Hardware-Efficient Automatic Modulation Classification and Blind SNR Estimation for Cognitive Radio Systems: A Novel DBSCAN-Based Approach with an Optimised Implementation

Bill Gavin

Supervisors: Edward Ball and Tiantai Deng

A report submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy

The University of Sheffield
Faculty of Engineering
Department of Electrical and Electronic Engineering

July 24, 2025

Declaration

All sentences or passages quoted in this document from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure.

Name: _____

Signature: _____

Date: _____

List of Publications

During my doctoral research, I have contributed to the following publications:

Journal Articles

1. Bill Gavin, Tiantai Deng, Edward A. Ball (2024). *Low Area and Low Power FPGA Implementation of a DBSCAN-Based RF Modulation Classifier*. IEEE Open Journal of the Computer Society. `https://doi.org/10.1109/OJCS.2024.3355693`

2. Yuqin Zhao, Tiantai Deng, Bill Gavin, Edward A. Ball, Luke Seed (2025). *A Ultra-Low Cost and Accurate AMC Algorithm and Its Hardware Implementation*. IEEE Open Journal of the Computer Society. `https://doi.org/10.1109/OJCS.2024.3381827`

3. Sumin David Joseph, Bill Gavin, Edward A. Ball (2024). *28-GHz Rural Close-to-Ground Propagation Field Test Results and Models*. IEEE Open Journal of Antennas and Propagation. `https://doi.org/10.1109/OJAP.2024.3370968`

Conference Proceedings

1. Bill Gavin, Tiantai Deng, Edward A. Ball (2023). *A Novel Method of Automatic Modulation Classification with an Optimised 1D DBSCAN*. In: Arai, K. (eds) Intelligent Computing. SAI 2023. Lecture Notes in Networks and Systems, vol 711. Springer, Cham. `https://doi.org/10.1007/978-3-031-37717-4_63`

Abstract

Artificial Intelligence is a technology which has the potential to provide significant enhancements for digital communication systems, particularly through the concept of Cognitive Radio (CR). Automatic Modulation Classification (AMC) is a critical function of CR; it offers the ability to identify the modulation scheme of received signals to enable dynamic reconfiguration of physical layer hardware with the aim of maximising data rates and minimising error rates. Current research into AMC systems focuses primarily on achieving high classification accuracy, when the state-of-the-art algorithms are implemented in hardware, the resultant systems suffer from high utilisation and power consumption. To overcome this limitation, this thesis develops a novel, hardware-efficient AMC method based on the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) clustering algorithm. Several novel optimisations to the DBSCAN algorithm are developed which improve hardware efficiency and overcome the inability of the algorithm to differentiate between same-order modulation schemes. Additionally, an automated heuristic method for hyperparameter selection is devised which results in up to a 9.8% increase in classification accuracy in comparison to traditional optimisation methods. Furthermore, a novel hardware implementation of insertion sort is proposed which enables real-time classification with low latency. The proposed optimisations result in a hardware implementation with approximately equivalent size to the state-of-the-art in terms of Flip-Flops and Look-Up Tables, as well as being 71.7% more power-efficient. This approach is also shown to achieve competitive, and in some cases superior, classification accuracy by achieving 100% accuracy at a Signal-to-Noise Ratio (SNR) as low as 10dB in certain cases. Finally, it is demonstrated that the same hardware architecture can be reused for the purpose of non-data-aided SNR estimation with competitive accuracy and is effective across a larger range of SNRs and modulation schemes than existing methods, further enhancing the efficiency of the proposed system.

Contents

List of Figures

List of Tables

List of Abbreviations

AI Artificial Intelligence

ANN Artificial Neural Network

AM Amplitude Modulation

AM-DSB-SC Amplitude Modulation - Double Side Band - Suppressed Carrier

AM-DSB-WC Amplitude Modulation - Double Side Band - With Carrier

AM-SSB-SC Amplitude Modulation - Single Side Band - Suppressed Carrier

AM-SSB-WC Amplitude Modulation - Single Side Band - With Carrier

AMC Automatic Modulation Classification

APSK Amplitude and Phase-Shift Keying

ASIC Application Specific Integrated Circuit

ASK Amplitude-Shift Keying

AWGN Additive White Gaussian Noise

BPSO Binary Particle Swarm Optimisation

CAF Cyclic Autocorrelation Function

CBPSO-MIWS Co-evolution Binary Particle Swarm Optimisation with Multiple Inertia Weight Strategy

CD Constellation Diagram

CDF Cumulative Distribution Functions

CFO Carrier-Frequency Offset

CNN Convolutional Neural Network

CORDIC Coordinate Rotation Digital Computer

CR Cognitive Radio

CSE Common Sub-expression Elimination

CV Cross-Validation

DBMC Density-Based Modulation Classifier

DBSCAN Density-Based Spatial Clustering of Applications with Noise

DBSNR Density-Based Signal-to-Noise Ratio (estimator)

DFT Discrete Fourier Transform

DL Deep Learning

DNN Deep Neural Network

DSP Digital Signal Processing

DT Decision Tree

EDF Empirical Distribution Function

EOS Eighth-Order Statistics

FF Flip-Flop

FFN Feed-Forward Network

FFT Fast Fourier Transform

FM Frequency Modulation

FPGA Field-Programmable Gate Array

FR2 Frequency Range 2

FSK Frequency-Shift Keying

GA Genetic Algorithm

I In-Phase

I/Q In-phase and Quadrature

IDE Integrated Development Environment

LM Levenberg-Marquardt

LO Local Oscillator

LOS Line of Sight

LSTM Long Short-Term Memory

LUT Look-Up Table

MAC Multiply and Accumulate

MAE Mean Absolute Error

MIMO Multiple-Input Multiple-Output

ML Machine Learning

MLP Multilayer Perceptron

mmWave Millimetre Wave

MMCM Mixed Clock Mode Manager

MSE Mean Square Error

NDA Non-Data Aided

NMSE Normalised Mean Square Error

NN Neural Network

OFDM Orthogonal Frequency Division Multiplexing

OOK On-Off Keying

PC Personal Computer

PSK Phase-Shift Keying

Q Quadrature

QAM Quadrature Amplitude Modulation

RAM Random Access Memory

RF Random Forest

RFE Recursive Feature Elimination

RMSE Root Mean Square Error

RMS Root Mean Square

RNN Recurrent Neural Network

SD Standard Deviation

SDR Software-Defined Radio

SMV Square of the Mean by the Variance

SNR Signal-to-Noise Ratio

SSME Split-Symbol Moments Estimator

SVM Support Vector Machines

WBFM Wideband Frequency Modulation

WHS Worst Hold Slack

WNS Worst Negative Slack

Chapter 1

Introduction

The ever-increasing demand for higher data rates, reliability, and coverage for wireless communications continues to drive advancements in network technologies. 5G networks began to see implementation in many European cities in 2019 [1], with the aim of improving wireless communications in these regards. The largest gains in data rates and bandwidth were obtained with the Frequency Range 2 (FR2) band which utilises Millimetre Wave (mmWave) frequencies greater than 24.25GHz [2]. The FR2 band was shown to provide increased data rates, yet signals transmitted within this frequency range suffer from poor propagation characteristics due to large path loss, requirements for Line of Sight (LOS), and high levels of Doppler shift [3]. These limitations of high frequency communications are further exacerbated by the urban environments where they are primarily deployed [4], which inherently features numerous obstacles such as buildings, vehicles, and natural objects which can frequently obscure the LOS [5]. Furthering these issues is the transient nature of urban environments where people and vehicles constantly change location meaning that LOS conditions and the distance from transmitters can change quickly and unpredictably. Therefore, connection quality can vary significantly depending upon movement within an environment and the obstacles in the path between the transmitter and receiver. Machine Learning (ML) and Deep Learning (DL) have been identified as technologies which can offer a means of managing this challenge. Wang et al. and Letaief et al. envision that future wireless systems will be fundamentally different from previous generations, with the key difference being embedded intelligence enabling real-time adaptation to the local environment [6, 7]. Within this vision of intelligent wireless systems is the concept of Cognitive Radio (CR), which utilises ML and DL technologies to enable dynamic spectrum access and adaptive modulation. Two critical component technologies of CR are Automatic Modulation Classification (AMC) and Signal-to-Noise Ratio (SNR) estimation [8]. AMC enables receivers to identify changes to the modulation format utilised for communication, while SNR estimation provides the necessary information for identifying when changes in modulation scheme would be advantageous to mitigate changes in channel conditions. By continuously monitoring the rapidly changing channel conditions caused by blockages, movement, and interference, these techniques can provide the necessary functionality to allow for adjustments to signal formats and physical hardware, thus maintaining target error rates when high frequency carriers are utilised in complex environments.

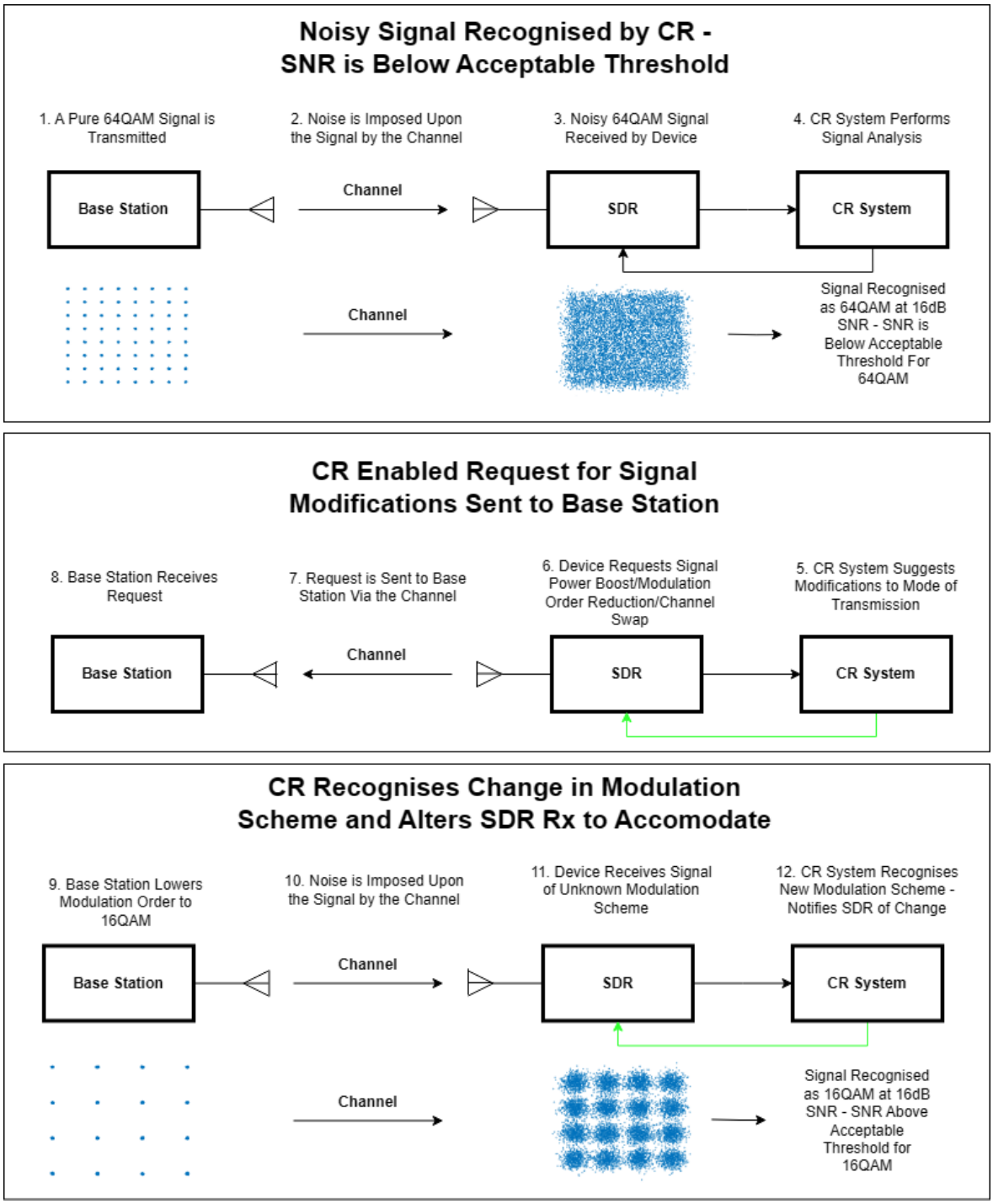


Figure 1.1: *How AMC and SNR Estimation May Assist CR Functionality*

1.1 AMC and SNR Estimation

AMC has been identified as a key component of CR systems that is particularly relevant for addressing the propagation challenges inherent with high frequency communications, as it enables the ability to dynamically adjust the modulation scheme which is utilised for communication as well as informing the system of the required alterations to the physical layer hardware to handle such adjustments [8, 9]. While lower order modulation formats are more robust to the impairments imposed by noise, they impose a limit on achievable data rates [10]. Conversely, higher order modulation schemes can provide increased data rates as each symbol may represent a greater number of bits but are more sensitive to signal quality degradation due to SNR [10].

