attention layer that allows weights of input IQ signals to better track the importance of features in the identification of the modulation scheme. This architecture is taken as the standard baseline for RNN AMC model for

performance evaluation. All the parameters of the previous model were kept unchanged. The newly added attention layer allows us to weigh the importance of each time step of our input data into the model, thus allowing the network to better track the more important information regarding the modulation scheme. The overall architecture of the model is shown in Figure 4.5. Here, all the hidden states of the previous LSTM layers are passed to the attention layer. The attention layer used contains its own weight matrix $\mathbf{W_a}$ and a bias vector $\mathbf{b_a}$. If $\mathbf{H}$ is the input matrix of dimension $M \times N$ to the attention layer, the output of the layer $\mathbf{y}$ is calculated as:

$$\mathbf{e} = \tanh\left(\mathbf{H}\mathbf{W_a} + \mathbf{b_a}\right) \tag{4.7}$$

$$\boldsymbol{\alpha} = \text{softmax}(\mathbf{e}) \tag{4.8}$$

$$y_k = \sum_{i=0}^{M-1} \alpha_i H_{ik} \tag{4.9}$$

The vector $\boldsymbol{\alpha}$ is commonly known as attention weights. As stated above, the matrix $\mathbf{H}$ is the matrix containing all the hidden states of the previous LSTM layer, and which in our case has the dimension 128×32. The matrices $\mathbf{W_a}$ and $\mathbf{b_a}$ have dimensions 32×1 and 128×1 respectively.

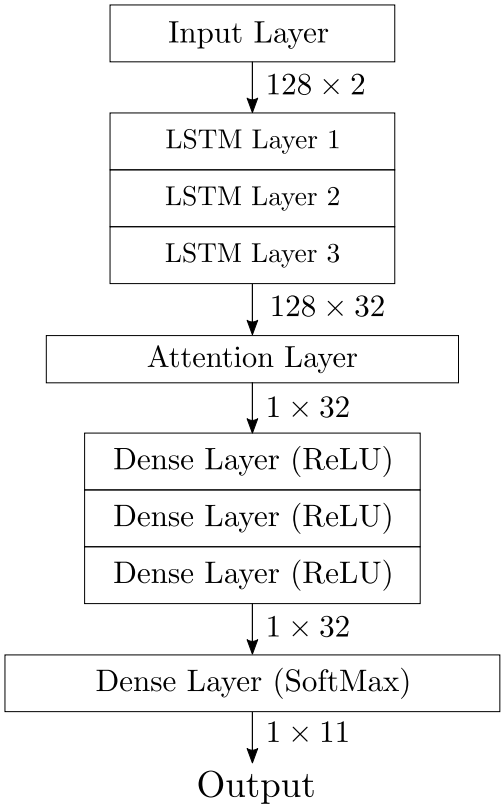


Figure 4.5: Network architecture for AMC using LSTM with attention layer

To increase the preservation of temporal information, the LSTM layers in the baseline model were replaced with Bi-LSTM layers. The Bi-LSTM layer allows a two-way data flow using 2 LSTM blocks. This improves performance than the uni-directional LSTM by preserving past as well as future information. The increased number of LSTM blocks, in turn, results in an increased number of parameters.The resulting architecture is similar to the previous architectures with only a slight increase in the dimensions. The Bi-LSTM architecture is shown in Figure 4.6.

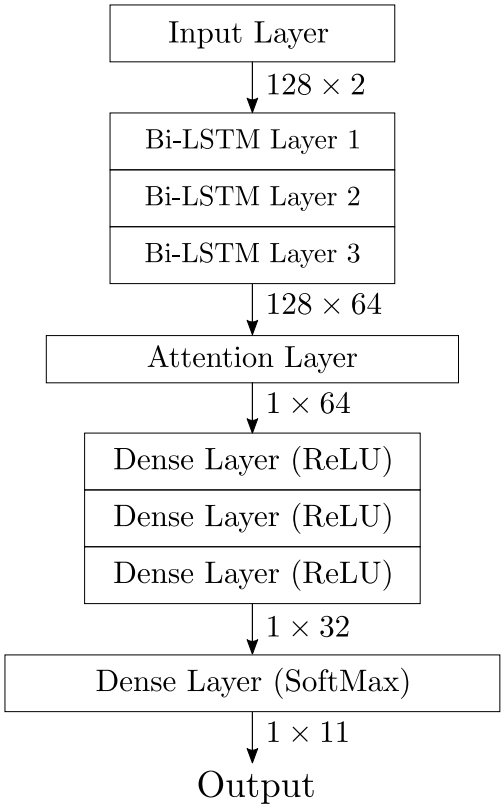


Figure 4.6: Network architecture for AMC using Bi-LSTM layers.

For these architectures, each of the LSTM layers consists of 32 outputs units. The LSTM network was trained on the entire RadioML-2016.10A. The input data to the network are in the form of sequential IQ samples and make up an input vector of dimension 128×2. We used a $50-50\%$ split between training and testing data for training these networks. These networks were trained for 100 epochs with a batch size of 1024 with early stopping enabled. The Adam optimization algorithm was used along with the categorical cross-entropy as the loss function for these networks.

4.3.4. Classification using quantum neural networks

A modulation classifier based on quantum machine learning was also implemented. To achieve this, the TensorFlow Quantum (TFQ) framework which is provided by Tensorflow 2 for quantum machine learning applications was used. The TFQ framework is based on the Cirq quantum library which is Google's python library for writing, manipulating and optimizing quantum circuits.

The classifier based on quantum machine learning is derived from Tensorflow2's ex-

ample classifier, where it is used to classify digits from the MNIST dataset [22]. Due to the current limitations of quantum computers, the MNIST classifier was restricted to classifying only 2 digits (namely 3 and 6). Similarly, deu to the restrictions on the number of qubits, the MNIST dataset had to be heavily modified in order to feed it to the quantum network. The approach itself is based on [15], which uses one qubit to represent one pixel of the image. Currently, only a few qubits can be simulated and run. Therefore, the image data itself was resized down to just 4x4 pixels to get a reasonable number of qubits.

The approach to modulation classification in a quantum setting is similar to Tensorflow2's example, where spectrogram images of the signals are used for classification. Similar to the MNIST classification example, only 2 classes of modulations namely BPSK and QPSK, are used.

4.3.4.1. Dataset pre-processing

A good amount of data pre-processing was needed before the RadioML-2016 dataset could be successfully trained on a Quantum Neural Network (QNN). The inputs to the QNN are in the form of quantum circuits which are designed according to the input data. The input data is in the form of a 4x4 binary image, i.e, an image containing only 0s and 1s. Each pixel is represented by one single qubit and the value of the pixel (0 or 1) is used to design the circuit for that qubit. The approach in the Tensorflow2's example is used to prepare the data for the QNN.

First of all, the spectrogram images of BPSK and QPSK modulations were generated from the IQ samples of the signals in the RadioML-2016 dataset. The spectrogram images were then resized to a 4x4 image, which was then converted into a binary image by using a certain threshold. The full approach is given in algorithm 4.2.

Algorithm 4.2 Dataset Preparation for Quantum Neural Network

Input: Vector of IQ signals $\mathbf{x}$ of length L and signal labels $\mathbf{y}$

Output: Vector of binary images $\mathbf{t}$ of size 4×4

procedure PREPARE_DATA($\mathbf{x}$)
  for $i = 1$ to L **do**
    $\boldsymbol{s}_i = \text{SPECTROGRAM}(x_i)$
    $\boldsymbol{s}_i = 10 \log_{10} (\boldsymbol{s}_i)$
    $\boldsymbol{r}_i = \text{RESIZE}(\boldsymbol{s}_i)$
  end for
  $\boldsymbol{t} = \text{REMOVE\_CONTRADICTIONS}(\boldsymbol{r}, \boldsymbol{y})$
  for $i = 1$ to $\text{LENGTH}(\boldsymbol{t})$ **do**
    for every pixel p in $\boldsymbol{t}_i$ **do**
      if $p >$ THRESHOLD **then**
        $p = 1$
      else
        $p = 0$
      end if
    end for
  end for
  return $\boldsymbol{t}$
end procedure

In algorithm 4.2, each element in the input vector $\mathbf{x}$ is a complex vector of IQ samples. The procedure SPECTROGRAM returns the 2D spectrogram of a complex signal. The values of spectrogram are converted to decibel (dB) scales to get more reasonable values. The images are then resized by the procedure RESIZE to 4x4 images. The contradictory images are then removed from the set of resized images. Here, a contradictory image has the same values for all its pixels but belongs to a different class. Contradictory images become quite common when the number of pixels in the image are very low and must be removed as they can degrade the training performance.

Once the contradictory images have been removed, we apply a threshold to the pixel values. Any pixel value greater than the THRESHOLD will get a value of 1 otherwise it gets a value of 0.

Each of the binary images is then converted into a quantum circuit. Algorithm 4.3 details this procedure. Here, we generate one qubit for each pixel in an image. If the pixel has the value 1, we pass the qubit associated with the pixel through an X gate, otherwise the state of the qubit remains unchanged. The output after this state will be a vector of quantum circuits for each binary image.

Algorithm 4.3 Conversion to Quantum circuit from binary image

Input: Vector of binary images $\mathbf{t}$ of length T

Output: Vector of quantum circuits $\boldsymbol{C}$

procedure CONVERT_TO_CIRCUIT($\boldsymbol{t}$)
    for $i = 1$ to T **do**
        $\mathcal{R} =$ QUANTUM_CIRCUIT()
        for every pixel p in $\boldsymbol{t}_i$ **do**
            $q =$ NEW_QUBIT()
            if $p = 1$ **then**
                CIRCUIT_ADD($\mathcal{R}, \mathcal{X}(q)$)
            end if
        end for
        $\boldsymbol{C}_i = \mathcal{R}$
    end for
    return $\boldsymbol{t}$
end procedure

An example of this conversion of binary image to quantum circuit is given in Figure 4.7.

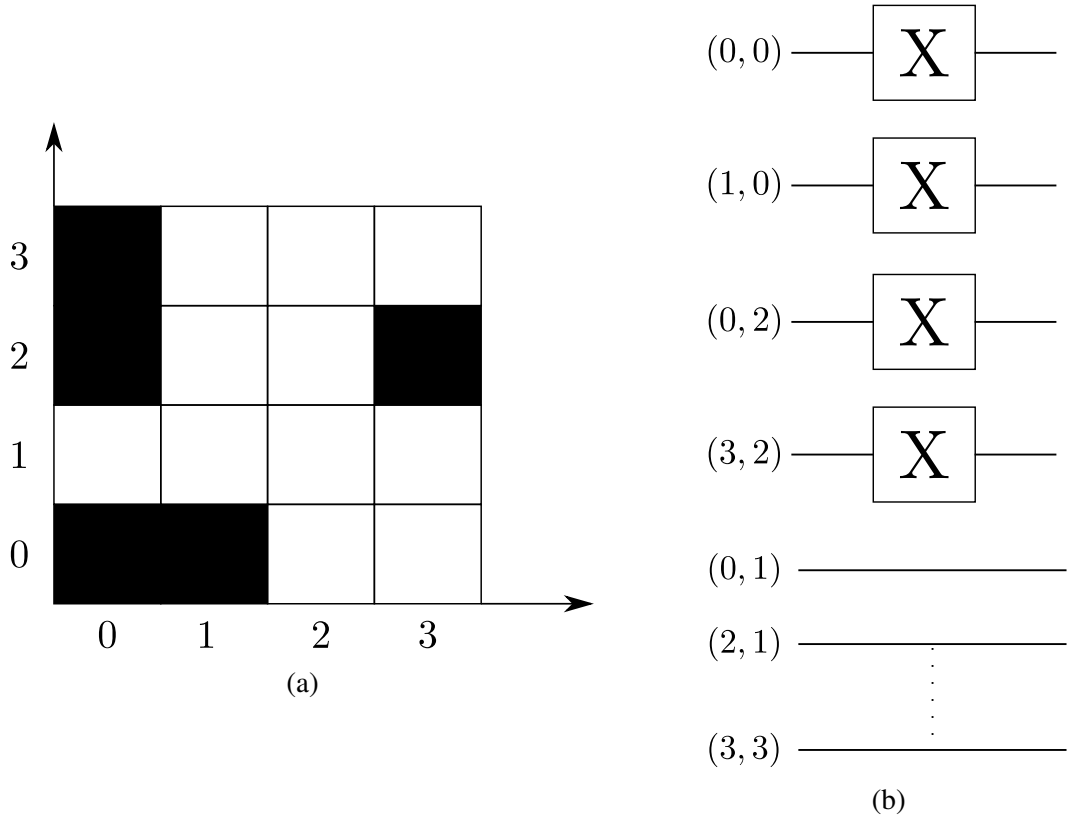


Figure 4.7: Conversion of binary image (a) to its quantum circuit equivalent given in (b). The black pixels represent a pixel value of 1.

4.3.4.2. Construction of Quantum Neural Network

The quantum neural network will take an input of 16 quantum circuits (since we have 4x4 = 16 pixels) generated from algorithm 4.3. An extra qubit, called the readout qubit, is also added to the network. The readout qubit acts as the output qubit of the quantum neural network.

The QNN consists of a set of 16 gates. Each of these gates take 2 qubits as input. One of the inputs is the readout bit and the other input is one of the 16 different qubits that are input to the neural network. The gates are also parameterized, i.e, there is a certain variable that will control the operation of that gate. It is through these parameters that the network is trained. The powered parity gates are used as the layer gates in the network. The powered parity gates are two input parity gates that are raised to a certain power, which acts as the parameters in our circuit. For example, the powered X parity gate, denoted by XX is given by the unitary matrix in 4.10.

$$(XX)^t = \begin{bmatrix} c & 0 & 0 & s \\ 0 & c & s & 0 \\ 0 & s & c & 0 \\ s & 0 & 0 & c \end{bmatrix} \tag{4.10}$$

where,

$$c = f \cos\left(\frac{\pi t}{2}\right)$$

$$s = -je^{j\pi/2} \sin\left(\frac{\pi t}{2}\right)$$

$$f = e^{j\pi t/2}$$

Two layers of XX and ZZ gates are used in the network, along with a combination of Pauli X and Hadamard gates for the readout qubit. A sample figure of such a network having only 2 inputs is shown in Figure 4.8.