SNR estimation techniques can assist in managing this trade-off by providing the required information for CR enabled systems to dynamically modify the employed modulation scheme to maximise data rates while maintaining error rates within acceptable boundaries. For instance, when a high SNR is identified, a higher-order modulation scheme can be employed to maximise throughput. When the SNR is found to decrease due to channel fading or interference, the system can switch to a more robust, lower-order modulation format to ensure reliable communication. This adaptive process, including the mechanism by which the Software-Defined Radio (SDR) is modified to accommodate the changes, is shown in figure 1.1.

The interplay between AMC and SNR estimation is therefore crucial for mitigating the SNR impairments in 5G and future 6G networks. But there are also numerous additional applications for these technologies, with one of the most important fields in which AMC can particularly be applied is signal intelligence. By determining the modulation formats of signals within the local environment, potential threats and unauthorised transmissions may be identified [11]. Rather than being utilised for security, AMC may also be applied for offensive means. Recognition of the modulation scheme being employed by adversaries may assist with obtaining intelligence via the decoding of the transmitted signals, furthermore, when applied with SNR estimation a measure of distance to the location of transmission can be obtained [12]. It is therefore clear that both AMC and SNR estimation are techniques which will play an important role in the future of communications systems.

1.2 Pilot-Based and NDA SNR Estimation

To effectively utilize AMC, particularly in the challenging mmWave environments envisioned for CR systems, the method of SNR estimation becomes critical. Typically, the SNR is determined via the exchange of pilot symbols [10]. Pilots are known signals inserted into a transmitted data stream at regular intervals. As the receiver knows the expected appearance of the pilot, the received pilot can therefore be used a measure of the distortion applied to the signal via the channel. The distortion may be cause by a variety of effects such as multipath fading, Doppler shift, and noise.

While the usage of pilot symbols has been shown to provide a means of accurately and reliably estimating channel conditions, the technique is not without limitations. Firstly, every pilot symbol transmitted is a message symbol not transmitted; the proportion of resources (time, power, frequency) dedicated to the transmission of pilot symbols rather than message

symbols is known as pilot symbol overhead [10]. A high-frequency communications link in a complex environment such as a city may require an overhead of at least 10% in single user scenario, and as much as 25% for multiple users [13]. Thus a significant proportion of all transmitted information is dedicated to enabling channel estimation. Secondly, it has been found that systems with many antennas such as massive Multiple-Input Multiple-Output (MIMO) require increased pilot overhead, Jindal et al. found that the optimum pilot overhead scales with the number of employed antennas [14]. The required pilot overhead therefore diminishes the gains in data throughput which this emerging technology offers. Finally, pilot-based channel estimation inherently introduces latency in responding to rapid channel changes. Estimation is only performed upon the periodic arrival of pilot symbols, limiting the system's ability to adapt in real-time.

In contrast, performing channel estimation without the usage of pilot symbols offers the potential for all of these limitations to be mitigated. Non-Data Aided (NDA) SNR estimation eliminates the requirement to transmit pilot sequences, directly increasing data rates. Furthermore, by reducing or eliminating pilot overhead, the practical deployment of promising technologies such as massive MIMO are facilitated. Finally, by operating directly on the data signal, the potential for more continuous channel monitoring is provided, thus faster responses to rapidly changing channel conditions may be enabled. Therefore, NDA SNR estimation emerges as a highly promising alternative to the traditional pilot-based techniques, unlocking the full potential of advanced wireless technologies such as massive MIMO and AMC.

1.3 Limitations of Existing AMC and NDA SNR Estimation Hardware

The research community has developed various Artificial Intelligence (AI)-enabled solutions for dynamic modulation scheme recognition [15–17]. However, state-of-the-art implementations have thus far generally been implemented in software. Hardware implementations of DL-based solutions demonstrate their unsuitability for deployment in mobile systems due to their focus on performance at the expense of efficient implementation efficiency, resulting in high power consumption and implementation sizes [17–19]. Techniques such as quantisation and pruning [17–19] have been used to optimise DL networks for deployment in resource constrained applications but the achieved utilisation and power consumption remains considerable. Conversely, feature extraction methods have been shown to offer smaller implementations but fall short of the accuracy achieved by the techniques which employ DL, particularly for high order modulation schemes and at lower SNRs [20, 21].

Similar challenges exist in NDA SNR estimation. Several effective methods have been proposed such as M_2M_4 [22], Square of the Mean by the Variance (SMV) [22], and AI-based systems [23, 24]. It has been found that algorithmic blind SNR estimators may result in low hardware utilisations, but many are designed for and thus limited to specific modulation schemes [22,25,26]. Some algorithmic methods were found to have wider modulation applicability by making use of reconfigurable parameters [27, 28]. Algorithmic estimators in general have been found to only exhibit strong performance within limited SNR ranges, their estimated SNR values tend to asymptote above and below certain thresholds. While DL-based SNR estimators were found to have the strongest estimation accuracy and did not suffer from asymptotic behaviour [23, 24, 29, 30], their strong performance has only been demonstrated

on small sets of low order modulation schemes. As with AMC the implementation of DL systems in resource constrained applications poses significant challenges. There is a clear need for a hardware implemented NDA SNR estimation system capable of maintaining consistently strong performance on a variety of modulation schemes and SNRs, without requiring excessive resources in terms of hardware utilisation and power.

Wang et al. [7] highlight the challenges with deploying deep learning systems in mobile devices, concluding that either reducing model complexity or offloading processing to the cloud are the only solutions to mitigating the high power consumption and chip area requirements that machine learning models suffer from. While cloud offloading is viable for many functions which are performed on mobile devices, such as machine translation [31], photograph editing [32], and text generation [33], communication systems are in the unique position where they are themselves the mechanism by which offloading to the cloud is performed. Reliance upon communication with a data centre to perform communication enhancing tasks is therefore paradoxical. Thus, developing resource-efficient ML techniques is the only viable path forward for on-device implementation.

This is the challenge which this thesis aims to address. By developing from first principles a hardware implementation of a technique which has shown promising performance in terms of both AMC and SNR estimation in software, this work demonstrates that strong AMC and SNR estimation performance can be achieved with a single efficient hardware structure. This represents a crucial advancement for viable AI/ML enhancements in communications systems, particularly for deployment in battery-powered mobile devices. While separate smaller and more power-efficient AMC or SNR estimation systems provide valuable enhancements for next-generation communications, a joint system with a unified architecture offers the comprehensive solution needed to overcome the resource constraints that currently limit high-frequency communications to infrastructure-only deployments. This is particularly critical as 5G/6G networks expand into urban environments where mobile devices must adapt to rapidly changing channel conditions without access to the computational resources available at base stations.

1.4 Contributions Towards AMC

The work in this thesis attempts to move towards CR systems which are suitable for deployment in mobile and edge devices by developing an underexplored technique of signal analysis based upon clustering algorithms. The proposed solution is specifically optimised for usage in hardware and on real-time streams of data. The clustering algorithm which is the focus for development is known as Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [34,35]. This algorithm has previously been applied to the task of modulation classification in the context of both wireless [36], and optical [37,38], to limited success. Prior AMC systems based upon the DBSCAN technology have demonstrated the potential for strong classification accuracy but have been unable to distinguish between modulation schemes of the same order and have only been implemented in software, lacking hardware optimisations.

This thesis proposes a hardware-focused solution which not only improves the worst-case computational complexity of DBSCAN from $O(n^2)$ to $O(n)$ by reducing the number of *rangeQuery* operations required to obtain a clustering result by a factor of n (where n is equivalent

to a clustering batch size), but does so whilst solving the issue of the algorithm being unable to differentiate modulation schemes of the same order. These two goals are achieved by decomposing the DBSCAN algorithm from a 2D clustering algorithm to operating in a single dimension on the magnitude and argument data which forms the constellation diagram. Clustering the data in this manner allows for differing modulation schemes of the same order to be distinguished. The decomposition to 1D datasets also allows for various optimisations the DBSCAN algorithm, much of the extraneous functionality can be discarded to result in a more efficient algorithm that is tailored for the task of AMC. One key optimisation is the ability to sort datasets which are unidimensional without the requirement for an indexing database structure, as is the requirement for 2D datasets, sorting the data eliminates the requirement to execute the computationally intensive *rangeQuery* function. A novel implementation of the insertion sort algorithm is developed which allows for the sorting of real-time data streams without any increases in latency. New techniques for optimising DBSCAN are also developed and explained, an automated heuristic for hyperparameter selection which is shown to provide increased accuracy compared to the manual and subjective elbow point method is proposed. Detailed investigations into the methods of optimising the proposed algorithm are given so that the reader may reproduce the optimised performance which is reached in this thesis. The Density-Based Modulation Classifier (DBMC) system is designed for and implemented on an FPGA, the process of implementing all the constituent functional blocks which form the complete algorithm is described, along with the system's functionality and control. Four configurations (DBMC-50, DBMC-250, DBMC-500, and DBMC-1000) of varying implementation size and classification performance are proposed to suit different use cases. The utilisation, power consumption, latency, and classification performance across a wide range of modern digital modulation is covered in depth and compared with the strongest hardware implemented AMC systems from the literature, the key contributions include:

- **The first hardware-implemented clustering-based AMC system:** A 2D to two 1D decomposition of the DBSCAN algorithm is employed to enable a highly efficient clustering algorithm-based modulation classifier, the first clustering hardware implementation to be employed for AMC purposes. The proposed decomposition facilitates a reduction in the number of iterations required to obtain a clustering result by a factor of n, where n is equal to the clustering batch size, via the elimination of the *rangeQuery* operation. This method also enables the system to distinguish between differing modulation schemes of the same order, a feat which has not been achieved by clustering based techniques in prior works [37].

- **Reductions in power consumption and latency:** An array of systems are proposed each suitable for specific use-cases. The two smallest of the proposed systems (DBMC-50 and DBMC-250) represent the smallest, quickest in terms of latency, and least power consuming of any logic-based AMC implementation. They are however found to be limited to classifying low-order modulation schemes. The largest implemented model (DBMC-1000) achieves comparable FF and LUT utilisation to the state-of-the-art CNN hardware implementations [19,39], yet improves upon them in terms of latency and power efficiency due to a pipelined architecture and minimal bit-switching due to a sorted datapath respectively.

- **State-of-the-art accuracy on particular datasets:** The proposed method achieves

AMC accuracy superior to the state-of-the-art when utilising a dataset of 4QAM, 16PSK, 64APSK, and 256QAM which is similar to what is employed in current 5G systems and provides a wide ratio of noise robustness to spectral efficiency. On this particular dataset the software implemented DBMC-5000 maintains 100% classification accuracy to a lower SNR than any deep learning system was demonstrated to be capable of, with the closest being the image classifier M-CNN [58], only the subtractive clustering algorithm was found to be capable of comparable accuracy at 5dB SNR [51]. This performance is achieved due to the ability of the system to distinguish between modulation formats with high-accuracy.

From these results it is found that implementing the DBSCAN method of AMC can result in a suite of implementations which achieve higher peak accuracies than the state-of-the-art whilst also providing significant reductions in terms of power consumption, latency, and in some cases utilisation. The system is designed for real-world deployment in mind, meaning that the data path is intricately designed to support inputs of real-time data streams and is pipelined to ensure maximum efficiency is always obtained. While primarily designed for single-carrier waveforms, the algorithm can be applied to multi-carrier systems following Fast Fourier Transform (FFT) demultiplexing.