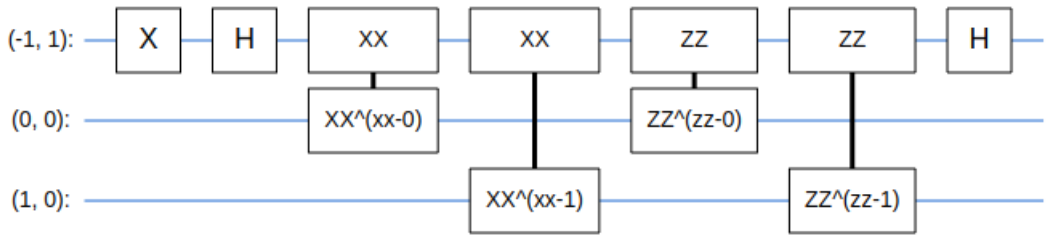


Figure 4.8: The QNN having 2 inputs for demonstration purposes

The qubit $(-1, 1)$ represents the readout qubit and `xx-0`, `xx-1`, `zz-0`, `zz-1` act as the learnable parameters of this particular neural network. The actual network used will have 16 input circuits and 32 such learnable parameters.

The circuit is then wrapped in a Parameterized Quantum Circuit (PQC) layer of Tensorflow. A measurement operator is also specified for the PQC layer which is used to perform measures on the readout bit. The measurement operator used is the Z measurement gate (also known as Pauli Z gate) given by the unitary matrix in equation 4.11.

$$Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{4.11}$$

The Z operator has two eigenvalues $+1$ and -1 which corresponds to two quantum states, the $|0\rangle$state and the $|1\rangle$. If qubit is measured in $|0\rangle$state then, the value of the operator will be +1 and it will be -1 if the $|1\rangle$state is measured. The PQC layer returns the expected value of all the measurements made which will be a value within the interval $[-1, 1]$. In the context of our classifier, a value of 1 will represent the input belonging to class of BPSK and -1 belonging to the class of QPSK.

Since the output of the network is between -1 and 1, a suitable loss function has to be used. The loss function used in [22] is the hinge loss, which is adopt for our classifier as well. The formula for calculating the hinge loss is given in equation 4.12.

$$L_H(\mathbf{y}, \hat{\mathbf{y}}) = \sum \max(0, 1 - \mathbf{y}\hat{\mathbf{y}}) \tag{4.12}$$

where, the vector $\mathbf{y}$ are the actual labels of the input data and $\hat{\mathbf{y}}$ are the predicted labels. The network is only trained for only 5 epochs as the size of the dataset used in training is rather small.

5 RESULTS AND ANALYSIS

5.1. Likelihood based classifier

5.1.1. Generated Signal

Signals of varying SNR were generated using the GENERATE_SIGNAL procedure in algorithm 4.1 with $m =$ BPSK or QPSK, $K = 100$, and the SNR parameter γ varying from -8 dB to 8 dB with a step size of 2. The noise variance parameter(N), was taken to be 1. For each SNR level, 1000 signals were generated from which the accuracy was calculated. The constellation plot for the generated BPSK and QPSK signals are given in figure 5.1 and 5.2 respectively.

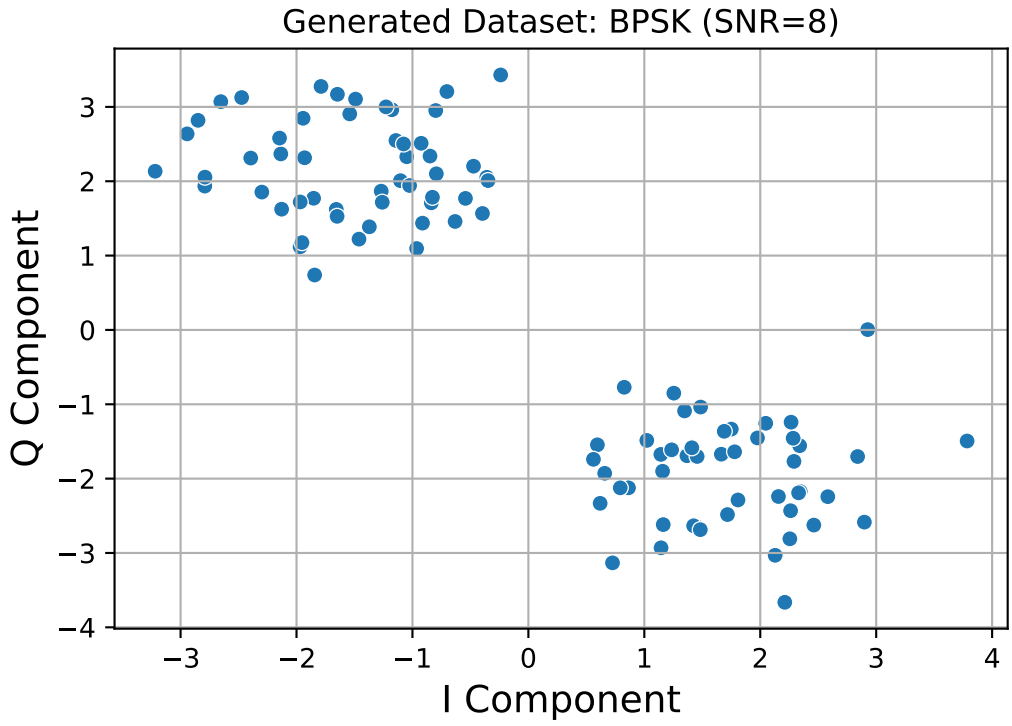


Figure 5.1: Generated BPSK signal with SNR of 8 dB.

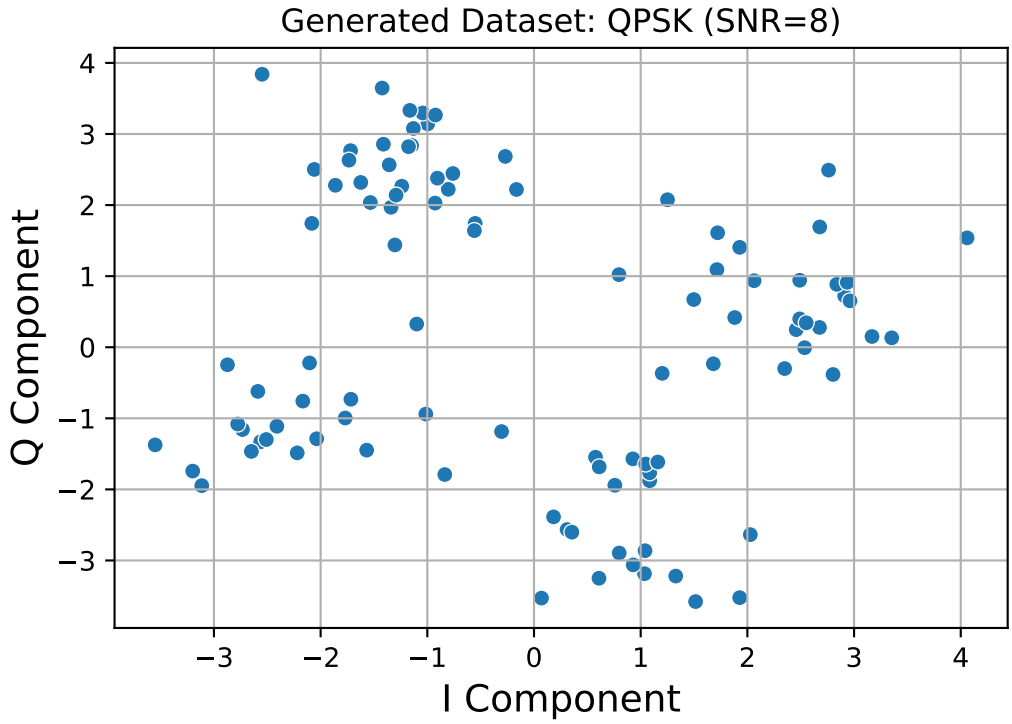


Figure 5.2: Generated QPSK signal with SNR of 8 dB.

5.1.2. Evaluation framework for ALRT system

We need some sort of evaluation metric to see the performance of our system and also to compare the results at various levels of SNRs.

Since the overall system itself is custom tailored, with multiple subsystems required to calculate likelihood of the given signal, we used simple accuracy calculation as the evaluation framework of the system.

We pass random signals with modulation label m to the system and count the number of the signals the system rightly classifies as m modulated signals. The accuracy of the system for that specific modulation type, denoted by $\eta^{(m)}$ is calculated as the ratio of rightly classified signals to the total number of signals input.

$$\eta^{(m)} = \frac{n^{(m)}}{N^{(m)}} \tag{5.1}$$

where, $n^{(m)}$ is the number of signals rightly classified as m modulated by the system and $N^{(m)}$ is the total number of m modulated signals passed to the system The overall accu-

racy of the system (η) is calculated by taking average of modulation specific accuracy over all the available modulation types, L.

$$\eta = \frac{1}{L}\sum_{i=1}^{L}\eta^{(i)} \tag{5.2}$$

5.1.3. Results

The accuracy plot of the ALRT classifier for BPSK and QPSK modulation for various SNR levels is shown in figure 5.3. Here, the accuracy of the classifier gradually increases as we increase the SNR of the generated signal. Above SNR of 0 dB, the ALRT gives almost perfect accuracy, correctly classifying almost all the signals.

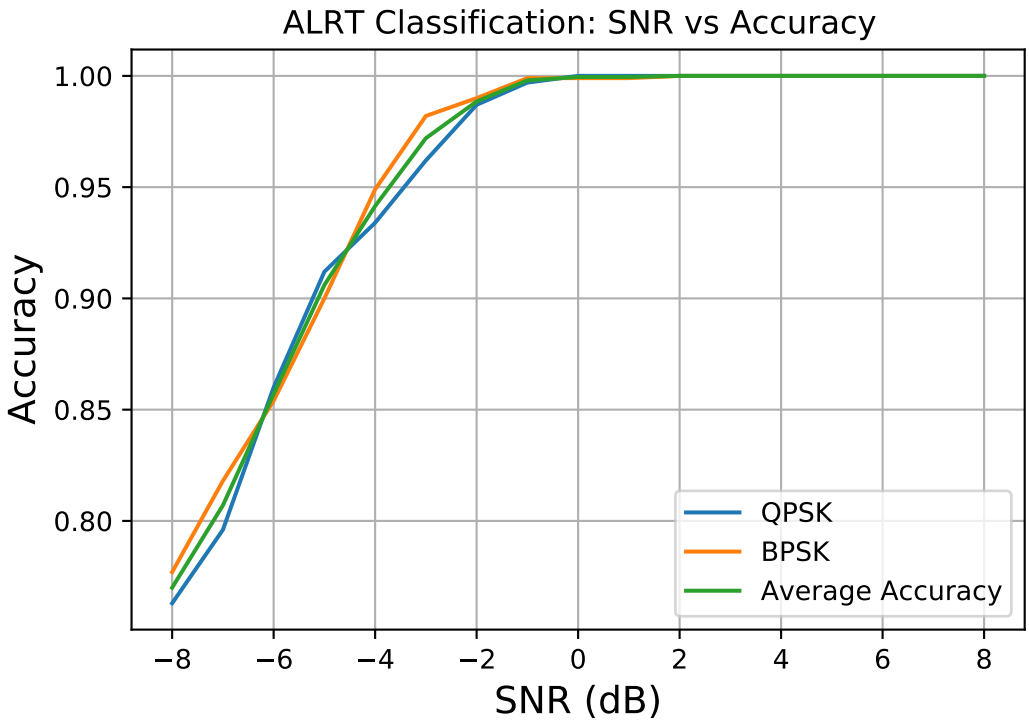


Figure 5.3: SNR vs. Accuracy plot for the ALRT classifier

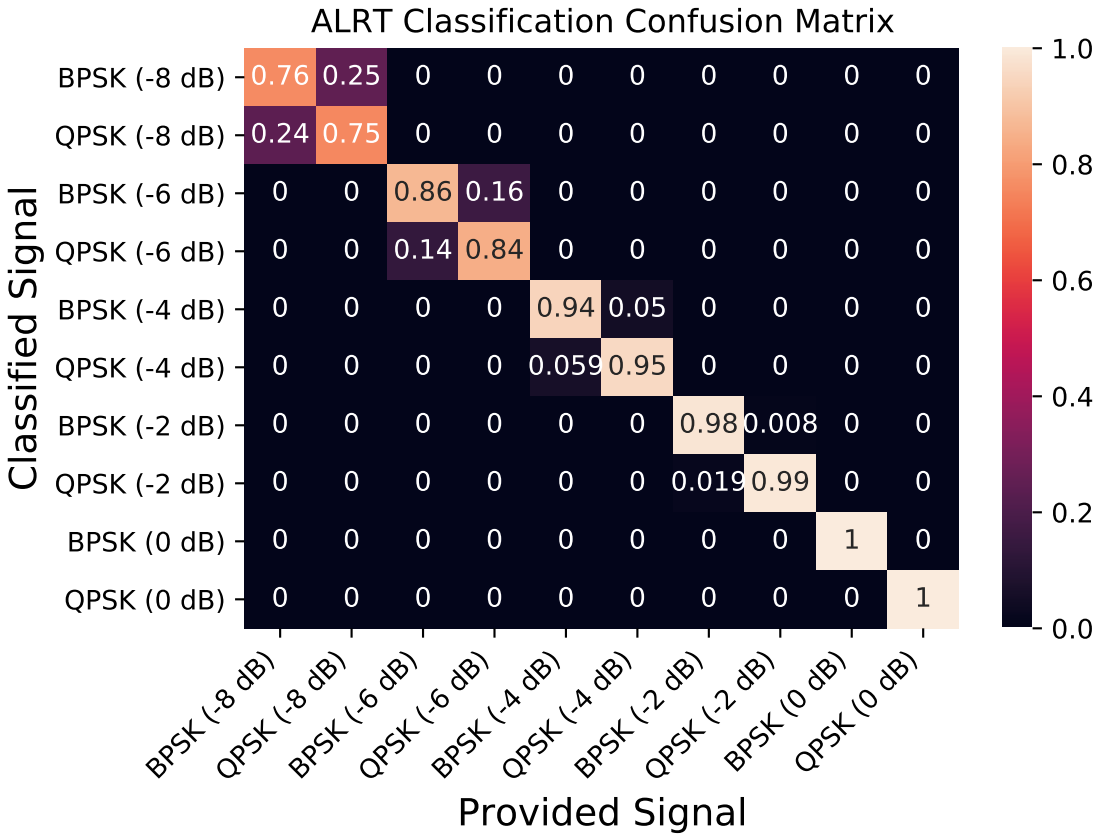


Figure 5.4: Confusion matrix for the ALRT classifier

5.2. Deep Learning Based classifiers

5.2.1. Evaluation Framework

The usual evaluation frameworks used for DNN are employed to measure and evaluate the performance of DL based classifiers. These include the epochs vs loss plot and confusion matrices. Specifically, we use two kinds of confusion matrices to evaluate our classifier. One is the overall confusion matrix which is generated by considering the accuracy of our classifier over all the signals over all the possible SNRs. Mathematically, the elements of this confusion matrix $\mathcal{P}$ are given by:

$$\mathcal{P}_{ij} = \Pr\left(\hat{y}_i = \mathcal{M}_i | \boldsymbol{x_i} = \mathcal{M}_j\right) = \frac{n_i^{(\mathcal{M}_j)}}{N^{(\mathcal{M}_j)}} \tag{5.3}$$

where, $\hat{y}_i$ is the predicted modulation type for an input signal $\boldsymbol{x_i}$. $\mathcal{M}$ is the set of all modulations that are classified by our model. Here, $n_i^{(\mathcal{M}_j)}$ is the number of signal classified as modulation $\mathcal{M}_i$ which have a modulation of $\mathcal{M}_j$ and $N^{(\mathcal{M}_j)}$ is the total number of signal with actual modulation $\mathcal{M}_j$.

The overall accuracy (η) of the models is calculated by taking the ratio of the number of correctly classified signals to the total number of signals.

$$\eta = \frac{\sum_i n_i^{(\mathcal{M}_i)}}{\sum_i N^{(\mathcal{M}_i)}} \tag{5.4}$$

Here, $n_i^{(\mathcal{M}_i)}$ is the total number of correctly classified signals of modulation $\mathcal{M}_i$ and $N^{(\mathcal{M}_i)}$ is the total number of signals belonging to modulation $\mathcal{M}_i$.

The second confusion matrix is generated by considering the accuracy of our classifier over signal having a specific value of SNR. Mathematically, the elements of this confusion matrix $\mathcal{Q}_{ij}^{s_k}$ is given by:

$$\mathcal{Q}_{ij}^{(s_k)} = \Pr\left(\hat{y}_i = \mathcal{M}_i | \boldsymbol{x_i} = \mathcal{M}_j \text{and SNR}(\boldsymbol{x_i}) = s_k\right) \tag{5.5}$$

Finally, we have the SNR vs Accuracy plot which gives us the accuracy of our classifier with respect to the SNR level of the incoming signal. It is simply the plot of overall accuracy given by the second confusion matrix and its SNR level. The probability of correctly classifying the input signal of a certain modulation type is the accuracy for that modulation type for a given SNR. Mathematically, the accuracy of classification of modulation $\mathcal{M}_\rangle$ for a given SNR s_k is:

$$\text{Acc}(s_k, \mathcal{M}_i) = \Pr\left(\hat{y}_i = \mathcal{M}_i | \boldsymbol{x_i} = \mathcal{M}_i \text{ and SNR}(\boldsymbol{x_i}) = s_k\right) \tag{5.6}$$

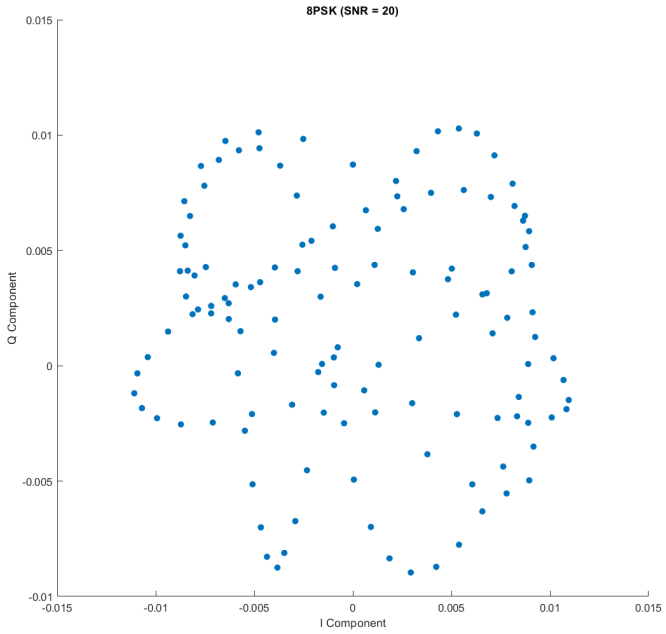
The total accuracy for a given SNR level is calculated by taking the ratio of number of correctly classifier inputs to total number of signals.

$$\text{Acc}(s_k) = \frac{\sum_i n_i^{(\mathcal{M}_i, s_k)}}{\sum_i N^{(\mathcal{M}_i, s_k)}} \tag{5.7}$$

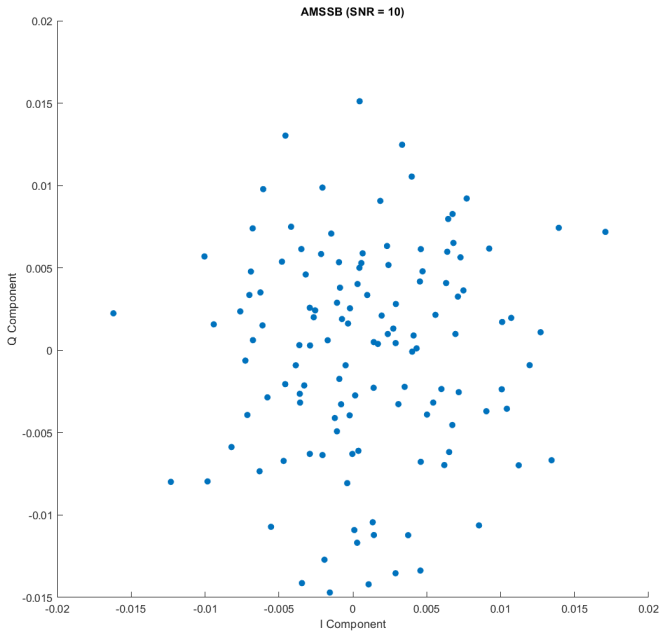
where $n_i^{(\mathcal{M}_i, s_k)}$ is the number of signals of modulation $\mathcal{M}_i$ classified correctly of a given SNR s_k and $N^{(\mathcal{M}_i, s_k)}$ is the number of signals of modulation $\mathcal{M}_i$ of a given SNR s_k.

5.2.2. Dataset

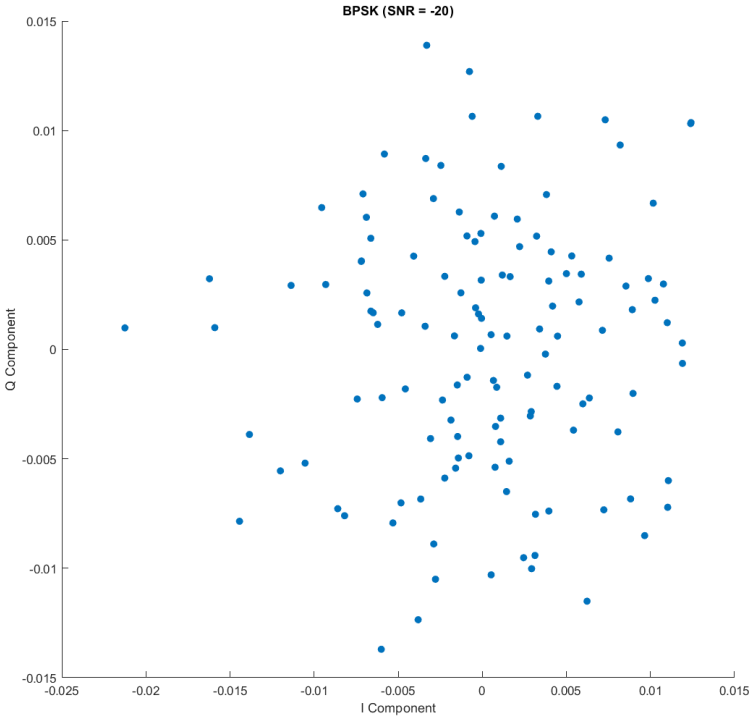
Some of the signal samples from the RadioML dataset were visualized as a constellation diagram. The results are given in Figure 5.5.



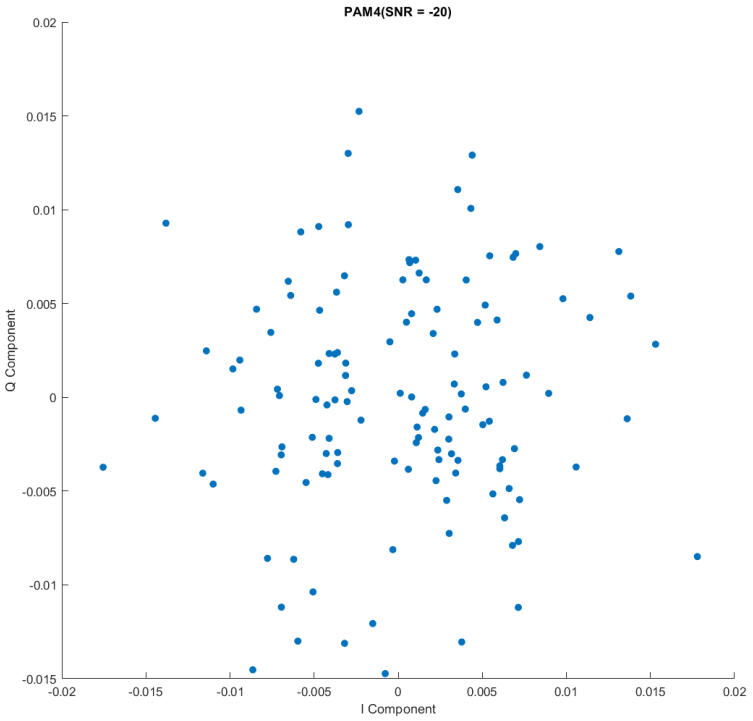
(a) 8PSK with SNR = 20 dB



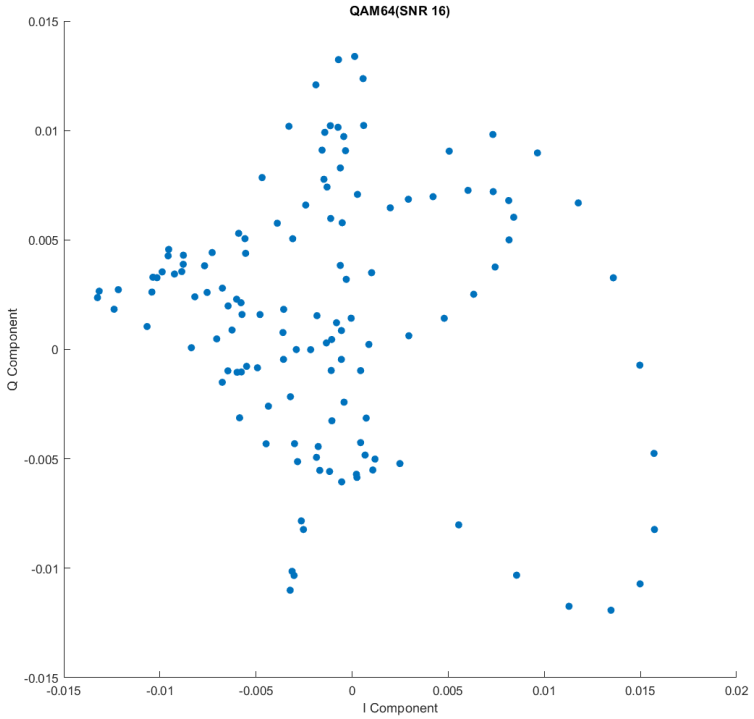
(b) AMSSB with SNR = 10 dB



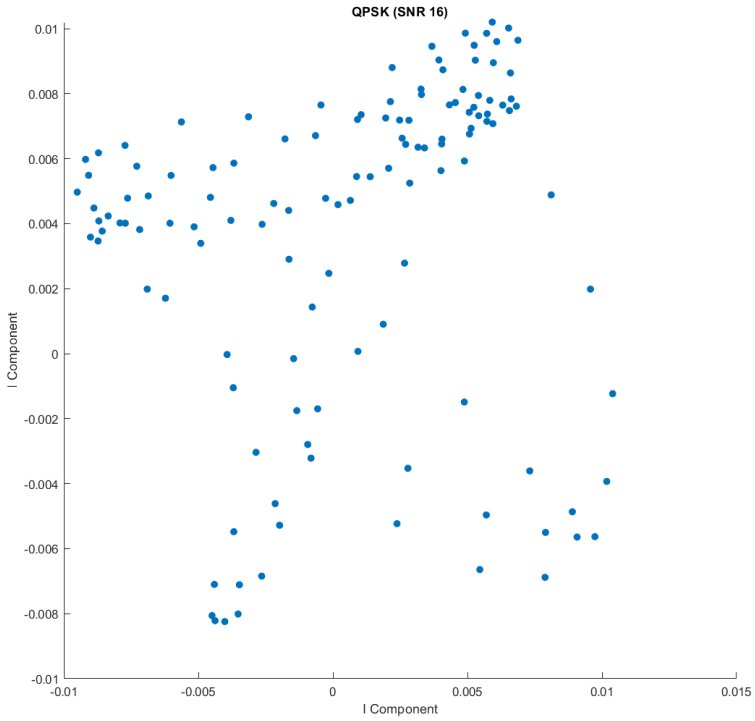
(c) BPSK with SNR = -20 dB



(d) PAM4 with SNR = -20 dB



(e) QAM64 with SNR = 16 dB



(f) QPSK with SNR = 16 dB

Figure 5.5: Plot of various signals contained in the RadioML dataset.

5.2.3. Results: CNN based classifiers

The training performance of RadioML CNN network and modified CNN is displayed in the Figure 9.1 and Figure 9.2 respectively. The SNR(dB) vs Accuracy Plot is also given below which shows that the accuracy of the model increases as the SNR level increases and tends to constant at positive SNRs. The overall accuracy of these models are $\eta_L = 51.31\%$ and $\eta_L = 54.84\%$ of RadioML CNN and modified CNN respectively.

This model somewhat performs poorly for QAM and AM-SSB modulation types. The confusion matrix for these models is shown in Figure 5.6 and Figure 5.8 respectively. Also the confusion matrix for these two CNN based models for different SNR levels are given in the Figure 10.1 and Figure 10.2.

Table 5.1: Accuracy for different SNR level for RadioML CNN network

SNR Level (dB)	**Accuracy**(%)
-10	22.51
-8	33.896
0	68.65
4	74.73
16	77.00
18	75.96

Figure 5.6: The overall confusion matrix for the RadioML CNN network.