1.5 Contributions Towards NDA SNR Estimation

As previously covered, a key requirement of CR is to react to the noise level of the environment and dynamically alter the power or modulation scheme of a transmitted signal. To do this, an accurate measure of SNR is required. This thesis demonstrates that the DBMC system can be repurposed as an effective SNR estimator with the same hardware, but modified classifier weights. This dual functionality highlights the versatility of the proposed signal analysis technique and represents a significant step towards a comprehensive CR system.

The proposed Density-Based Signal-to-Noise Ratio (estimator) (DBSNR) is a blind or NDA SNR estimator, eliminating the requirement for computation and hardware to measure background noise levels or increases the data-rate of transmitted signals due to the lack of a requirement for a pilot sequence to be transmitted instead of data. It was discussed that existing blind SNR estimation techniques feature one or more of the following limitations: poor accuracy over a large SNR range, the ability to estimate the SNR of only a few select modulation schemes, or large utilisations when implemented in hardware. The proposed system addresses these limitations. Key findings include:

- **Efficient Hardware Reuse:** The proposed SNR estimation functionality is achieved with minimal additional hardware overhead compared to the DBMC system, demonstrating efficient multi-use hardware design. This is the first hardware implementation to demonstrate both AMC and NDA SNR estimation in a single implementation. The estimation mechanism exploits the relationship between constellation point density and SNR, every modulation scheme exhibits such a relationship therefore a regression model can be trained to relate the obtained number of clusters (as a proxy for density) to the SNR.
- **Wide Modulation Range:** The proposed system estimates the SNR of Quadrature Amplitude Modulation (QAM) signals from order 2 to 1024, Phase-Shift Key-

ing (PSK) signals of orders 8 to 32, and Amplitude and Phase-Shift Keying (APSK) signals of orders 16 to 128, therefore exceeding the range of previously reported NDA SNR estimators. The wide modulation range is achieved because the clustering-based approach directly measures constellation point density, which maintains a predictable relationship with SNR across all modulation schemes. Unlike traditional moment-based or likelihood-based estimators that rely on scheme-specific statistical properties, this density-based approach generalizes naturally to any constellation structure.

- **Consistent Performance Across a Wide SNR Range:** Unlike many existing estimators that exhibit performance degradation at certain SNR levels, the proposed system maintains consistent performance from -10 dB to 40 dB SNR for most tested modulation schemes. This consistency is achieved because the density-based approach remains robust to the signal distortions that cause traditional estimators to fail at extreme SNR values. At low SNR, where moment-based methods suffer from noise-induced bias, the clustering approach still captures the underlying constellation structure. At high SNR, where some estimators saturate, the density measurement continues to provide meaningful discrimination.

- **Competitive Accuracy:** The proposed system achieves a minimum Mean Square Error (MSE) comparable to state-of-the-art NDA SNR estimators [24,27,29,38]. However, the proposed method was found to suffer occasional spikes in MSE due to ε hyperparameter tuning, but these spikes may be mitigated with differing parameter values.

These findings show that DBSNR can provide robust SNR estimation across a wide range of modulation orders and SNRs. The fact that only new Multilayer Perceptron (MLP) weights are required to be loaded into the classifier structure demonstrates that the DBMC system structure can provide both AMC and NDA SNR estimation functionality within a single efficient hardware core. Thus, the work proposed in this thesis is found to be an effective low complexity solution which can be utilised for low complexity CR implementations in mobile and edge devices.

1.6 Thesis Structure

The remaining chapters of this thesis can be thought of as comprising 3 distinct sections: The first two chapters following this introduction provide a background to modulation and discuss the state of the literature. Following this background being provided, all of the remaining chapters concern the creation, optimisation, and testing of the proposed system. Chapters 4 through 6 detail the creation of the proposed system, discussing how DBSCAN was modified to create an efficient hardware implementation as well as how it was optimised for the tasks of AMC and NDA SNR estimation. Chapters 7 though 9 conclude the thesis by exploring the results obtained via testing the proposed system, comparing said results with the literature, and finally discussing and highlighting the contributions of the proposed work.

A brief summary of each of the remaining chapters of this thesis is provided: Chapter 2 gives a brief introduction to the fundamental concepts which are required to be understood to grasp the topics explored later in this thesis. The chapter introduces modulation schemes, constellation diagrams, and the effects of noise upon constellation diagrams.

Chapter 3 then presents an in-depth review of the AMC and NDA SNR estimation literature, discussing the advantages which various techniques may have and identifies any limitations which can be addressed by the proposed system. The strongest performing software and hardware implementations are compared using accuracy and hardware utilisation statistics. Finally, ideal candidates for realising an accurate yet efficient AMC and SNR estimation system are identified.

Chapter 4 describes the design and structure of the DBSCAN AMC system, detailing the rationale behind each design decision. It is discussed how prior attempts to employ DBSCAN for AMC and SNR estimation are limited to operating on only modulation schemes of different modulation orders as well as suffering from inefficiencies due to the inclusion of DBSCAN functionality which is not required to achieve accurate AMC and SNR estimation. It is shown how through decomposing the constellation diagram data to 2 1D datasets not only enables the system to distinguish modulation schemes of equivalent order, but also results in an algorithm which is less computationally complex and more conducive to efficient implementation in hardware.

Chapter 5 builds upon the explanations of the DBSCAN AMC algorithm which are found in Chapter 4, to detail how the algorithm was implemented in hardware. Various design choices are explained, different configurations are proposed, and optimum use cases are identified. Finally, the implementation results are compared with the state-of-the-art hardware implementations reviewed in Chapter 3.

Chapter 6 discusses how best to optimise the hyperparameters of the proposed system. The effects of the dataset size, *minPts*, and ε hyperparameters upon the feature extraction mechanism are explored and recommendations for finding values which achieve a balance between implementation size and classification accuracy are provided. Traditional k-distance graph optimisation techniques are used to demonstrate the variability of the ε hyperparameter, this method is used to find strong ε values. It is then demonstrated how the k-distance graph elbow point method introduces significant subjectivity to the ε optimisation process, two novel methods are devised and proposed which are each shown to result in an increase of up to 10% accuracy when employed. Finally, strategies to optimise hyperparameters for the hardware implementation are detailed.

Chapter 7 evaluates the classification performance of each DBSCAN AMC configuration. Results are first presented for datasets containing an array of QAM, PSK, and APSK signals up to a modulation order of 1024. Then the achieved results are compared with the state-of-the-art systems from the literature across a variety of datasets including: QAM, PSK, and APSK signals with a maximum modulation order of 16, 64, and 256, as well as when only a specific set of signals which provide various ratios of SNR robustness to data throughput are utilised. These results are compared with the results of the leading hardware-implemented AMC systems from the literature.

Chapter 8 extends the proposed system to include NDA SNR estimation; it begins by discussing the minimal hardware modifications which are required to provide the DBMC system SNR estimation abilities and then explains the SNR estimation mechanism. SNR estimation accuracy results across QAM, PSK, and APSK signals are given in terms of MSE against SNR. Finally, these results are compared with the estimation performance of the leading techniques from the literature.

Chapter 9 concludes this thesis by summarising the key findings regarding the hardware

implementation, AMC accuracy, and NDA SNR estimation accuracy. A future work section identifies the limitations of this work and proposes potential future research directions, including further optimization and improvement of the proposed system.

Chapter 2

Background

This chapter provides a brief overview of the background concepts which are required to be known if the remaining work discussed in this thesis is to be understood. Firstly, the fundamentals of digital modulation are outlined. Then the derivation of the In-phase and Quadrature (I/Q) signal representation is explained. This representation is then utilised to explain the concept of the constellation diagram. Finally, the constellation diagram is employed to explain the effects of noise upon a modulated signal.

2.1 An Introduction to Modulation

Modern communication systems are built upon exploiting the fundamental properties of waveforms. All waveforms can be described by three features: frequency, phase, and amplitude. This description of a waveform may be expressed mathematically in the form shown in Equation 2.1:

$$y(t) = A(t)\cos(2\pi f(t) + \phi(t)) \tag{2.1}$$

Where $A(t)$ represents the amplitude as a function of time, $f(t)$ represents the frequency as a function of time, and $\phi(t)$ represents the phase of the signal as a function of time. This representation shows how the amplitude, phase, and frequency may be varied to encode information. For example, a basic 1-bit system could be created in which two frequencies are used to represent a 0 and a 1. A transmitter may transmit a 10kHz wave to represent a 0 and a 20kHz wave to represent a 1. Representing data in this way is known as modulation. An example of this described waveform in the time domain is shown in Figure 2.1.

At the receiver, demodulation circuitry reconverts each wave back into their binary representation. Modulating data with amplitude, frequency, or phase is known as Amplitude-Shift Keying (ASK), Frequency-Shift Keying (FSK), and Phase-Shift Keying (PSK) respectively. Modern communication systems primarily use PSK and two different combinations of PSK and ASK known as QAM and APSK [10]. Rather than modulation, frequency is generally used for multiplexing purposes, which means the frequency spectrum is divided and distributed across users, this allows for multiple signals to be transmitted within a space without suffering large amounts of interference as well as allowing for individual communica-

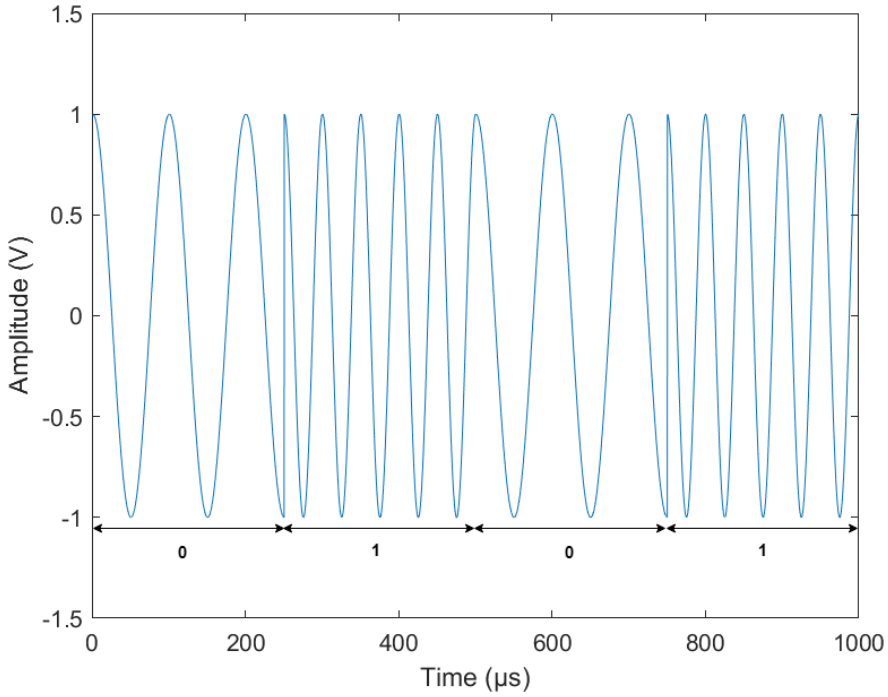


Figure 2.1: *An Example of a 2FSK Modulated Waveform*

tions devices to extract signals of only the frequency which they are tuned to [10]. The work discussed in this thesis will focus only upon PSK, QAM, and APSK modulation schemes due to their usage in digital modulation.

2.2 The Relationship Between Modulation Order and Bitrate

Rather than transmitting two waves with a binary representation as with the example shown in Figure 2.1, increasing the number of different transmitted waves allows for the mapping of binary sequences to each waveform. For instance, transmitting 8 different PSK modulated waves allows for each waveform to represent the numbers 0-7, or 3 bits, thus allowing for higher data rates. The number of waves chosen for a modulation scheme is known as the order and is written as a number before the shift-keying mode, such as 8PSK in this case. Generally, the number of different variations of the varied wave characteristic is given in terms of a number prefix, such as 8PSK for 8 phase variations, the exception to this is when the properties are only varied twice or four times, in these cases the number prefix is replaced with Binary (B) and Quadrature (Q) respectively [40]. Each unique variation of a modulated wave is known as a symbol. The number of bits which a symbol can represent is determined by Equation 2.2.

$$B = \log_2(M) \tag{2.2}$$

Where M is equal to the order of modulation and B is the number of bits. By increasing the modulation order, a higher data rate can be achieved for an equivalent symbol rate. This is demonstrated by viewing QPSK in the time domain which is shown in Figure 2.2

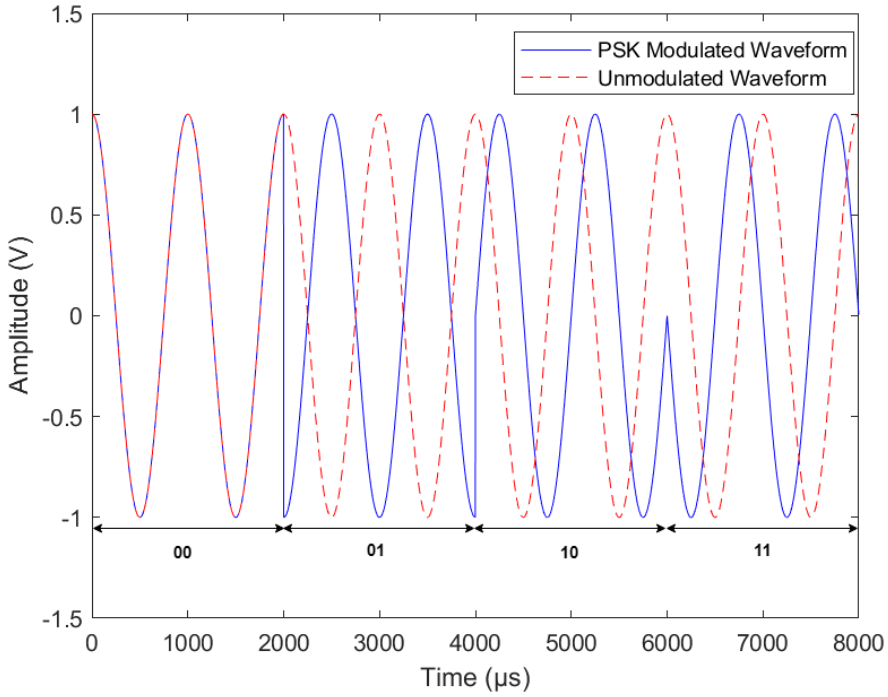


Figure 2.2: *An Example of How a QPSK Modulated Waveform Transmits 2-Bit Sequences of Data*

2.3 The In-Phase and Quadrature Decomposition

Digitally modulated waveforms are near universally described as a combination of two component waveforms: the In-Phase (I) and Quadrature (Q) components. Utilising this framework allows for the analysis and generation of digital modulation schemes in terms of the amplitudes of each component. Finding these components can be done as follows:
Take the general form of a modulated signal:

$$y(t) = A\cos(2\pi ft + \phi) \tag{2.3}$$

Euler's formula states that:

$$\cos(\phi) = \frac{e^{j\phi} + e^{-j\phi}}{2} \tag{2.4}$$

When substituting Equation (2.4) into Equation (2.3), the following is obtained:

$$y(t) = A\frac{e^{j(2\pi ft+\phi)} + e^{-j(2\pi ft+\phi)}}{2} \tag{2.5}$$

The exponentials may be further expanded to obtain:

$$y(t) = \frac{A}{2}\left(e^{j2\pi ft}e^{j\phi} + e^{-j2\pi ft}e^{-j\phi}\right) \tag{2.6}$$

The term $e^{j\phi}$ may be expressed in rectangular form using Euler's formula:

$$e^{j\phi} = \cos(\phi) + j\sin(\phi) \tag{2.7}$$

Similarly, $e^{-j\phi}$ may be expressed as:

$$e^{-j\phi} = \cos(\phi) - j\sin(\phi) \tag{2.8}$$

Substituting Equations (2.7) and (2.8) back into Equation (2.6) obtains:

$$y(t) = \frac{A}{2}\left(e^{j2\pi ft}(\cos(\phi) + j\sin(\phi)) + e^{-j2\pi ft}(\cos(\phi) - j\sin(\phi))\right) \tag{2.9}$$

By expanding and grouping terms, the following is obtained:

$$y(t) = \frac{A}{2}\left(\cos(\phi)(e^{j2\pi ft} + e^{-j2\pi ft}) + j\sin(\phi)(e^{j2\pi ft} - e^{-j2\pi ft})\right) \tag{2.10}$$

Euler's formula also states that:

$$\cos(2\pi ft) = \frac{e^{j2\pi ft} + e^{-j2\pi ft}}{2} \tag{2.11}$$

$$\sin(2\pi ft) = \frac{e^{j2\pi ft} - e^{-j2\pi ft}}{2j} \tag{2.12}$$

By substituting Equations (2.11) and (2.12), the final form is obtained:

$$y(t) = A\left(\cos(\phi)\cos(2\pi ft) - \sin(\phi)\sin(2\pi ft)\right) \tag{2.13}$$

Which may be expressed in the I/Q form:

$$y(t) = I(t)\cos(2\pi ft) - Q(t)\sin(2\pi ft) \tag{2.14}$$

Where:

$$I(t) = A\cos(\phi) \tag{2.15}$$

$$Q(t) = A\sin(\phi) \tag{2.16}$$

The original wave is now expressed in terms of two separate components which have the same frequency but are $\pi/2$ radians out of phase of one another. The In-phase or $I(t)$ component is in phase with the cosine carrier, the Quadrature or $Q(t)$ is therefore $\pi/2$ radians out of phase with the carrier. The benefit of this decomposition is that each component may be individually amplitude modulated and then combined to produce a carrier which is arbitrarily modulated with phase and/or amplitude [40]. How the two component waveforms combine to create the carrier waveform can be seen clearly when the amplitude of either the I component or Q component is set to 0, which can be seen in Figure 2.3.

This example illustrates visually how the addition of the two components results in modulation of the carrier, when the amplitude of either component is set to 0, the carrier becomes equivalent to whichever component is non-zero, which results in a phase shift of $\pi/2$ radians by definition. This example is not representative of an actual modulation scenario as a component with a 0 valued amplitude is never used, it is purely to demonstrate the simplest case of I and Q components combining to form a modulated carrier. A second example illustrates how two non-zero components may combine to result in a modulated carrier, this is shown in Figure 2.4.

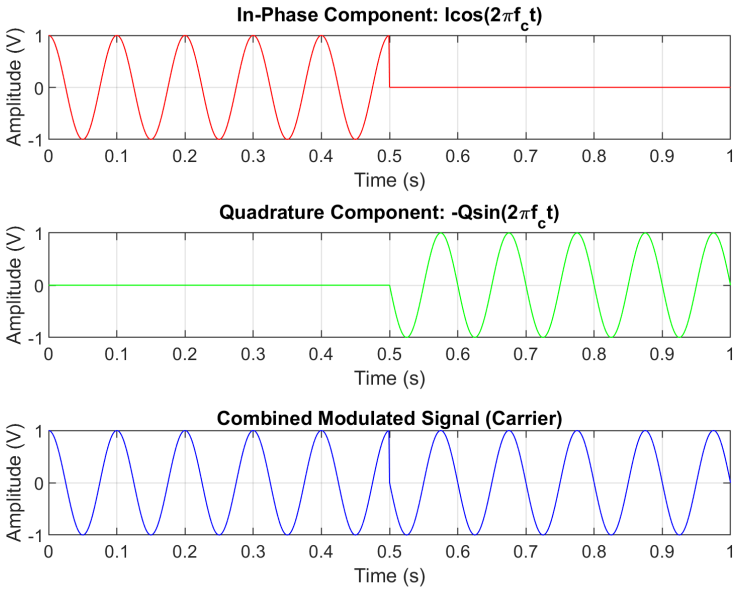


Figure 2.3: *How the In-Phase and Quadrature Components Combine to form the Carrier Wave*

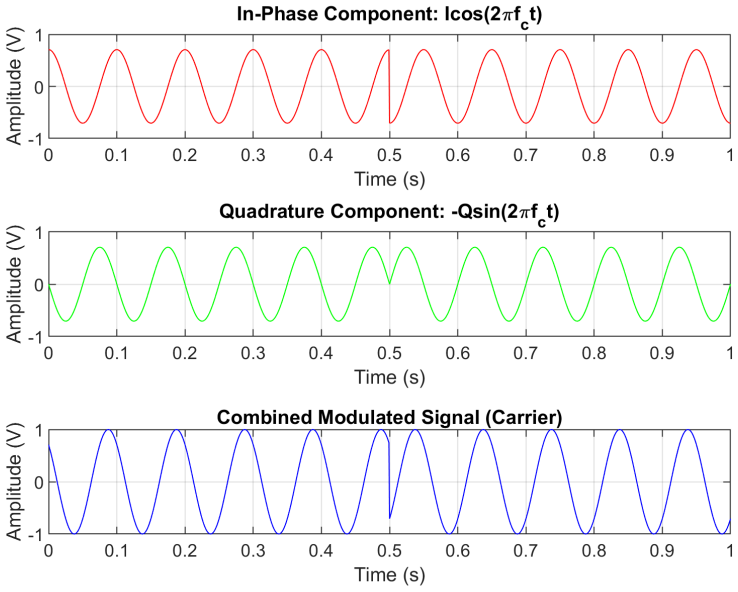


Figure 2.4: *How the In-Phase and Quadrature Components Combine to form the Carrier Wave when Components are Non-Zero*

Figure 2.4 illustrates the I and Q components transitioning from an amplitude of 0.707 to -0.707 at $t = 0.5s$, corresponding to a π radian phase shift in the carrier signal. The formulae to obtain the exact phase and amplitude of the carrier are given by the equations:

$$\phi = \text{atan2}\left(\frac{Q}{I}\right) \tag{2.17}$$

$$A = \sqrt{I^2 + Q^2} \tag{2.18}$$

Where ϕ is the carrier phase, A is the carrier amplitude, Q and I are the amplitudes of the Quadrature and In-phase components. Critically, the atan2 function is required rather than the traditional arctan function, as it extends the range from $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ to $[-\pi, \pi]$. In the case of this example, the amplitudes of the I and Q components begin at 0.707 at $t = 0$ s. Plugging these values into the provided equations results in the following:

$$\phi = \text{atan2}\left(\frac{0.707}{0.707}\right) = \frac{\pi}{4} \text{ rads} \tag{2.19}$$

$$A = \sqrt{0.707^2 + 0.707^2} \approx \sqrt{0.5 + 0.5} = 1 \tag{2.20}$$

Following the amplitudes of the I and Q components changing to -0.707 at $t = 0.5$ s, the phase and amplitude of the carrier becomes:

$$\phi = \text{atan2}\left(\frac{-0.707}{-0.707}\right) = \frac{5\pi}{4} \text{ rads} \tag{2.21}$$

$$A = \sqrt{(-0.707)^2 + (-0.707)^2} \approx \sqrt{0.5 + 0.5} = 1 \tag{2.22}$$

It is difficult to intuit exactly how changes to the amplitude of I and Q components will affect the carrier waveform. For this reason, a visual representation of the carrier waveform is instead used, this concept is discussed in the following section.

2.4 The Constellation Diagram

The constellation diagram is visual representation of the phase and amplitude combinations which a modulation scheme may take. It displays the signal as a two-dimensional scatter diagram on the complex plane at signal sampling instants. The example I and Q amplitudes which were used for demonstration in Figure 2.4 may be plotted on the constellation diagram, this can be seen in Figure 2.5.