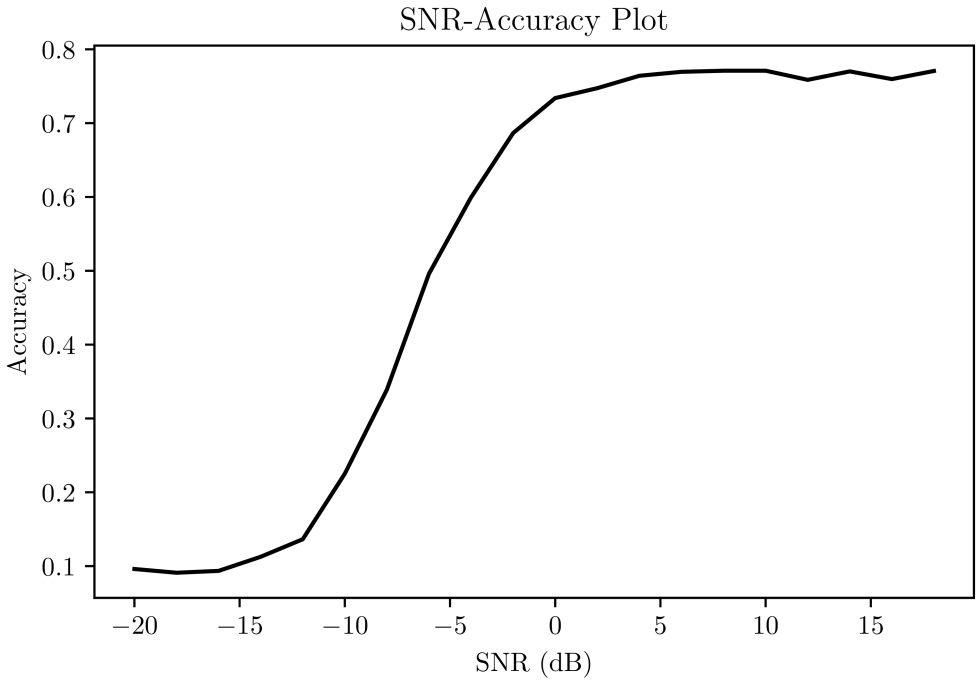


Figure 5.7: Plot of SNR (dB) vs Accuracy for the RadioML CNN network.

Table 5.2: Accuracy for different SNR level for Modified CNN network

SNR Level (dB)	**Accuracy**(%)
-10	19.36
-8	31.18
0	72.44
4	78.36
16	82.59
18	80.95

Figure 5.8: The overall confusion matrix for the Modified CNN network.

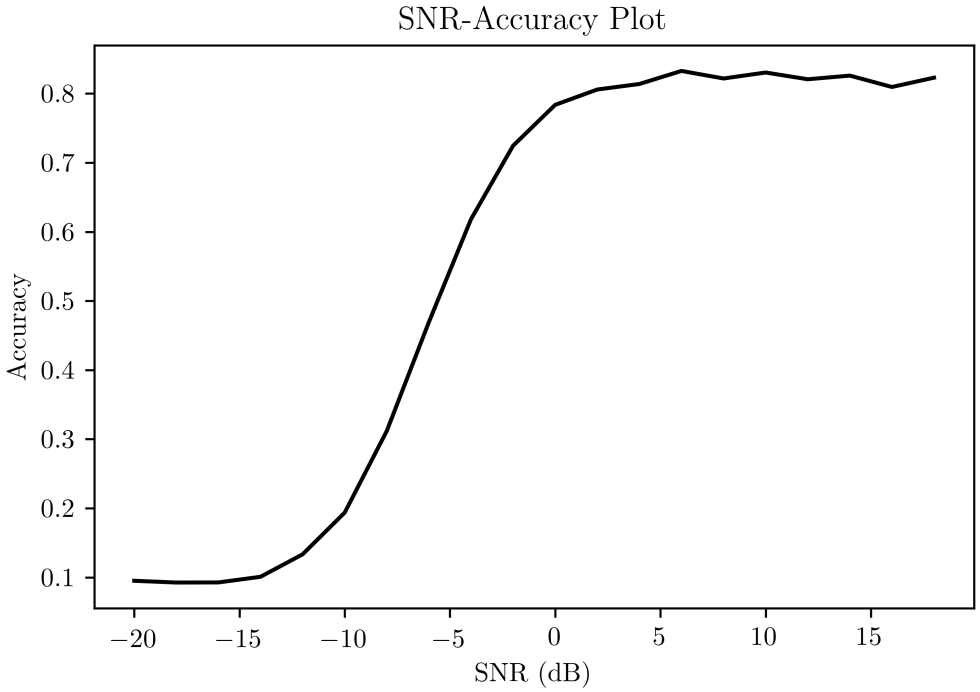


Figure 5.9: Plot of SNR (dB) vs Accuracy for the Modified CNN network.

5.2.4. Results: RNN based classifiers

5.2.4.1. LSTM based classifier

Google Colab and Tensorflow was used to train and validate the LSTM and Bi-LSTM based models. The training performance of our LSTM based network is given in Figure 9.3. Here, we see that the training loss is gradually decreasing and finally settling to a value of 1.64683. The early stopping employing during training then stopped further training on the network. The overall confusion matrix is shown in Figure 5.10. The overall accuracy of this model is $\eta_L = 37.33\%$.

The LSTM only classifier performs somewhat poorly and has a tendency to classify the input signals as AM-SSB. There is a very high degree of confusion for WBFM modulation type. The confusion matrices for various SNR levels results are given in Figure 10.3. Here, we see that the performance of our classifier gradually improves with the SNR level as the diagonal of the confusion matrix becomes more and more prominent with the increasing SNR level.

Finally, we have the SNR vs Accuracy plot given in Figure 5.11. For lower SNRs,

the classifier is very inaccurate. However, the accuracy is greatly increased for higher SNR levels. Some of the values of accuracy for SNR levels are given in Table 5.3.

Table 5.3: Accuracy for different SNR level for LSTM based network

SNR Level (dB)	**Accuracy**(%)
-10	15.70
-8	20.92
0	47.76
4	53.72
16	54.29
18	54.49

Figure 5.10: The overall confusion matrix for the LSTM based network

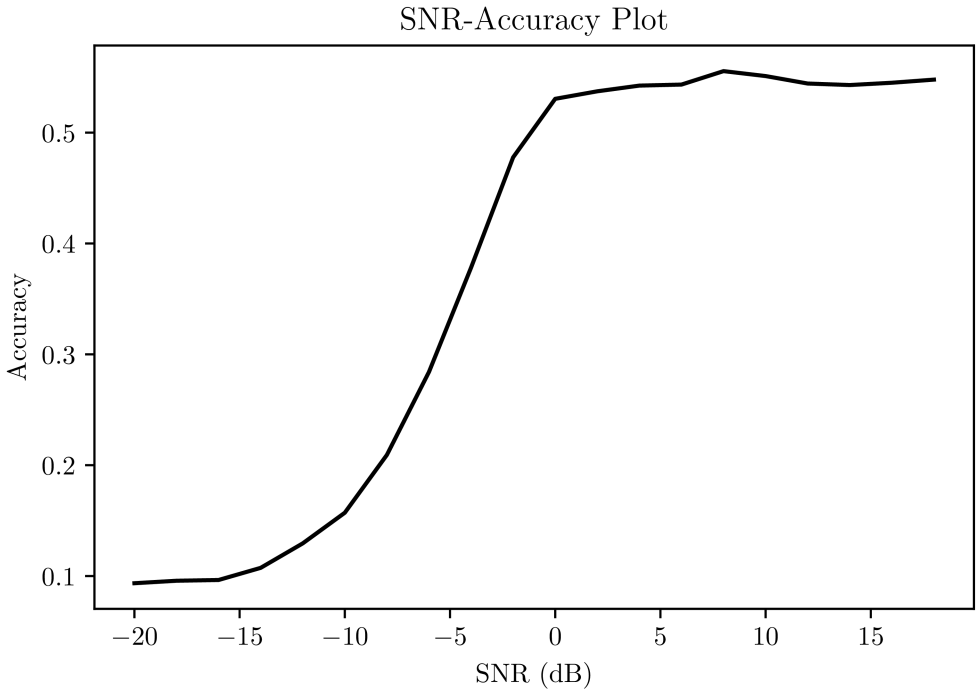


Figure 5.11: Plot of SNR (dB) vs Accuracy for the LSTM based network

5.2.4.2. LSTM with attention layer

The training performance of the LSTM based network with attention layer is given in Figure 9.4. Here, the training performance curve decreases more sharply than previous LSTM model. The value of loss of our network is also lower, about 1.5. The overall confusion matrix and the confusion matrices for different SNR levels are given in Figure 5.12 and Figure 10.4 respectively. The overall accuracy of this model is found to be $\eta_{LA} = 46.68\%$. The overall confusion matrix shows that this model is much more accurate than the previous LSTM model. There is still a large amount of confusion for the modulation types of QAM16, QAM64, QPSK and WBFM, as noticed by the larger spread of the data over the respective column. The confusion matrices for different SNR levels also tell a similar story, with the classifier becoming more and more accurate as the SNR level increases. The SNR vs Accuracy plot is given in Figure 5.13. The accuracy values for some SNR levels are given in Table 5.4. This model achieves a higher accuracy than the previous model at medium to high values of SNR.

Table 5.4: Accuracy for different SNR level for LSTM based network with attention layer

SNR Level (dB)	**Accuracy**(%)
-10	19.24
-8	23.55
0	53.41
4	66.84
16	71.37
18	69.99

Figure 5.12: The overall confusion matrix for the LSTM based network with attention

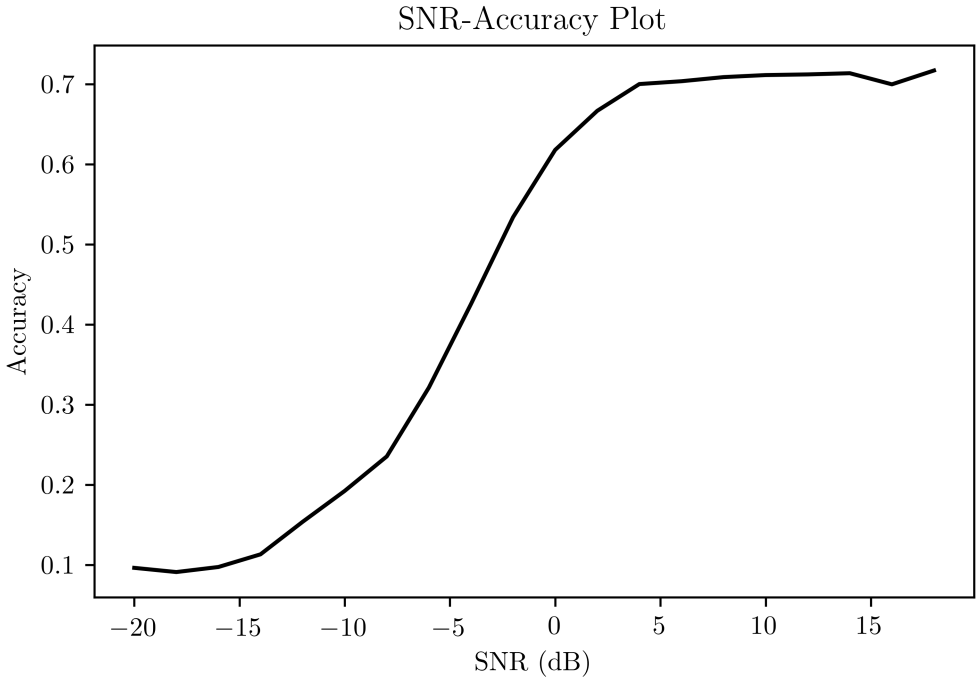


Figure 5.13: Plot of SNR vs Accuracy for the LSTM based network

5.2.4.3. Bi-LSTM network

The training performance, overall confusion matrix and the confusion matrix for different SNR levels for the Bi-LSTM network with attention layer are given in Figure 9.5, Figure 5.14 and Figure 5.15 respectively. We see that the drop in training loss is much steeper than that for previous RNN models. The training loss is also much lower reaching values of about 1.1038. The model is very accurate, having the overall accuracy of $\eta_{Bi} = 56.88\%$. This can also be observed through the confusion matrix where the diagonal values are much higher than that for the previous networks. A similar trend can be observed in the confusion matrices for each SNR level, where the accuracy is very high for high SNR levels. The total accuracy values for some of the SNR levels are given in Table 5.5.

Table 5.5: Accuracy for different SNR level for Bi-LSTM based network with attention layer

SNR Level (dB)	**Accuracy**(%)
-10	25.44
-8	37.46
0	72.06
4	80.93
16	84.22
18	84.12

Figure 5.14: The overall confusion matrix for the Bi-LSTM based network with attention

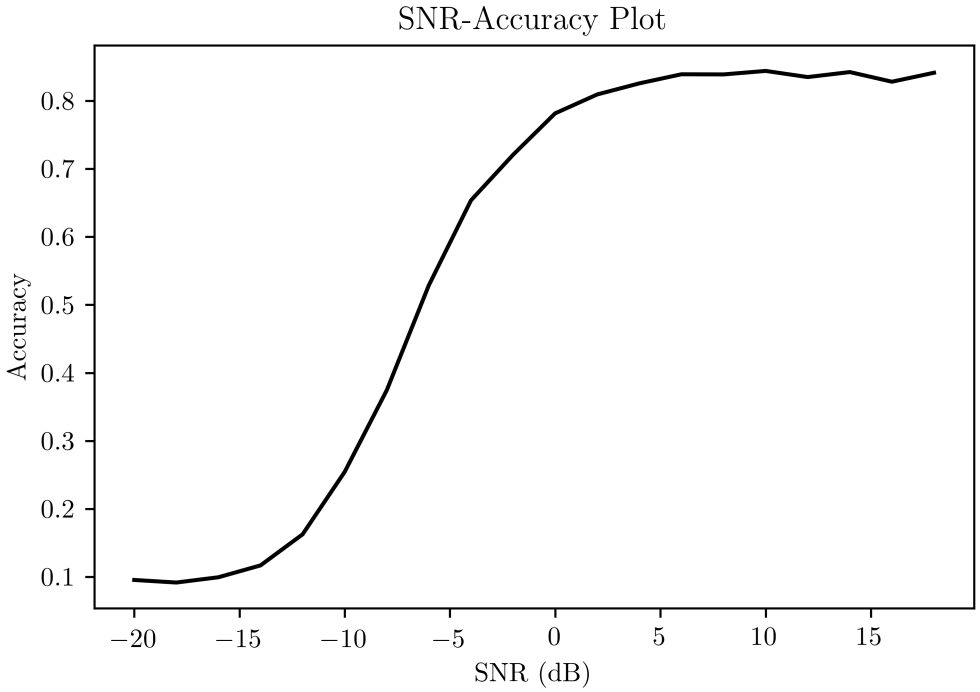


Figure 5.15: Plot of SNR vs Accuracy for the Bi-LSTM based network

5.2.5. Quantum Neural Network based classifier

The QNN was trained in Google colab using Tensorflow2. The quantum circuits were created using Cirq, Google's quantum computing framework. The training performance, confusion matrices and SNR-vs-Accuracy plots are given in figures 9.6, 5.16, 10.6 and 5.17 respectively. Here, we can see that the QNN based classifier performs somewhat poorly in comparison to other classification discussed previously. The total overall accuracy of the QNN based network is only 55.72%. There is also a lot of confusion in classification of BPSK signals as we observe from the confusion matrix that the majority of the BPSK signals are being classified as QPSK signals. The trend in SNR-vs-Accuracy is similar to that of other classifiers with accuracy increasing with increasing SNR. The accuracy values for some of the SNR are given in the table.