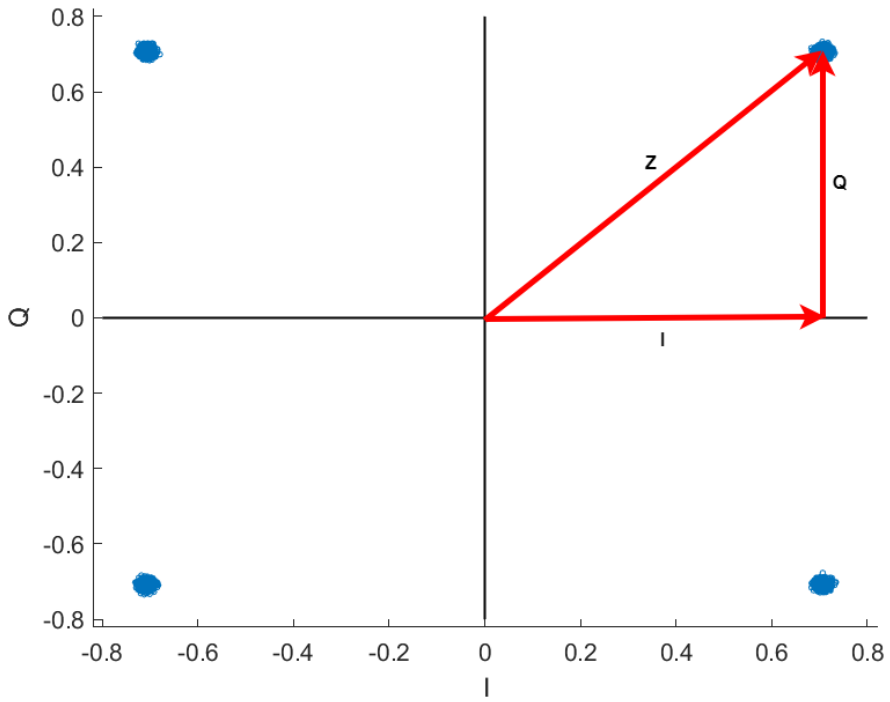


Figure 2.5: *The QPSK Constellation Diagram at 40dB SNR with the I, Q, and Z Phasor Relationship Overlaid*

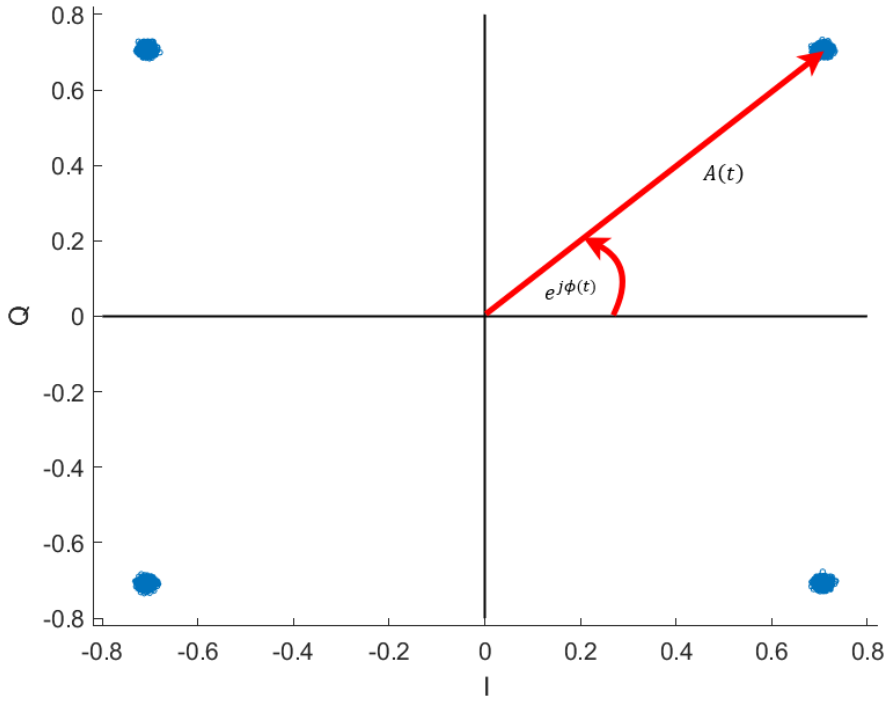


Figure 2.6: *The QPSK Constellation Diagram at 40dB SNR with the Complex Polar Representation Overlaid*

Figure 2.5 shows the constellation diagram of QPSK at 40dB SNR, the blue clusters in each quadrant of the diagram represent a different phase which a QPSK signal may take. The red arrows provide a phasor for how the I and Q values combine to result in the complex number of Z, the relationship may be expressed with Equation 2.23.

$$Z = I + jQ \tag{2.23}$$

Utilising the conversion formulae defined in 2.17 and 2.18, the polar representation of this relationship may be obtained, thus providing the amplitude and phase of the carrier signal, which is shown in Figure 2.6. Each constellation point therefore represents both the relationship between I and Q as well as the magnitude and phase of the resultant carrier.

The constellation diagrams shown in Figure 2.5 and Figure 2.6 have shown how combinations of I and Q components of consistent amplitude may provide a phase modulated carrier. According to Equations 2.17 and 2.18, by scaling the values of the I and Q components by equivalent amounts, the carrier signal amplitude may be varied without affecting the phase, thus the carrier signal may be modulated with both amplitude as well as phase. Using these techniques, engineers may design modulation schemes with constellation diagrams of arbitrary constellation point positioning. Examples of various constellation diagrams modulated with both amplitude and phase are shown in Figure 2.7. However, the constellation point positions are never arbitrary, they are generally placed in positions which maximise the distance between constellation points in the constellation diagram, the following section explains why this is the case.

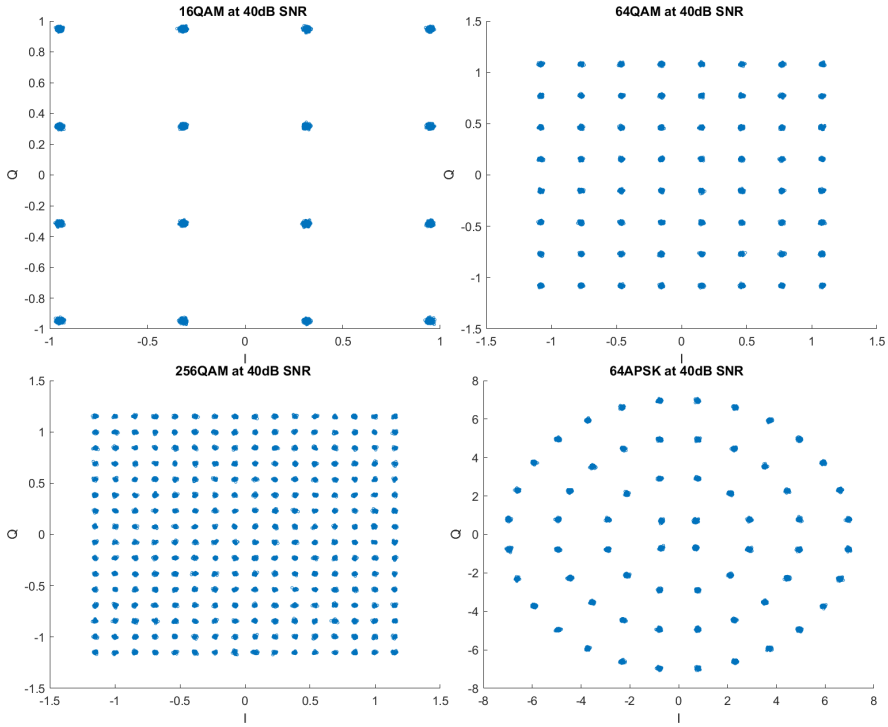


Figure 2.7: *A Collection of 4 Constellation Diagrams Modulated with both Amplitude and Phase at an SNR of 40dB*

2.5 Constellation Diagrams and Noise

This final background subsection explores the effects of noise upon the appearance of the constellation diagram, this information is then used to explain the importance of constellation point positioning and how modulation order relates to SNR robustness. Thus far in this thesis, the only constellation diagrams displayed in figures have been created with data at an SNR of 40dB, the constellation points have therefore had the appearance of a densely spaced cluster. SNR, defined in Equation 2.24 where A_{signal} and A_{noise} represent the amplitude of the signal and noise respectively, provides a measure of the ratio between the amplitude of the signal and noise.

$$\text{SNR}_{\text{dB}} = 20 \log_{10} \left(\frac{A_{\text{signal}}}{A_{\text{noise}}} \right) \tag{2.24}$$

Noise is an inherent characteristic of any communications channel, it introduces random deviations from the ideal scenario. An increase in the amplitude of noise results in the dispersion of the received symbols around the intended constellation points, manifesting as a "cloud" of points around the intended ideal constellation point position. The greater the amplitude of the noise with respect to the signal (or the lower the SNR), the greater the dispersion of received symbol points. Figure 2.8 illustrates this phenomenon by displaying 16PSK at an SNR of 30dB and 15dB.

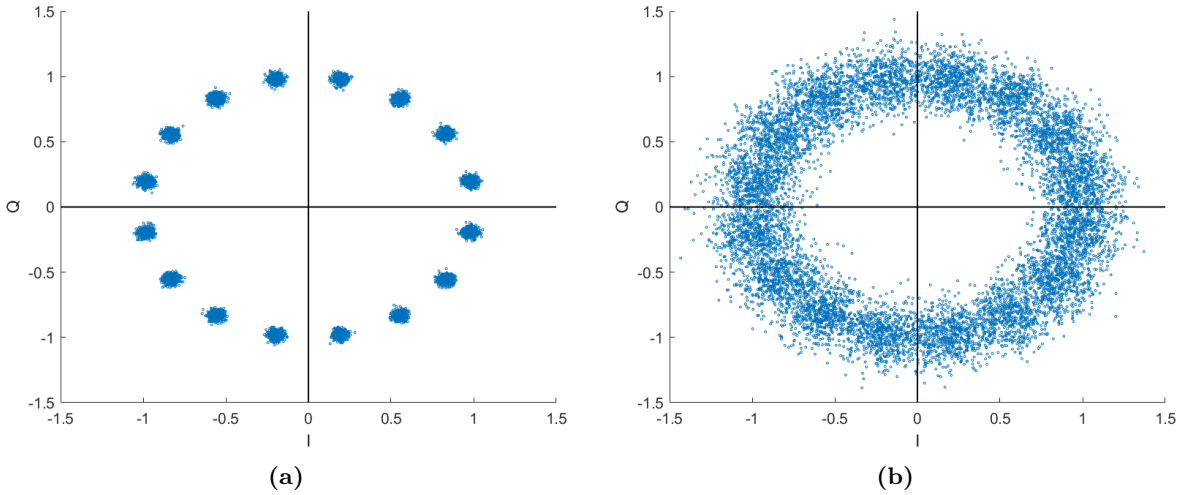


Figure 2.8: *16PSK Constellation Diagrams at SNRs 30dB (a) and 15dB (b)*

In Figure 2.8 (a) the constellation points remain well defined, there are still 16 clear clusters of symbol points. In Figure 2.8 (b) the low SNR has increased the dispersion of symbol points to such a degree that constellation points begin to overlap, they are no longer clearly defined. Demodulation hardware at the receiver is likely to produce errors due to the overlap of constellation points should the signal shown in Figure 2.8 (b) be received.

Figure 2.9 displays 16QAM at an SNR of 15dB. Despite having an equivalent SNR to Figure 2.8 (b), the constellation points still exist in clearly defined regions with minimal overlap exhibited. By utilising both amplitude and phase modulation, greater separation of

constellation points is achieved and therefore the 16QAM modulation scheme has a stronger robustness to the effects of noise.