Table 5.6: Accuracy for different SNR level for QNN

SNR Level (dB)	**Accuracy**(%)
-2	50.85
0	51.78
2	55.93
4	58.89
8	60.88

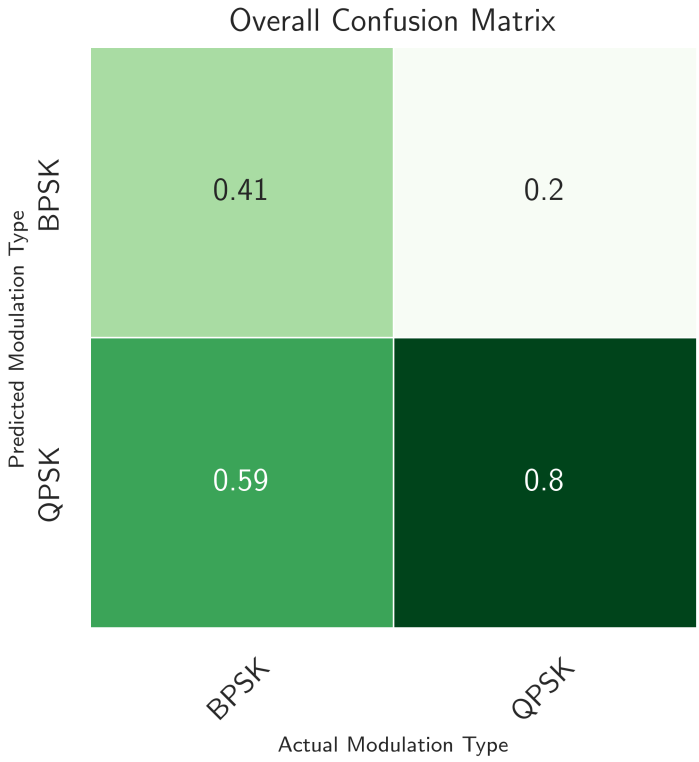


Figure 5.16: The overall confusion matrix for the QNN based classifier.

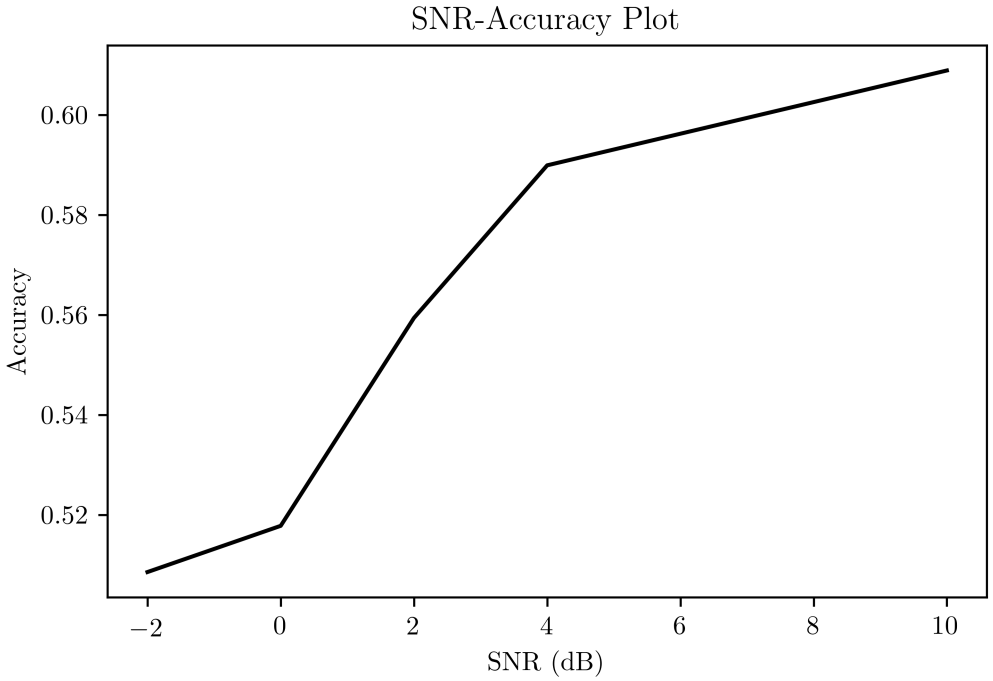


Figure 5.17: Plot of SNR (dB) vs Accuracy for the QNN based classifier.

Though having appreciable accuracy, the resource restriction due to lack of number of available qbits makes QNN not a viable solution of AMC problem; at least for now.

5.2.6. Analysis of Results

A comparison of SNRs of the classifier is given in Figure 5.18. Here, we see that the LSTM network has the least accuracy across all SNRs. The overall accuracy of this model is 37.33%. The LSTM network with an attention layer has better accuracy than the LSTM network whose overall accuracy is found to be 46.68%. The RadioML, Modified CNN, and the Bi-LSTM network provide better accuracy across all the SNRs whose overall accuracy are 51.31%, 54.84% and 56.88% respectively. Besides these, the QNN also has good performance giving an accuracy of 55.72%. The graph below shows the accuracy of all the DL models for different SNR levels.

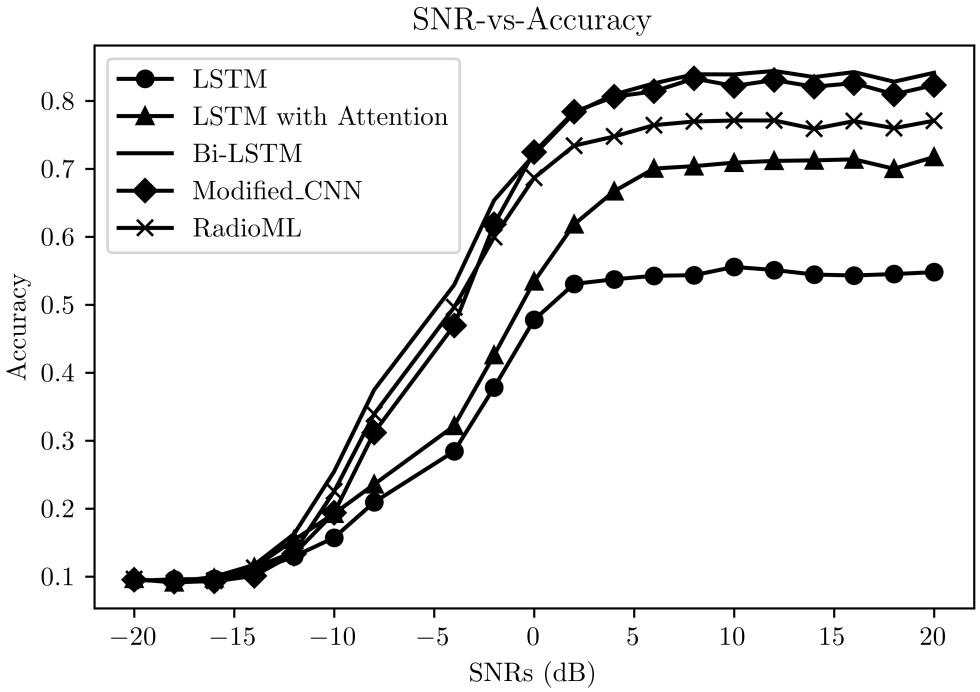


Figure 5.18: Comparison of SNR and Accuracies for different classifier

Among all the approaches implemented, the likelihood ratio test has the highest accuracy of 100% for positive SNRs. This 100% accuracy can be attributed to the perfect knowledge of channel state and the limited modulation scheme used in the input data. However the time and space complexity of this approach is very high due to heavy mathematical computations required for each signal. This fact along with unreal requirement of channel state knowledge make the implementation of ALRT far less practicable than the deep learning-based techniques.

In the modified CNN addition of batch normalization and the removal of softmax activation output layer resulted in higher accuracy of 83.27% (8dB) in comparison to baseline RadioML CNN model with accuray of 77.10% (12dB). This increase in accuracy can be linked to increased generalization and faster convergence due to architectural changes.

In case of RNN, the removal of attention layer resulted in poorer accuracy of 55.54% (10dB) in the LSTM model than the baseline model: LSTM with attention with accuracy of 71.71% (20dB). This decrease in accuracy can be attributed to removal of attention layer due to which all the input samples were accounted equally regardless of their importance.

The Bi-LSTM model drastically increases the accuracy to 84.48% (14 dB) . This increase in accuracy can be attributed to increased number of parameters because of additional LSTM block; which can track the future information (reverse direction in time) in addition

of past information.

We also computed the average time taken to perform prediction by each model whose results are given in Figure 5.19. The tests were all run on Tensor Processing Unit (TPU) provided by Google collab. Here, we see that the prediction time for a single sample is similar across all the models with little variations. The time maximum time taken for the single sample prediction is maximum for the Bi-LSTM model and the fastest among all with the minimum time is by the LSTM with attention layer model.

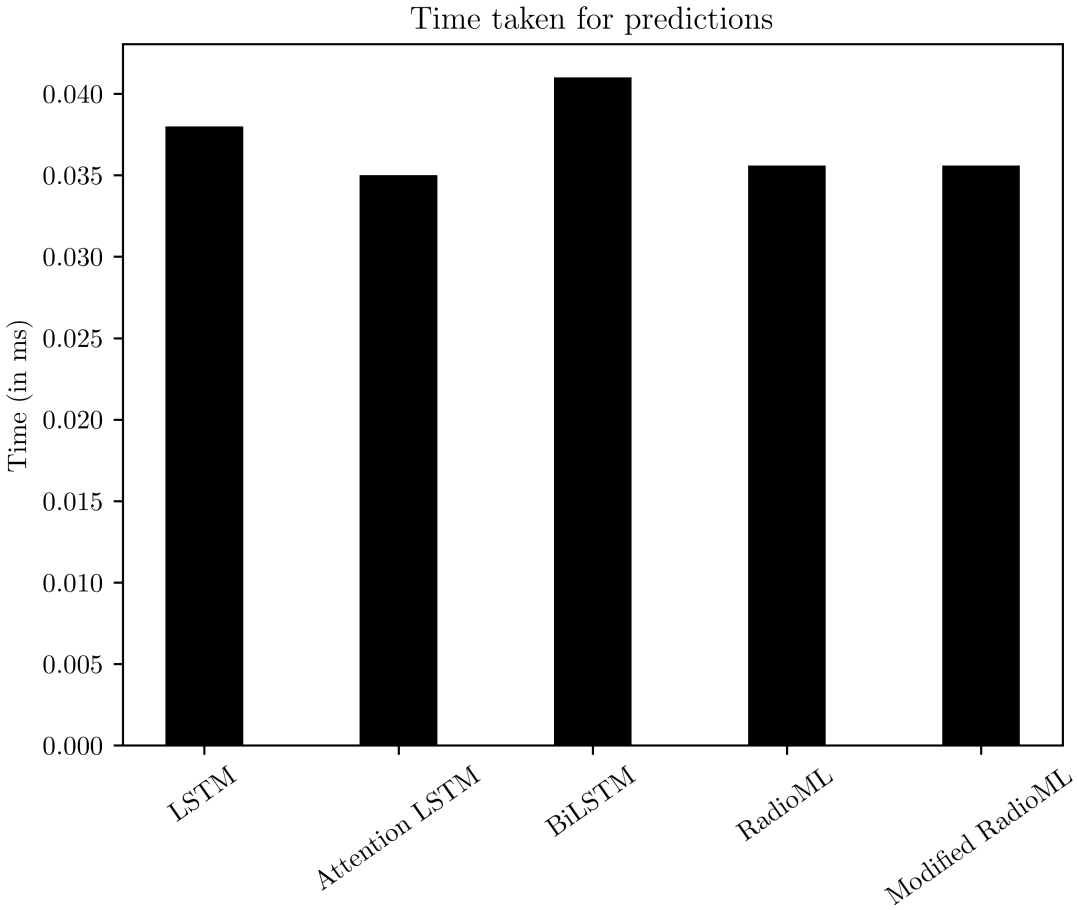


Figure 5.19: Comparison of average time taken to perform a single prediction.

6 CONCLUSION

In this project, we have look at various approaches towards the AMC problem. We performed the classification of various modulation types using various likelihood based classifiers, deep learning based classifiers and even QNN based classifiers. The likelihood based classification was done by computing the likelihood signal and then classifying based on which modulation type has the highest likelihood. The deep learning based classifiers involved passing the signal data through a neural network from which the most probable modulation type is extracted. Different neural network architectures were tested namely the RadioML CNN architecture and its modified version, LSTM networks with/without attention and also a Bi-LSTM network. We also trained and tested a Quantum Neural Network based classifier which was able to classify BPSK and QPSK signals with an accuracy of upto 60.88% at an SNR of 8 dB.

The lack of computation power of present quantum computers make QNN infeasible for AMC solution at present. The likelihood based approach require a perfect knowledge of channel which is infeasible in real life. The QNN and likelihood based approaches are hence not viable solutions of AMC leaving DL based approach as a realistic alternative.

All of the DL based models implemented here were able to produce ac curacies above 50% at SNR levels greater than 2dB. The Bi-LSTM network with attention layer outperforms all the other DL based models with a maximum accuracy of 84.48% (14dB) and not much increase in average prediction time to 0.041 ms for a signal.