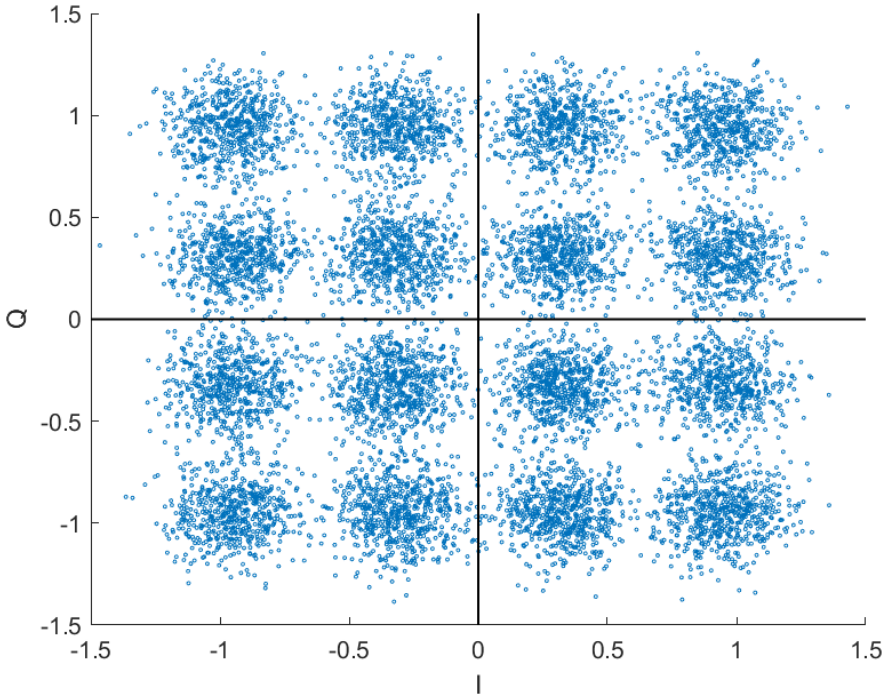


Figure 2.9: *The 16QAM Constellation Diagram at an SNR of 15dB*

While this example has demonstrated that QAM modulation schemes are inherently more robust to noise distortion than PSK modulation schemes of equivalent order. Higher-order modulation schemes inherently have a higher density of constellation points, the SNR at which constellation points begin to overlap is therefore greater than that of lower-order modulation schemes as the distance between constellation points is reduced. To illustrate, Figure 2.10 displays the 1024QAM constellation diagram at an SNR of 40dB and 30dB.

In Figure 2.10 (a) the 1024QAM constellation diagram is shown to have clearly defined constellation points. Conversely, in Figure 2.10 (b) the well defined constellation point spacing is lost and significant overlap between constellation points can be seen. In Figure 2.8 30dB SNR was used to demonstrate 16PSK with well defined constellation points, yet in this case at an SNR of 30dB the characteristic grid formation of 1024QAM is lost. 1024QAM is therefore less robust to noise than 16QAM.

These examples have illustrated how utilising both amplitude and phase modulation provides increased SNR robustness in comparison to utilising only amplitude or phase. Furthermore, it has also been shown how high-order modulation schemes have reduced noise robustness due to the density of constellation point positioning. This short background section has now provided the necessary foundations for the remaining chapters in this thesis to be understood. The next chapter provides an in-depth review of the literature in the fields of AMC and NDA SNR estimation.

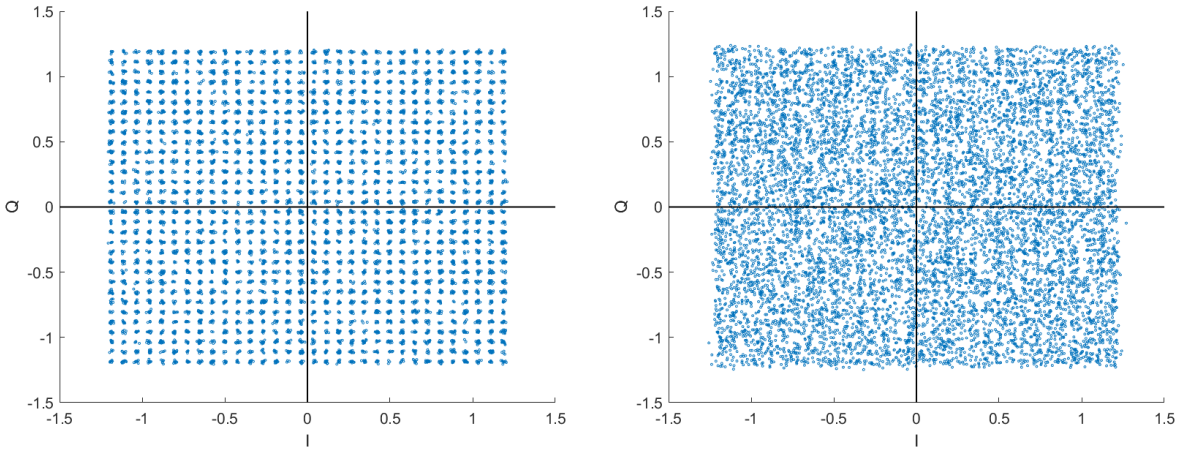


Figure 2.10: *The 1024QAM Constellation Diagrams at SNRs 40dB (a) and 30dB (b)*

Chapter 3

Literature Review

Following the discussion of the background concepts on which AMC systems rely, the methods which have previously been investigated for the purposes of AMC may be compared within this context. The peak accuracy, the lowest SNR at which the peak accuracy is achieved, and the SNR at which the accuracy becomes equivalent to a random guess are the 3 primary metrics will be considered when comparing the performance of each AMC method.

The proposed AMC system targets resource-constrained CR applications, where minimising hardware footprint, power consumption, and latency is critical for real-time deployment. Thus, the primary focus is to realise a classifier that matches state-of-the-art performance while drastically reducing utilisation. Following the discussion of the different techniques which can be used to achieve AMC, the focus will be narrowed to the strongest AMC systems which have been implemented in hardware. Comparisons drawn in this section will use each performance metric described in the above paragraph whilst also incorporating comparisons between the various statistics which are required to characterise hardware implementations, such as number of Flip-Flop (FF)s, Look-Up Table (LUT)s, Digital Signal Processing (DSP) slices, Blocks of Random Access Memory (RAM), power consumption, and latency. By utilising these metrics for comparison, an analysis of the techniques which have the potential for efficient and accurate CR implementation is obtained, and the strongest implementations which the proposed system must outperform are identified.

In addition to modulation classification, CR systems require robust SNR estimation to dynamically adapt to channel conditions, necessitating an evaluation of blind estimation techniques. This literature review will cover this field of research and use MSE as the primary metric for comparison. However, just as with AMC there are more factors to consider than solely the accuracy of a particular method. Many techniques which will be discussed display strong accuracy yet only achieve this accuracy within a particular SNR range, outside the bounds of this range, they exhibit asymptotic behaviour (e.g. error floors or divergence in estimation accuracy). Similarly, many algorithms are designed for usage on single modulation schemes, making them unviable for CR applications where the modulation scheme may vary. While some algorithms may be reconfigured to handle a range of modulation schemes, the scope of the range which they can handle may also be a limiting factor. The strongest NDA SNR estimator will therefore be the system which demonstrates strong performance over a large SNR range and on a large variety of modulation schemes.

This literature review is structured as follows: Sections 3.2 and 3.3 examine software-based

AMC approaches, diving them into feature-based (Section 3.2) and DL-based approaches (Section 3.3) to contrast the characteristics and performance of each paradigm. Section 3.4 applies a critical analysis of each software approach and compares them in terms of the criteria outlined at the outset of this review. Section 3.5 applies the knowledge obtained from this comparison and explores the merits of the available hardware implementations, discussing both Field-Programmable Gate Array (FPGA) and Application Specific Integrated Circuit (ASIC)-based solutions and comparing their respective advantages and disadvantages in terms of both accuracy against SNR and hardware implementation characteristics. This analysis is used to identify the strongest examples of hardware implemented AMC algorithms which then are used for comparison later in this thesis. Section 3.6 concludes this review by exploring the techniques for NDA SNR estimation which may be found in the literature, each technique is compared in terms of estimation accuracy within the context of SNR range of accurate estimation and the variety of applicable modulation schemes.

Ultimately, the findings obtained from this literature review are used to identify a technology which offers a great deal of potential towards the realising an efficient yet effective AMC and NDA SNR estimation system.

3.1 Dataset Differences Across Each Work

Before the literature review proper may begin, it is important to understand that the accuracy obtained by each model is not solely dependent on the quality of the implementation. Equally as important as the model structure is the choice of modulation schemes which are included in the training and testing datasets. Depending upon the set of utilised modulation schemes, the achieved classification accuracy may be either optimistic or pessimistic in comparison to another work. There are therefore several choices of modulation scheme selection to identify when comparing model performance.

The first is the order of the modulation schemes included within the dataset. When comparing the performance of AMC systems, the inclusion of high-order modulation schemes within the dataset universally imposes a performance penalty on the system. This is particularly true when works opt to include high-order signals such as 256QAM which are particularly vulnerable to noise due to the increased density of constellation points. Therefore, systems which demonstrate the ability to classify high-order modulation schemes such as 256QAM will be judged more favourably in comparisons.

Secondly, some works [19,50,51,58,70] do not include differing modulation schemes of equivalent order within the dataset, for example 16PSK and 16QAM. They therefore do not demonstrate the ability to distinguish between equivalent order modulation schemes and will be judged less favourably in comparisons.

Finally, the number of different modulation schemes included in the dataset should also be considered. A model with the ability to classify a wide range of schemes will be judged more favourably in the comparison text than one which only includes a limited set of schemes in the test dataset.

Another point which must be emphasised with respect to the number of classes in the test dataset is that many works opt to use the open-source datasets RadioML.2016.10A [41] and RadioML.2018.01A [41]. Works which opt to use these datasets make for more direct comparisons as each system is trained and tested on the same dataset, thus model structure

the only differentiating factor. Additionally, RadioML.2018.01A features an extensive number of classes such as 5 orders of QAM from 16 to 256, APSK orders 16 to 128, and PSK orders from 2 to 32. This results in a dataset which consists of high-order modulation schemes, schemes of equivalent order, schemes of the same format, and a large set of modulation schemes. However, the datasets also include a number of analogue modulation schemes such as Amplitude Modulation - Single Side Band - With Carrier (AM-SSB-WC), Amplitude Modulation - Double Side Band - With Carrier (AM-DSB-WC), and 4ASK etc. This thesis is not concerned with the performance of systems when classifying these analogue modulation schemes, only digital modulation formats will be considered. When these works provide results for their obtained classification accuracy, this accuracy also includes the accuracy of the system on the analogue modulation schemes. It is impossible to obtain the performance of the system if only digital schemes were included, instead an estimation of how the obtained accuracy may be affected if these modulation schemes were discounted will be provided.

Universally, every system which employs these datasets cannot distinguish between Wideband Frequency Modulation (WBFM) and AM-SSB-WC. Even at high SNRs the accuracy of the system on one of these modulation schemes will be 0 and the other will be 100, therefore the peak performance at high SNRs of models which use RadioML datasets will be assumed to slightly higher than what is reported. Conversely, there are many low-order analogue modulation schemes such as On-Off Keying (OOK), 4ASK, and 8ASK, which models classify with high accuracy at low SNRs. Consequently, the performance of models which use RadioML datasets will be judged to have a slightly optimistic performance in the low SNR range. Where possible, high and low SNR accuracy on only digital modulation formats will be calculated from provided confusion matrices and given in the comparison text.

3.2 The Feature-Based and Deep Learning AMC Paradigms

Having established the importance of high accuracy, inter-order classification, inter-format classification, robustness to channel impairments, and the maximum order of modulation scheme included in the dataset, this section explores the 2 dominant paradigms of AMC from the perspective of software implementation. Although this thesis is aimed towards the realisation of an efficient hardware implementation, to ensure that all the best performing systems are explored, it is first important to explore the techniques which have shown to be effective in software but have not necessarily received attention in terms of hardware implementation.

There are 2 predominant paradigms of AMC, the first uses expert features in conjunction with classification models to distinguish between modulation schemes. This technique was the first method of AMC to be proposed as modern DL model structures, optimisation techniques, and the hardware upon which they rely were yet to developed. The usage of "expert features" refers to the process of the engineer identifying statistical characteristics of an input dataset which can be obtained algorithmically and then used to differentiate between classes. A classification model structure is employed to provide the discriminatory functionality, it learns how each combination of features relates to the classes which they are extracted from and thus learns to classify future feature inputs.

Conversely, modern DL methods offload feature generation to the classifier directly, eliminating the requirement for features to be identified or designed by the engineer. Through this

process it is thought that more information can be learned by the system as it is provided as much information as possible. Furthermore, the features which are learned may not be able to be mathematically calculated as they are perhaps more akin to pattern recognition and reasoning operations than algorithmic operations. It is through this information maximisation and the learning of more abstract features which DL systems aim to improve upon the traditional feature-based classifiers.