7 LIMITATIONS

1. The approaches used in our project show better performance only for higher values of SNR.

2. The optimal model was chosen solely based on the testing performances. The cost and complexity of implementing these models on a lower powered edge devices are not considered. Furthermore, the complexity of architecture will also highly impact the prediction times on those devices. Thus, the chosen model might not be the most optimal when implemented on an edge device.

3. Only number of layers and number of nodes were considered for hyper-parameter tuning.

4. The RadioML dataset used for training and testing the models are synthetic i.e they are generated form software simulation and not from a real world source.

5. Complex architectures like vision transformers, encoder-decoder have not been explored.

8 RECOMMENDATIONS

The main purpose of this project was to explore and evaluate various approaches towards AMC. We implemented some of the suitable solution for of these approaches. We also compared their performances based on model accuracy and prediction times and selected the best model for developing a classifier. Few enhancements and further research that can be done for bettering these results are listed below:

1. Use of a dataset containing collection of real-world received signals, instead of synthesized signals for the training and using more sophisticated hyper-tuning.

2. Reducing time and space complexity of these models to make it feasible for edge devices such as Software Defined Radios (SDRs) in adaptive modulation communication scheme.

3. Preprocessing input data to increase classification accuracy such as conversion to spectrum.

4. We have chosen our optimal classifier purely based upon the testing performance metrics. Alternatively, the optimal classifier can also be chosen by measuring the cost, complexity and performance measurements on edge devices.

5. Various other deep learning models like autoencoders, reinforcement leaning can also be considered for a even more comprehensive study of approaches towards AMC problem.

6. The possibilities of using other types of parameterized gates for encoding qubits with acutal values of spectrogram instead of 0s and 1s can be further explored. Similarly, other types of layer gates can also be experimented on to achieve different results.

REFERENCES

[1] Z. Zhu and A. Nandi, *Automatic Modulation Classification: Principles, Algorithms and Applications*. Wiley, 2015.

[2] O. Dobre, A. Abdi, Y. Bar-Ness, and W. Su, "Survey of automatic modulation classification techniques: classical approaches and new trends," *IET Communications*, vol. 1, no. 2, p. 137, 2007.

[3] A. Polydoros and K. Kim, "On the detection and classification of quadrature digital modulations in broad-band noise," *IEEE Transactions on Communications*, vol. 38, no. 8, p. 1199–1211, 1990.

[4] A. Ali and F. Yangyu, "Automatic modulation classification using principle composition analysis based features selection," *2017 Computing Conference*, 2017.

[5] D. H. Al-Nuaimi, M. F. Akbar, L. B. Salman, I. S. Z. Abidin, and N. A. M. Isa, "Amc2n: Automatic modulation classification using feature clustering-based two-lane capsule networks," *Electronics*, vol. 10, no. 1, p. 76, 2021.

[6] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," *arXiv preprint arXiv:1602.04105*, 2016.

[7] T. Huynh-The, Q.-V. Pham, T.-V. Nguyen, T. T. Nguyen, R. Ruby, M. Zeng, and D.-S. Kim, "Automatic modulation classification: A deep architecture survey," *IEEE Access*, vol. 9, pp. 142950–142971, 2021.

[8] S. Hu, Y. Pei, P. P. Liang, and Y.-C. Liang, "Robust modulation classification under uncertain noise condition using recurrent neural network," in *2018 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–7, IEEE, 2018.

[9] T. J. O'Shea and N. West, "Radio machine learning dataset generation with gnu radio," *Proceedings of the 6th GNU Radio Conference*, 2016.

[10] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th annual symposium on foundations of computer science*, pp. 124–134, Ieee, 1994.

[11] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pp. 212–219, 1996.

[12] J. Biamonte, P. Wittek, N. Pancotti, P. Rebentrost, N. Wiebe, and S. Lloyd, "Quantum machine learning," *Nature*, vol. 549, no. 7671, pp. 195–202, 2017.

[13] J. Adcock, E. Allen, M. Day, S. Frick, J. Hinchliff, M. Johnson, S. Morley-Short, S. Pallister, A. Price, and S. Stanisic, "Advances in quantum machine learning," *arXiv preprint arXiv:1512.02900*, 2015.

[14] M. Broughton, G. Verdon, T. McCourt, A. J. Martinez, J. H. Yoo, S. V. Isakov, P. Massey, R. Halavati, M. Y. Niu, A. Zlokapa, *et al.*, "Tensorflow quantum: A software framework for quantum machine learning," *arXiv preprint arXiv:2003.02989*, 2020.

[15] E. Farhi and H. Neven, "Classification with quantum neural networks on near term processors," *arXiv preprint arXiv:1802.06002*, 2018.

[16] S. S. Haykin and M. Moher, *Introduction to analog and digital communications*. Wiley, 2 ed., 2007.

[17] J. Proakis and M. Salehi, *Digital Communications*. McGraw-Hill, 2008.

[18] W.-k. Lu and Q. Zhang, "Deconvolutive short-time fourier transform spectrogram," *IEEE Signal Processing Letters*, vol. 16, no. 7, pp. 576–579, 2009.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[20] "Convolutional neural network."

[21] F. Hameed, O. A. Dobre, and D. C. Popescu, "On the likelihood-based approach to modulation classification," *IEEE Transactions on Wireless Communications*, vol. 8, no. 12, pp. 5884–5892, 2009.

[22] "Mnist classification — tensorflow quantum."

[23] A. Ali, F. Yangyu, and S. Liu, "Automatic modulation classification of digital modulation signals with stacked autoencoders," *Digital Signal Processing*, vol. 71, p. 108–116, 2017.

[24] Y. Xu, D. Li, Z. Wang, Q. Guo, and W. Xiang, "A deep learning method based on convolutional neural network for automatic modulation classification of wireless signals," *Wireless Networks*, vol. 25, no. 7, p. 3735–3746, 2018.

[25] Y. Wang, J. Yang, M. Liu, and G. Gui, "Lightamc: Lightweight automatic modulation classification via deep learning and compressive sensing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 3, pp. 3491–3495, 2020.

[26] E. Biglieri, J. Proakis, and S. Shamai, "Fading channels: information-theoretic and communications aspects," *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2619–2692, 1998.

[27] M. Abdel-Moneim, W. El-Shafai, N. El-Salam, E.-S. El-Rabaie, and F. Abd El-Samie, "A survey of traditional and advanced automatic modulation classification techniques, challenges and some novel trends," *International Journal of Communication Systems*, p. 2, 01 2021.

APPENDIX A: TRAINING AND VALIDATION LOSSES

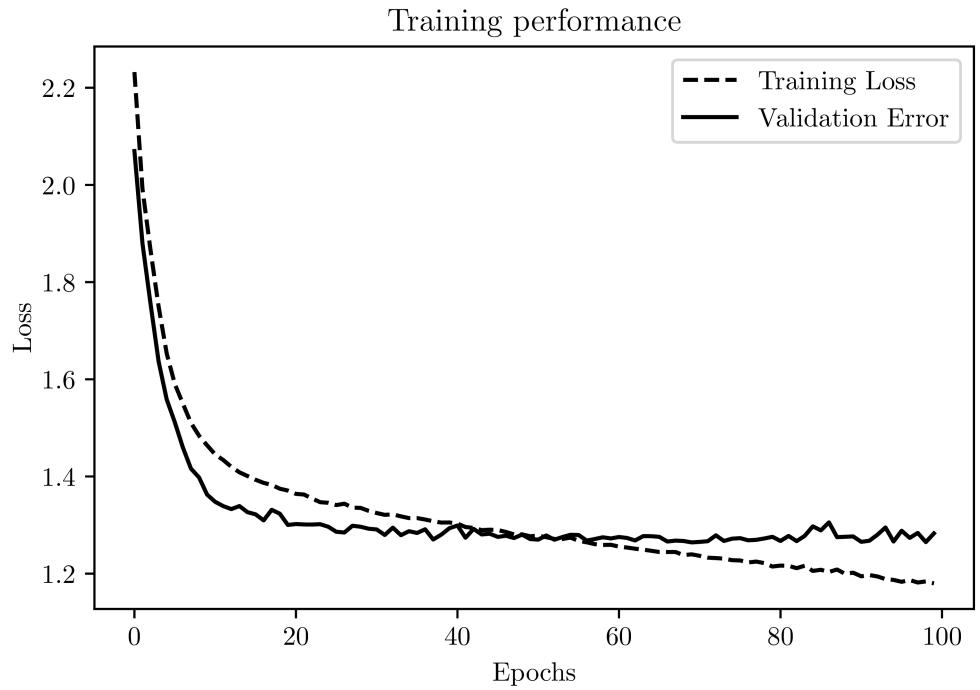


Figure 9.1: Training and Validation Loss for the RadioML CNN network.

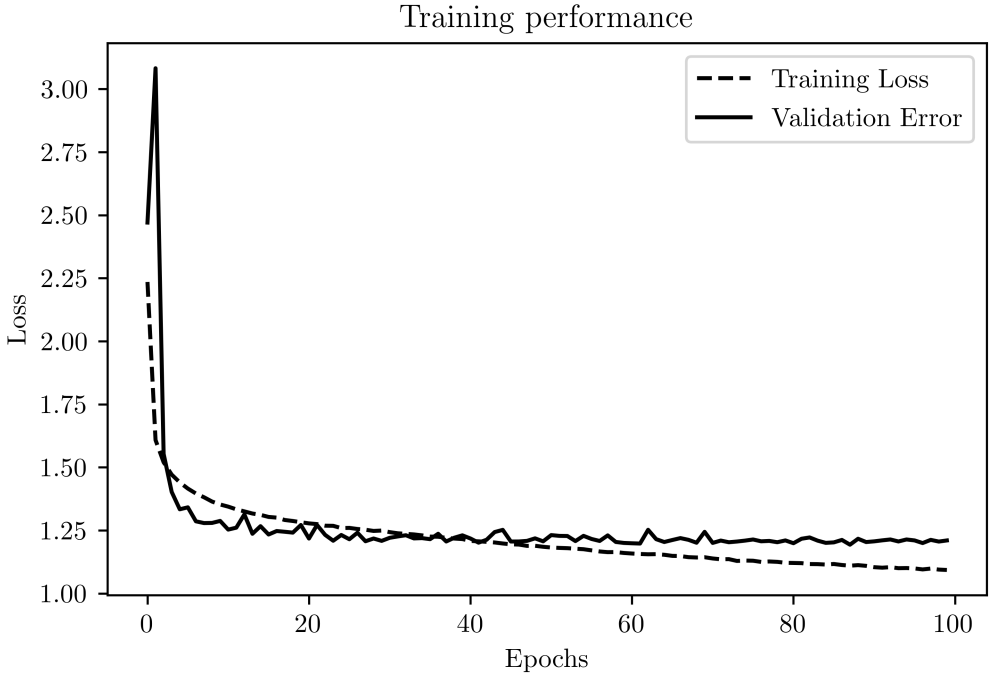


Figure 9.2: Training and Validation Loss for the Modified CNN network.