Figure 3.1 illustrates the differences as well as the similarities between each approach. Both methods have largely the same structure with the key differentiator being that the feature-based method extracts features with human-designed algorithmic cores whereas the DL system learns its own features during the training process, each method requires a classification algorithm following the feature processing step.

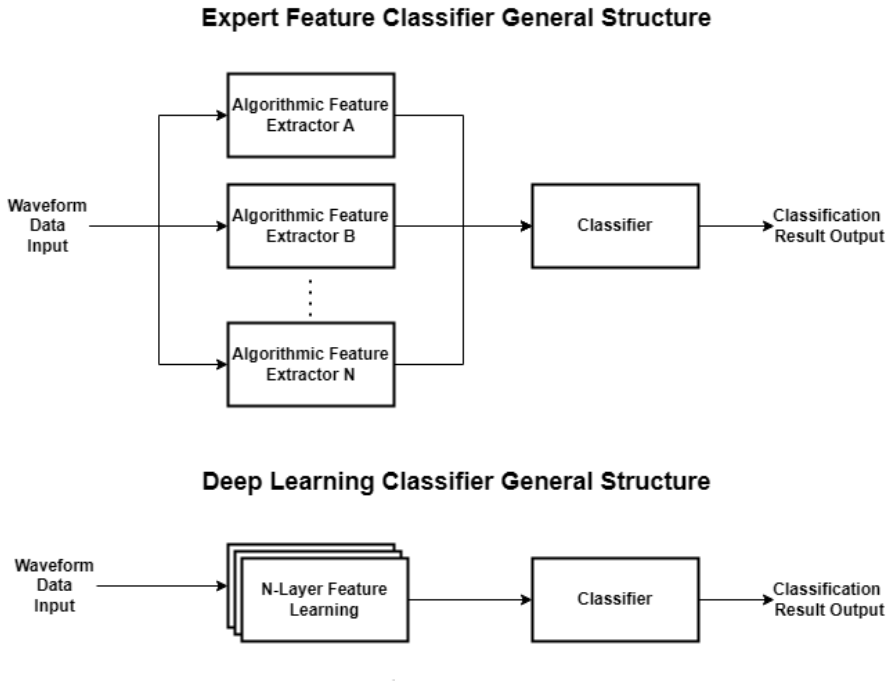


Figure 3.1: *The General Structures of Feature-Based and DL Classifiers*

With the modern AI revolution underway, it may be tempting to assume that the discriminatory power offered by DL models is superior to the older feature-based approaches. However, it may be the case that well designed features in conjunction with appropriately chosen classifier structures can offer equivalent performance with a major reduction in hardware requirements. This question is the principal concern of this literature review and indeed this thesis as a whole.

The broad outline of both dominant methodologies has now been discussed, the detail of the implementations of each technique will now be investigated. This investigation will begin by examining expert feature classification, the following section concerns the types of features which can be employed, then the various classifier structures will be presented, finally the obtained performance from utilising combinations of varying features and classification structures will be compared and evaluated with the aim of determining which strategies show

promise for implementation in mobile CR systems.

3.2.1 An Exploration of Expert Features

The principal component of feature-based classification is naturally the choice of the features themselves, early approaches used statistical measures based upon information extracted from the frequency and bandwidth of the signal. Later works progressively introduce a greater number of features which utilise statistics derived from the instantaneous amplitude and phase. Modern works which have been published in recent years use a combination of the features which have been shown to result in strong discriminatory capabilities [46,50].

The earliest example of AMC which could be found in the literature is the method introduced by Gardner et al. which proposed the use of cyclostationary features as a method of classifying orders of PSK signals [42]. However, the proposed work is purely theoretical. A signal is considered cyclostationary if its signal properties vary periodically with time, by applying the Cyclic Autocorrelation Function (CAF), distinct patterns for various modulation types are obtained. This method has been shown to be effective in recent practical implementations such as in the work by Câmara et al. [43]. The features have been found to be particularly robust to noise interference as Additive White Gaussian Noise (AWGN) is typically stationary, therefore it does not contribute to cyclic frequencies.

Assaleh et al. [44] proposed the usage of the standard deviation and the mean value of the differential of the instantaneous frequency, as well as the standard deviation and peak values of the bandwidth. Of these features, the only one to see common usage in later works is the standard deviation of the instantaneous frequency as it was found to be highly discriminative.

Nandi and Azzouz [45] expanded the feature set to include statistics derived from the instantaneous frequency, phase, and amplitude. The authors propose a plethora of features which are: the maximum value of the spectral power density of the normalised-centered instantaneous amplitude, the standard deviation of the nonlinear component of the instantaneous phase (both absolute and direct values measured in nonweak segments), a ratio which measures the spectrum symmetry of the RF signal, the standard deviation of the absolute value of the normalised centered instantaneous amplitude of a signal, the standard deviation of the absolute value of the normalised instantaneous frequency of a signal, the standard deviation of the normalised-centered instantaneous amplitude in the nonweak segment of a signal, and the kurtosis of the normalised instantaneous amplitude and frequency.

A key contribution of the work is the usage of features derived from the spectral power density. This statistic provides information about the frequency content of the amplitude variations, this feature sees frequent use in more modern AMC work. However, of the set of utilised features the 2 values which have particular importance are the 2 kurtosis statistics. These statistics are an early example of the usage of this statistical moment for AMC purposes. Kurtosis is referred to as the fourth-order moment, it provides a measure of the "tailedness" of a distribution, higher values imply a sharper peak and large tails, low values imply a flatter peak and thinner tails, a value of 3 results from a normal distribution.

Saharia et al. [46] use the same set of features as Nandi and Azzouz with the addition of skewness which is the third-order moment. They use a feature selection algorithm to determine which features have the highest importance upon the classification result. They find that the spectral density has the highest importance, followed by the kurtosis, skewness, and standard deviation of the instantaneous frequency and amplitude. The standard deviation of

the instantaneous phase is found to have the least importance upon the classification result for their dataset and model structure.

The usage of kurtosis by Nandi and Azzouz marked a shift towards leveraging higher-order statistical measurements to capture more nuanced characteristics of the modulated signals, this paved the way for the adoption of moments and cumulants, which are now used near ubiquitously in time-series feature-based AMC algorithms. These statistics provide a general description of the probability distribution of a signal. The nth moment of a random variable provides the average of the variable raised to the nth power. The first central moment of a random variable is the mean, the second is the variance, the third is the skewness, and the fourth is the kurtosis.

Cumulants are closely related to moments as they are derived from the logarithm of the characteristic function (which is related to the moment generating function). The first and second cumulant are equal to the first and second moment, however it is with the third and fourth cumulant that the largest advantage is gained. The third, fourth, and higher-order cumulants of a Gaussian distribution are equal to 0, this means that utilising these metrics can provide features which are robust to the effects of AWGN, theoretically improving the accuracy of an AMC system in low SNR scenarios. Noise sources which are non-Gaussian such as interference are not influenced by this robustness. Figure 3.2 illustrates the characteristics of waveforms which each order of moment describes.

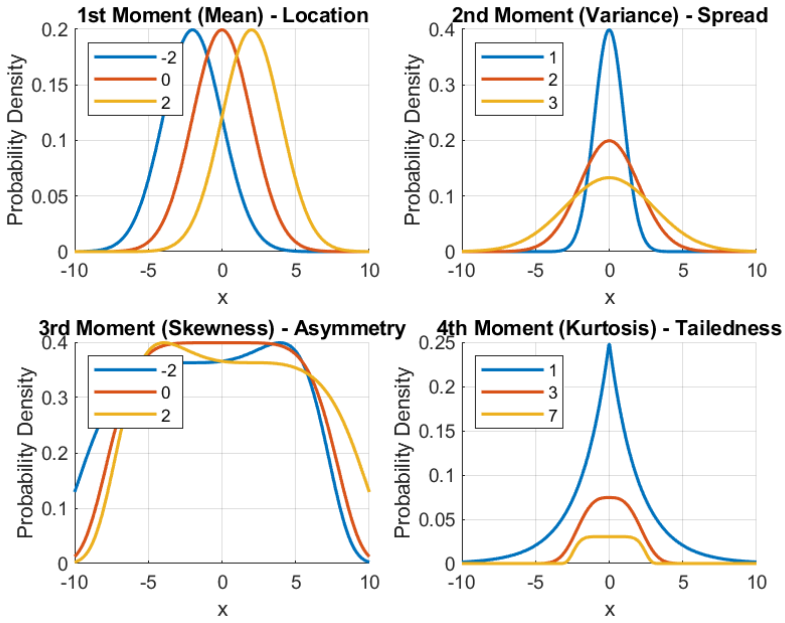


Figure 3.2: *The Waveform Characteristics Described by the 1st to 4th Order Statistical Moments*

Swami et al. [47] proposed an AMC system using only cumulants as features. They utilised variations of the fourth-order cumulant, denoted as C_{40}, C_{41}, and C_{42}. These cumulants describe various aspects of a signal's properties, C_{40} describes the circular symmetry, C_{41} the asymmetry, and C_{42} the tailedness similarly to kurtosis. Crucially they also utilised the

variance of C_{40} and C_{42} which allowed them to capture the variability of these cumulants over time. Each of these values are found by using a complex input of both the phase and amplitude data. This provided additional discriminatory information as some modulation schemes may be more susceptible to channel impairments and would therefore exhibit higher variance in these cumulants.

Zhou et al. [48] expand upon the work by Swami et al. by including even ordered cumulants from the second to the eighth. This approach aims to capture a more complete statistical description of modulated signals. While the second-order cumulant is equal to the variance, the higher-order cumulants can capture the finer details of the signal's distribution. The sixth-order cumulants are more sensitive to extreme values or outliers in the signal, for the context of AMC this can help differentiate between modulation schemes with subtle differences in their constellation diagram. The eighth-order cumulants provide even higher sensitivity to these details, they are said to be most useful when discriminating between higher-order constellations, in this case between 16QAM and 64QAM. The authors perform feature selection across all the included cumulants to determine which provide the strongest predictive power. A variety of feature selection methods are utilised, they include Genetic Algorithm (GA), Binary Particle Swarm Optimisation (BPSO), Co-evolution Binary Particle Swarm Optimisation with Multiple Inertia Weight Strategy (CBPSO-MIWS), and Recursive Feature Elimination (RFE). Each feature selection method results in a different set of features being selected, but in general all algorithms prioritise the fourth, sixth, and eighth-order cumulants. There is only one example where a feature selection algorithm finds a second-order cumulant to be a viable feature and this is when CBPSO-MIWS selects C_{21}, however when the number of utilised features is reduced to only 3, the algorithm opts to not select C_{21}, instead choosing C_{62}, C_{63}, and C_{80}. C_{63} and C_{80} are found to have the highest discrimination ability as they are the most frequently selected features when the number of utilised features is set to 3.

Another example of work which uses high-order cumulants is that of Wong et al. [49]. However, the authors use fourth and sixth-order cumulants, opting not to include second and eighth-order cumulants.

Alarabi et al. [50] use the second, third, and fourth-order moment of the instantaneous amplitude and phase. They also utilise the entropy of both amplitude and phase. The entropy is a measure of the uncertainty or randomness of a signal's amplitude and phase. This metric may be useful in the context of AMC as a modulation scheme with a high amplitude entropy may utilise amplitude modulation, similarly a high entropy of the phase values indicates phase modulation.

The features discussed thus far have all been extracted from the I/Q waveform in either the time or frequency domain. Information may also be extracted from a signal's constellation diagram. This approach can offer the advantages of the ability to distinguish modulation types with similar spectral characteristics. There are several features of the constellation diagram which may be extracted for usage in classification tasks, these include the number of constellation points, the density of constellation points, and arrangement of the constellation points.

Clustering algorithms have been identified as an effective method of constellation diagram feature extraction. Zhang et al. [37] use the clustering algorithm DBSCAN to extract the number of constellation points of various orders of QAM from 4 to 256. DBSCAN sees usage in

this context as it is a non-parametric clustering algorithm, meaning that it finds an arbitrary number of clusters within a dataset. This is contrary to parametric clustering algorithms such as k-Nearest Neighbours (kNN) which require the expected number of clusters to be specified. In the context of constellation diagram feature extraction this is advantageous as the number of constellations is feature which is to be extracted.