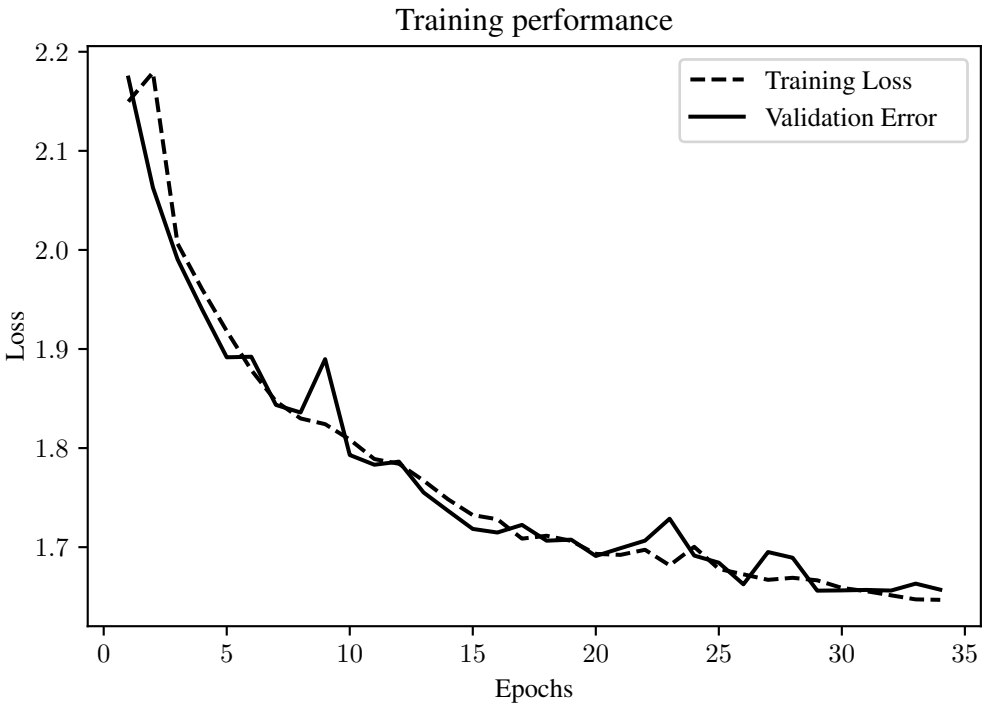


Figure 9.3: Training and Validation Loss for the LSTM based network

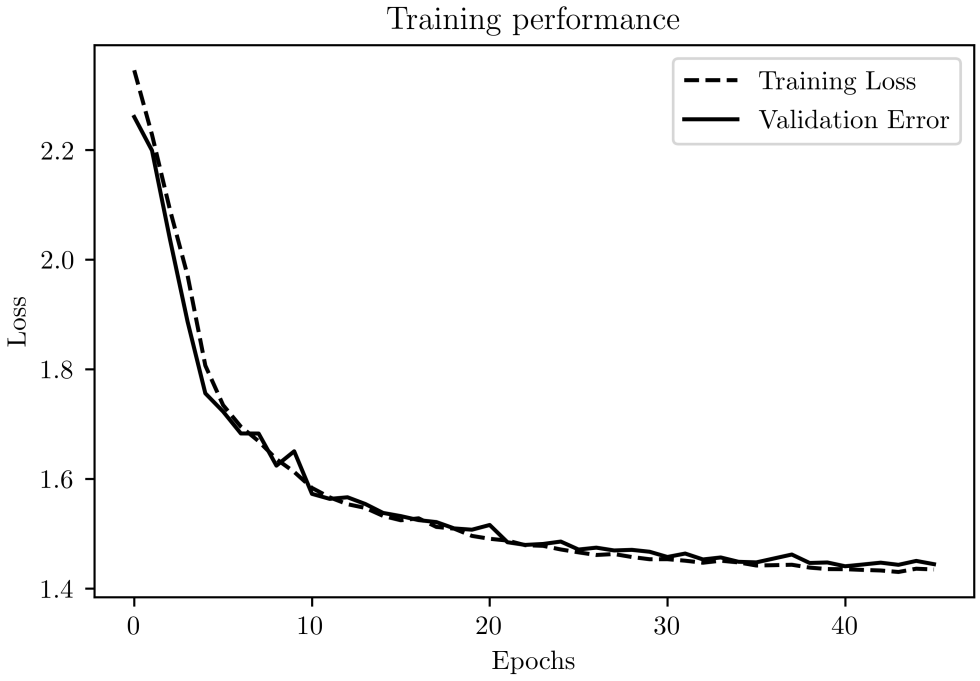


Figure 9.4: Training and Validation Loss for the LSTM network with attention

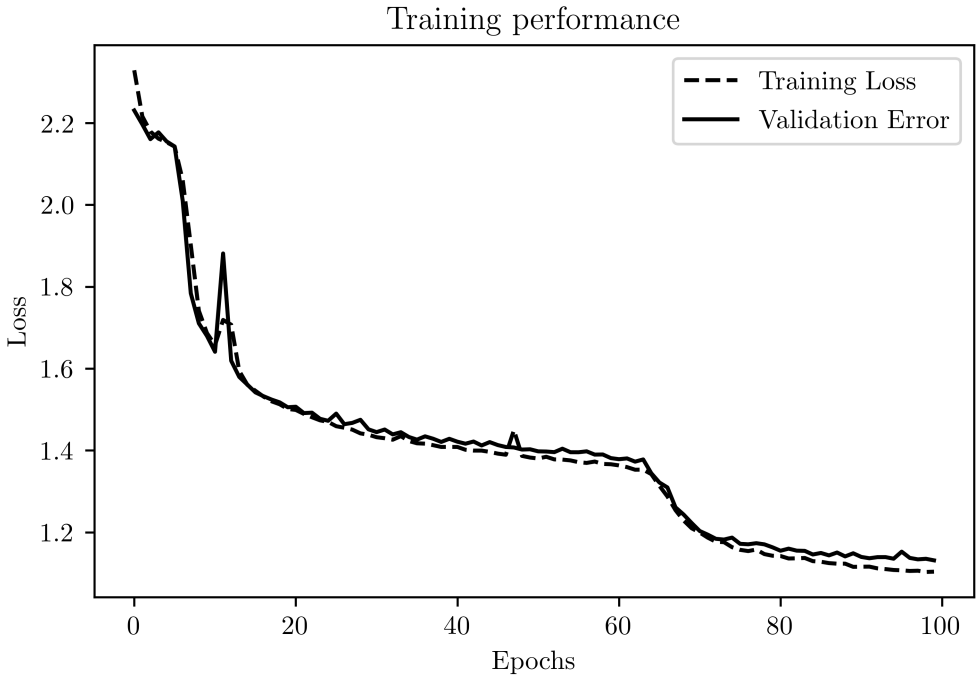


Figure 9.5: Training and Validation Loss for the Bi-LSTM network with attention

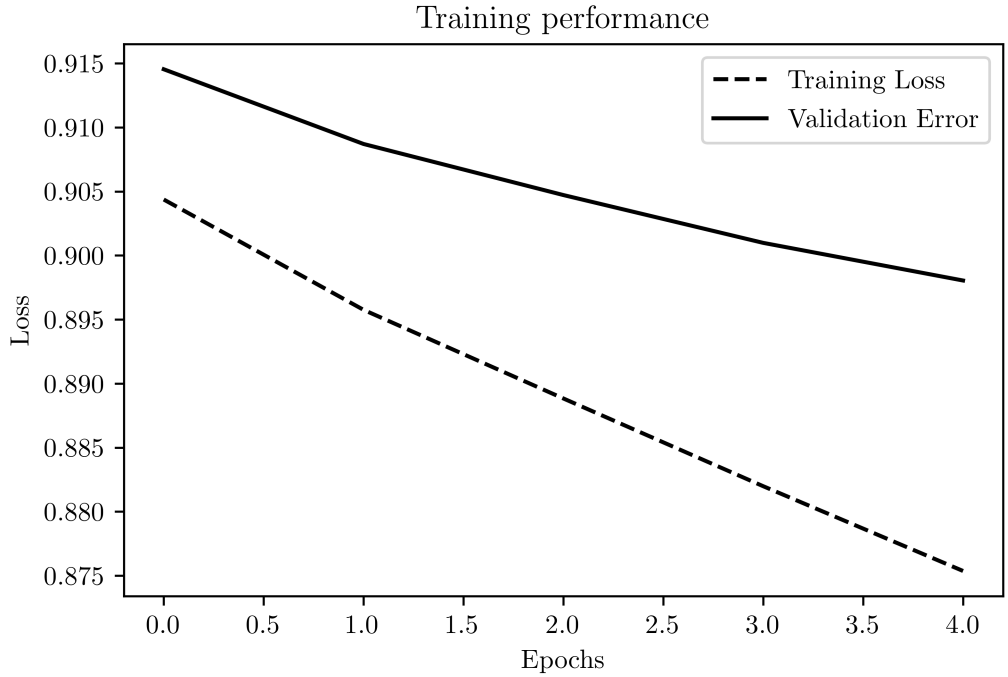
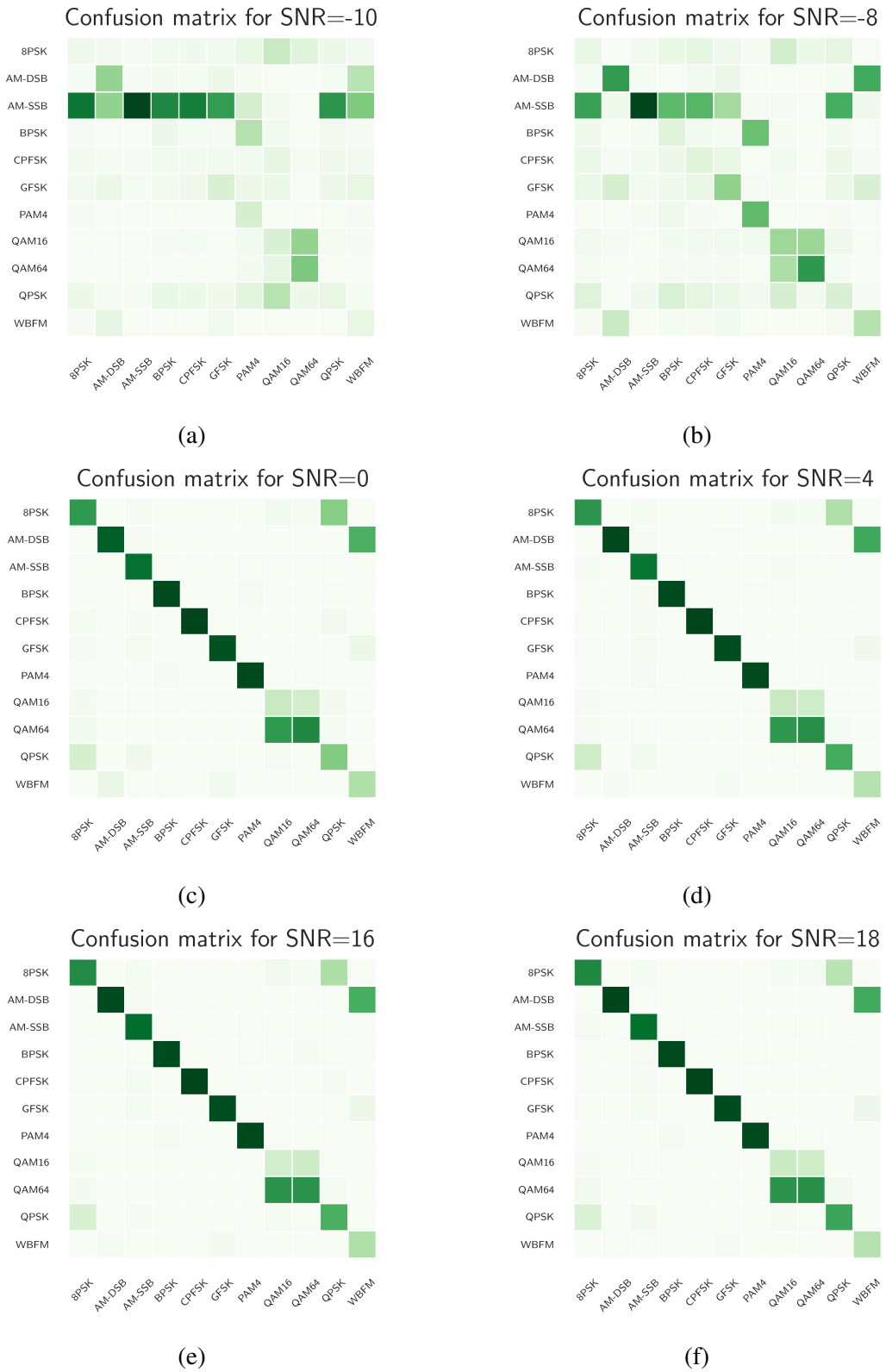


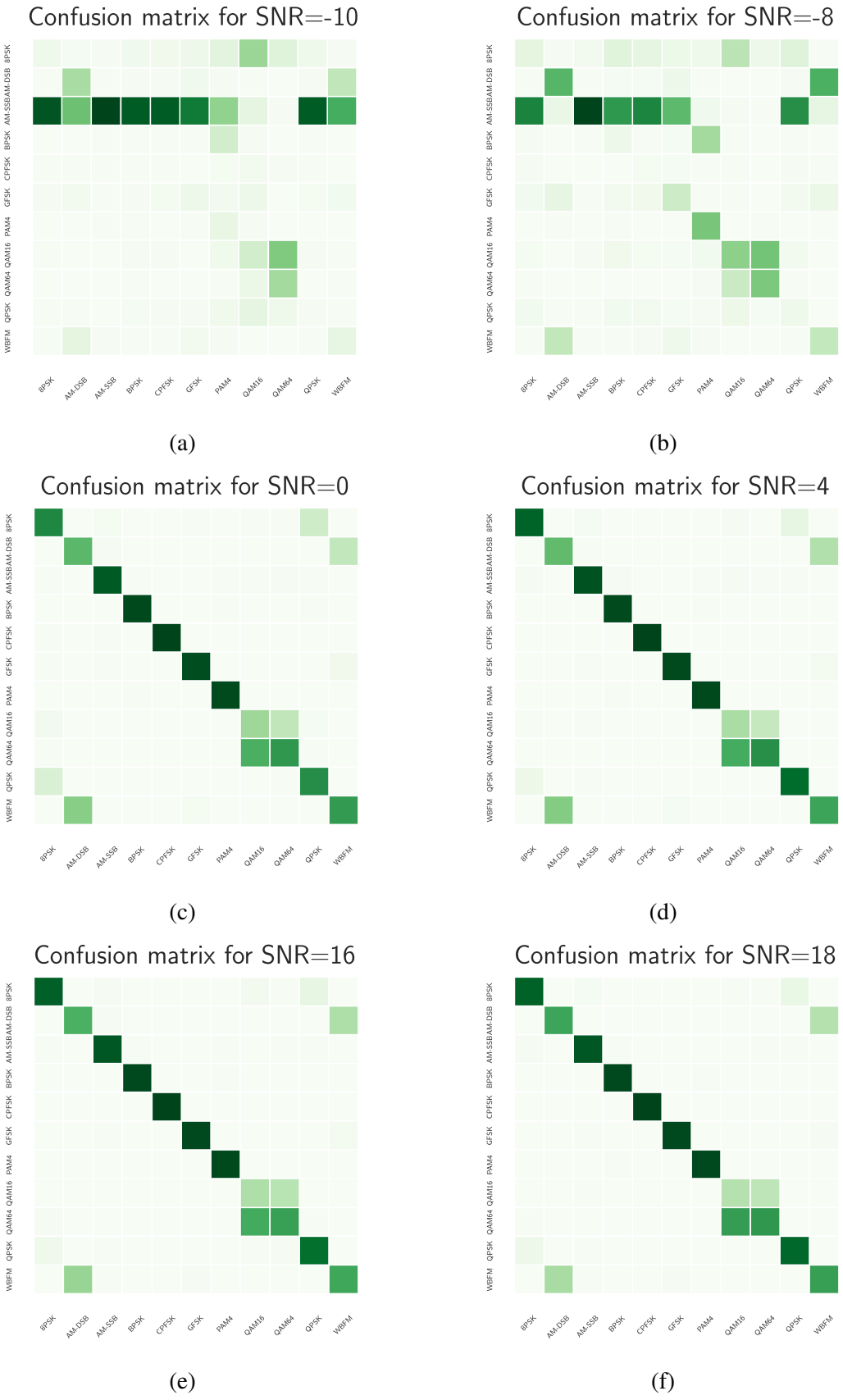
Figure 9.6: Training and Validation Loss for the QNN based classifier.

APPENDIX B: CONFUSTION MATRICES FOR EACH SNR LEVEL



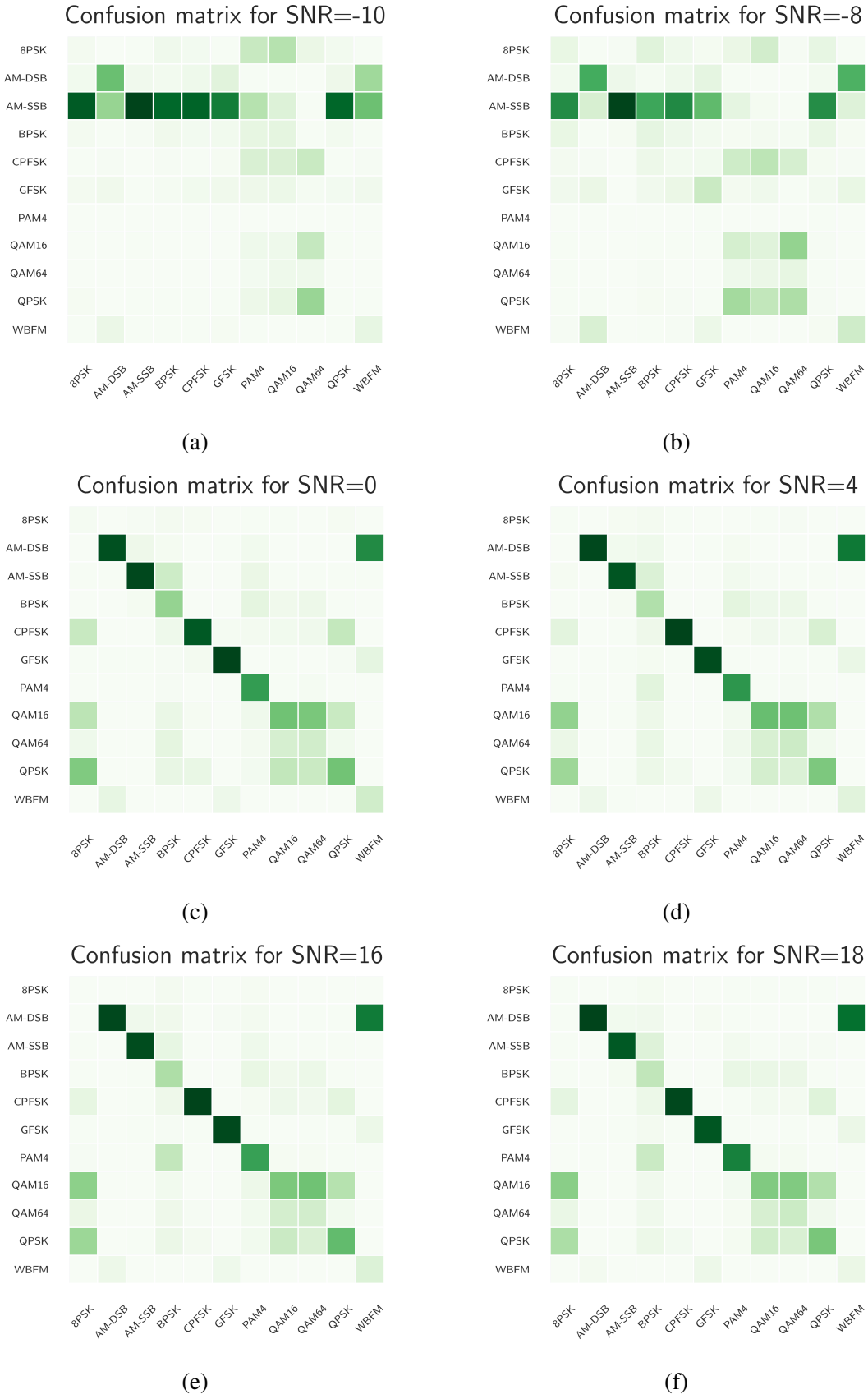
(a) (b) (c) (d) (e) (f)

Figure 10.1: Confusion matrices for the RadioML CNN based network for signals having different SNRs (in dB).



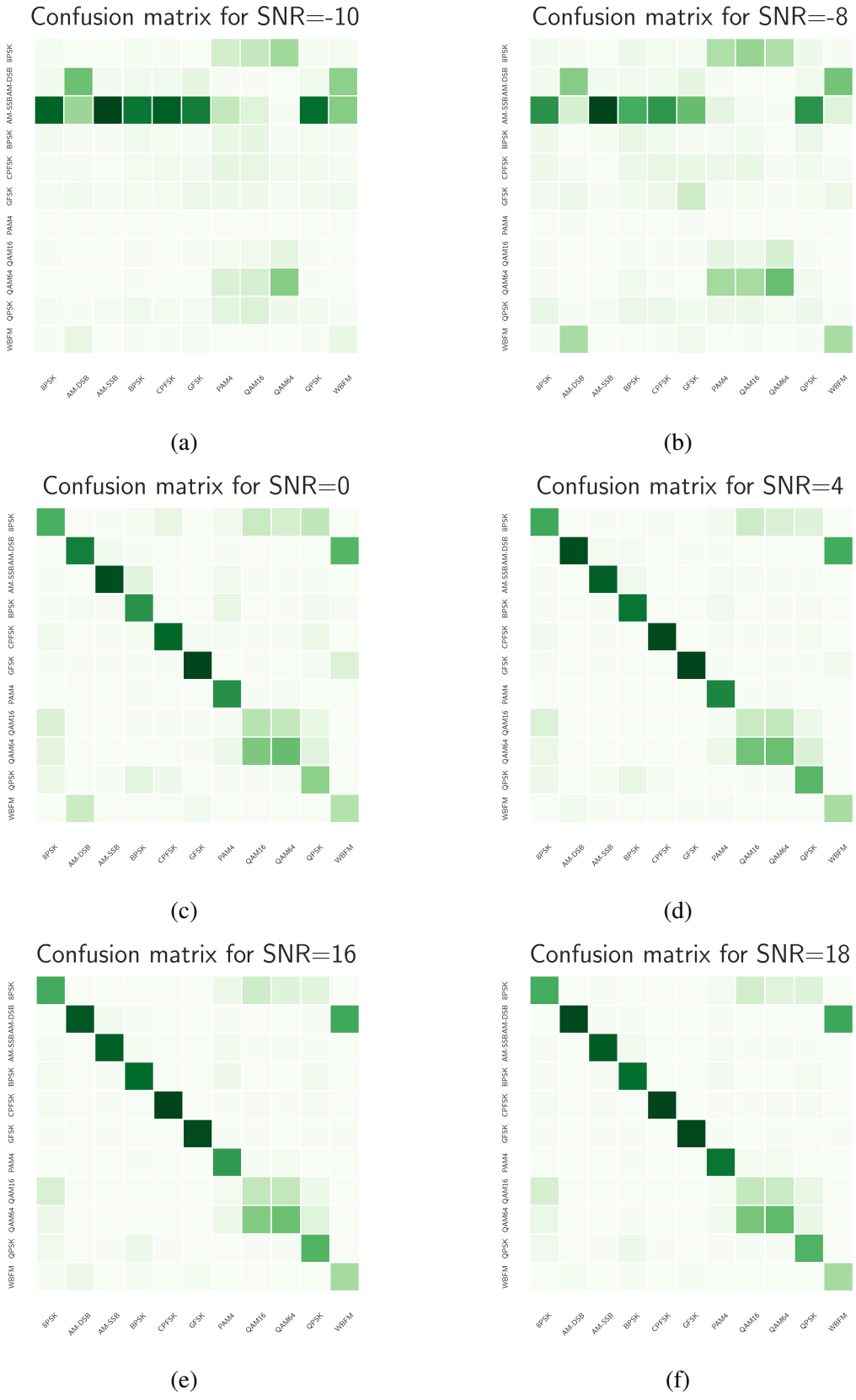
(a) (b) (c) (d) (e) (f)

Figure 10.2: Confusion matrices for the Modified CNN based network for signals having different SNRs (in dB).



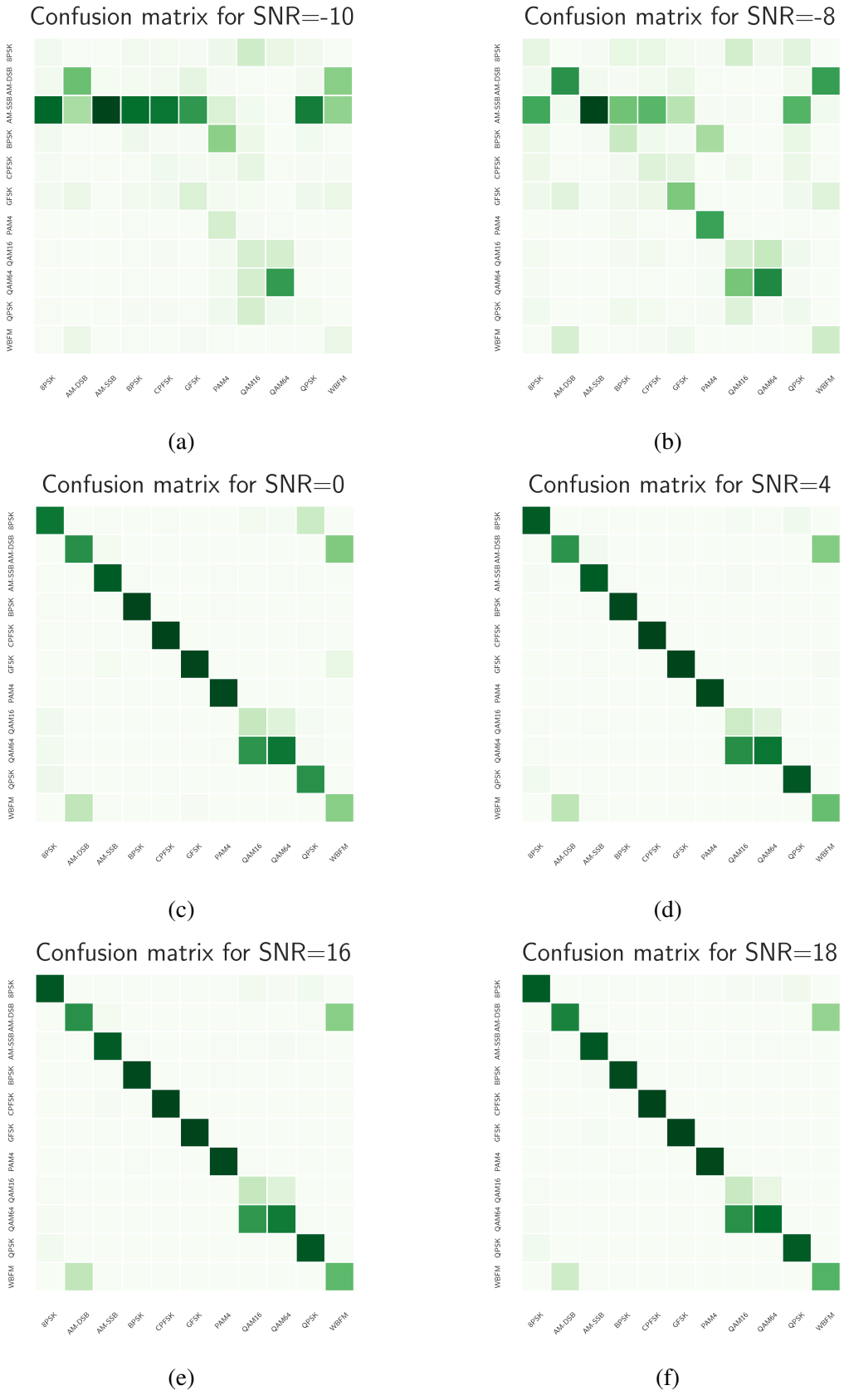
(a) (b) (c) (d) (e) (f)

Figure 10.3: Confusion matrices for the LSTM based network for signals having different SNRs (in dB).



(a) (b) (c) (d) (e) (f)

Figure 10.4: Confusion matrices for the LSTM network with attention for signals having different SNRs (in dB).



(a) (b) (c) (d) (e) (f)

Figure 10.5: Confusion matrices for the Bi-LSTM network with attention for signals having different SNRs (in dB).

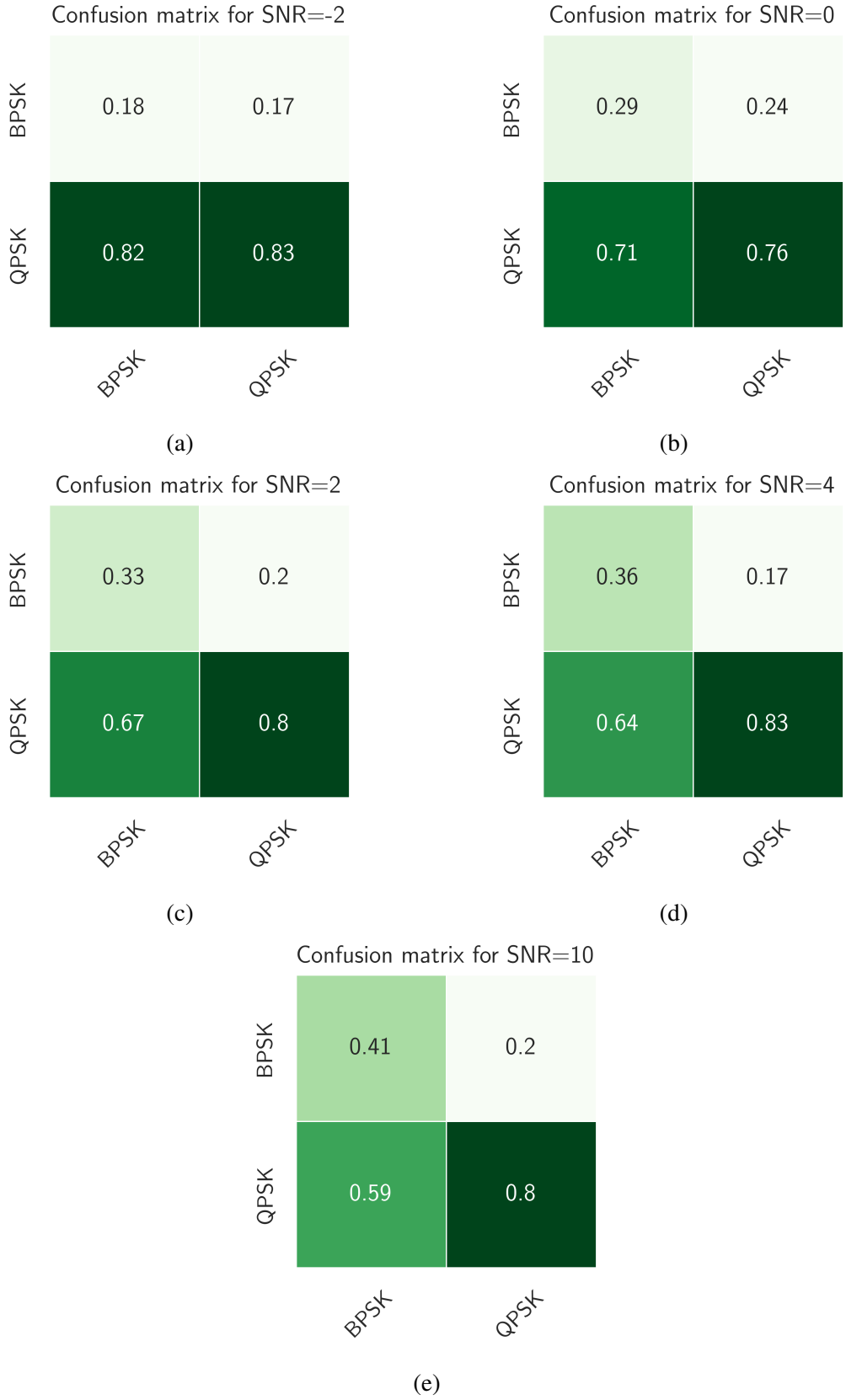


Figure 10.6: Confusion matrices for the QNN based network for signals having different SNRs (in dB).