Zhao et al. [38] use a similar method with the additional feature of the number of core points identified by the DBSCAN algorithm. The core points metric was used as an additional feature to discriminate between 32QAM and 64QAM, this metric acts as proxy for density as for a given dataset size a constellation diagram with a greater number of constellations will naturally have a reduced amount of core points per constellation or cluster. In this case QPSK, 8PSK, 16QAM, 32QAM, and 64QAM were the modulated signals in the dataset.

Wang et al. [51] employ a similar approach but opt to use the Subtractive Clustering algorithm rather than DBSCAN. The principle is broadly similar in that the number of constellations is the metric to be obtained and used for classification. However, in this case the algorithm identifies the centres of clusters (in this case the constellation points) rather than the clusters themselves.

Many of the most popular and effective features have now been discussed, the following section details the variety of classifier structures which have been employed as well as the decision-making process behind selecting an appropriate model.

3.2.2 The Choice of Classifier Structure for AMC

Various classifier structures have been used to achieve feature-based AMC. Unlike DL, the accuracy achieved with this paradigm of AMC does not necessarily scale with classifier power and complexity. Rather, it is the feature representation which fundamentally defines the problem space in which the classifier operates [52–54]. The extracted features constitute the basis for distinguishing between classes, meaning that suboptimal feature selection can result in significant overlap between class distributions within the feature space [53]. Overlap such as this inherently limits the effectiveness of any chosen classifier. Conversely well-chosen features result in strong separation of classes within the feature space, in such cases even the simplest of classifier structures may be capable of finding decision boundaries which result in a high level of accuracy [55, 56]. An exception to this is in the cases where there is strong separation of classes, but they are not linearly separable, in this case a more complex classifier which can find non-linear decision boundaries would be advantageous [54]. This problem is illustrated in Figure 3.3 where 3 plots show an example of classes without strong separation, linearly separable classes with strong separation, and classes with strong separation which are not linearly separable.

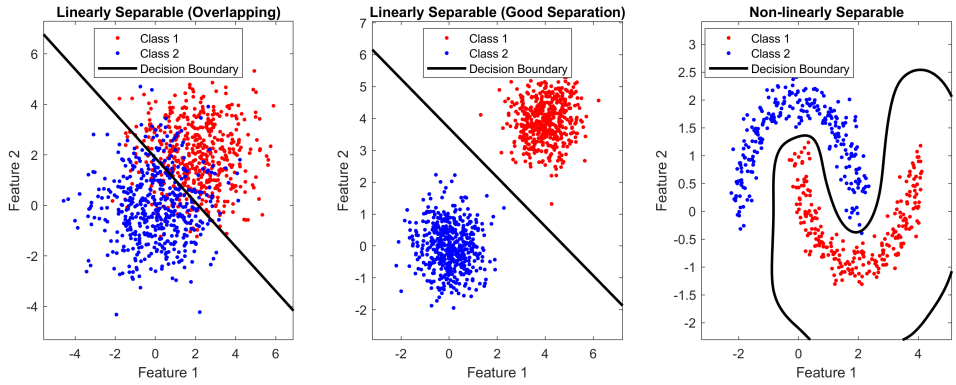


Figure 3.3: *Examples of Linearly Separable and Non-Linearly Separable Data with Appropriate Decision boundaries*

In the first plot of Figure 3.3 the classification is performed with a linear decision boundary, however as there is overlap between the classes there is no classifier which could differentiate between each class with perfect accuracy, regardless of complexity. In the second plot there is strong separation, any classifier should be capable of reaching perfect accuracy in this case. In the final plot there is strong separation between the classes, but they are non-linearly separable, the linear decision boundary cannot capture the form of the data and thus the classification result will not be ideal. It is however unlikely that any set of selected features would be linearly separable without overlap, especially considering that the influence of noise can obfuscate the information held within a waveform which can therefore inhibit accurate feature extraction.

Another situation where a more complex classifier is advantageous is to accommodate a larger number of features and classes. Increasing the number of features increases the dimensionality of the feature space and introduces more complex relationships between features and classes [52, 53]. Likewise, increasing the number of classes can require a greater number of more intricate decision boundaries. Thus, increasing both aspects means that there is a requirement for intricate decision boundaries in high dimensional feature-space, something which complex classifiers such as Support Vector Machines (SVM)s and Artificial Neural Network (ANN)s are better suited to provide [52, 53]. To illustrate this point Figure 3.4 shows 2 feature spaces which resulted from data used in the work performed in this thesis. In each case the decision boundaries are found by a linear SVM and a more complex non-linear SVM.

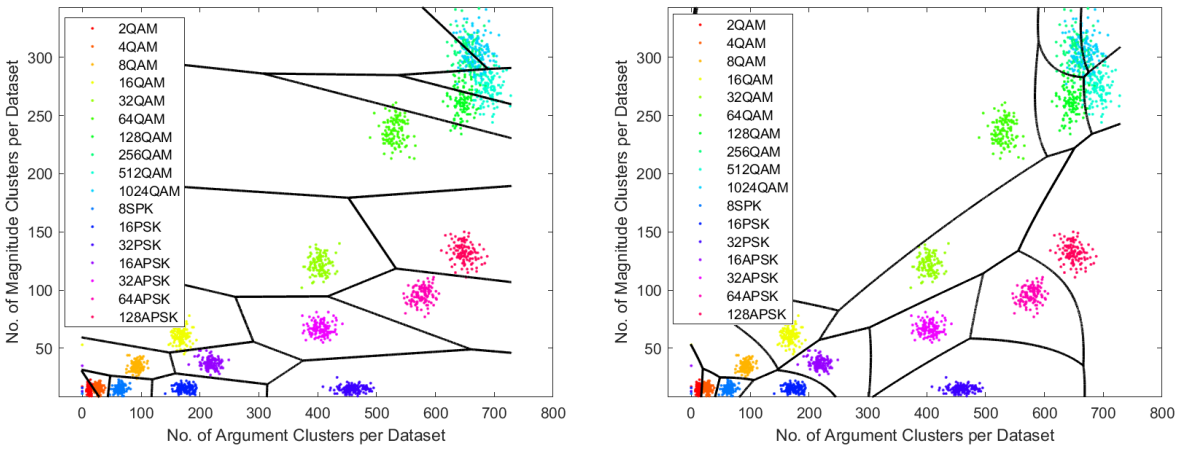


Figure 3.4: *Features Extracted from a Set of Modulation Schemes, Classified with (a) Linear Decision Boundaries and (b) Non-Linear Decision Boundaries*

Paying particular attention to the overlapping feature clusters at (700,300), it may be seen that while these feature clusters overlap, there are still distinct regions where each class is of a higher density, in (a) the linear decision boundaries fail to capture this relationship as the decision boundary contours often pass through the regions of high class density, conversely the more complex non-linear boundaries in (b) capture the regions where each class cluster resides with greater accuracy. The majority of the remaining feature clusters exhibit a good quality of separation, therefore both methods find good decision boundaries. The overall accuracy achieved in (a) will suffer a slight performance loss in comparison to (b) due to the high-order QAM overlap however.

To summarise this discussion, the optimum choice of classifier is dependent on the quality of feature separation, the dimensionality of the feature space, and the number of classes. While models of a lower complexity may perform well when there is a small number of highly separated classes which exist in low-dimensional feature space, their usage will impose a performance reduction should any of these factors cease to be the case. Higher complexity classifiers offer increased generality.

3.2.3 Classifier Structures Utilised for Feature-Based AMC

Early works in feature-based AMC often employed simpler classifier models due to computational limitations and the limited numbers of low-order modulation schemes which were included in the dataset. For example, Assaleh et al. [44] used a Decision Tree (DT) with fixed thresholds. This approach is inherently limited to creating piecewise linear decision boundaries [53] and as illustrated in Figure 3.3, it would struggle with overlapping class distributions and non-linearly separable data. However, given the small number of classes which were considered, and the high SNR of the test data, the simple approach proved effective and achieved high classification accuracy.

As feature sets became larger and the number of classes increased, researchers began exploring more powerful classifiers. Nandi and Azzouz [45] compared a DT with an MLP neural network. The DT performed better with simpler classification tasks (e.g. distinguish-

ing between 2ASK and 4ASK), the MLP demonstrated superior performance when classifying a wider range of modulation schemes, suggesting a superior ability to learn more complex decision boundaries in high dimensional feature spaces.

Swami et al. [47] still employed a DT but only utilised 5 features. In this case strong performance was obtained despite the less complex classifier, suggesting that the high-order cumulant features provide linearly separable classes.

Subsequent works begin to explore more powerful classifiers to fully leverage expanded feature sets. Saharia et al. [46] compare DTs with a Random Forest (RF) classifier, RF classifiers are an ensemble of DTs which combine to provide more robust classification [53, 55]. The RF classifier consistently outperformed the DT, particularly when distinguishing between higher-orders of QAM. This work utilised a large number of features, the superior performance of the RF classifier in this case demonstrates that the more complex model was superior in the high dimensional feature space.

Wong et al. [49] used a Naïve Bayes classifier in conjunction with 4 high-order cumulants. While the obtained accuracy reached 100% at high SNRs, the accuracy deteriorated at a higher SNR than seen with other classification models. This suggests that the information captured by the high-order cumulants may not be optimally utilised by the relatively simpler Naïve Bayes classifier, potentially due to the inter-dependencies between the cumulants which violate the classifier's core assumption of feature independence [52].

Zhou et al. [48] used an SVM, a powerful classifier capable of learning both linear and non-linear decision boundaries depending upon the employed kernel function [52]. In this case Zhou et al. used a linear kernel, stating that the reason for this choice was that the high-order cumulant features were linearly separable. Excellent accuracy was achieved, demonstrating the potential of the combination of strong features and complex classifiers.

Alarabi et al. [50] also achieved strong results by utilising an ANN classifier with a large feature set of lower-order moments and entropy. They also used a dataset with many classes. The obtained accuracy is comparable to that of Zhou et al. [48] at SNRs greater than 12dB but shows more robustness to noise. Even despite the large numbers of features and classes best-in-class average accuracy was obtained, this demonstrates that a more complex classifier in conjunction with a strong set of features can maintain equivalent performance to comparatively simpler classification tasks.

The 3 works which used clustering algorithms as a feature extraction mechanism each opted for a DT classifier [37, 38, 51]. In this case each work extracted the total number of constellation points as the feature for classification, as their classifier had only a single input feature with strong separation, it is unlikely that a more complex classifier would be able to improve upon the achieved results. The case for using a DT is further enhanced by the expected value of the features being obvious, i.e. 256QAM is expected to have roughly 256 constellations, therefore manually setting thresholds is trivial and enables fine tuning to reach the desired accuracy.

This section has provided an overview of the various classifier models which have seen use for feature-based AMC, the prior section has discussed the range of employed features. Declaring an overall best performing classifier is impossible as the set and number of employed features and classes varies significantly between studies. What can be concluded from this section is that all classifier structures have been demonstrated to be effective, however more complex classification tasks can benefit from the usage of more complex models, as with

the improved performance of the ANN compared to the DT in [45], and the RF classifier compared to the DT in [46]. The systems which show a strong ability to classify a large number of classes leverage feature sets which provide strong separation in high dimensions in conjunction with a complex model to make best use of the input feature set. The next section will delve deeper into the achieved results by the strongest systems discussed here and analyse how the combinations of feature sets, modulation schemes in the datasets, and classifier model structures effect overall classification performance.

3.2.4 Feature-Based AMC Results

This section examines the results obtained by employing combinations of the various features and classifiers using works found in the literature. Per the discussion about the variety of datasets which have been used throughout each work in the literature, information about the types of datasets used by each work may be found in Table 3.1 alongside the classifier structure and utilised features. Figure 3.5 shows a comparison graph of the average accuracy against SNR of each feature-based AMC system.

Table 3.1: *Comparison of Feature-Based Modulation Scheme Classification Methods and Datasets*

Author	Max Order	No. of MS	Inter Order	Inter Format	Classifier Model	Features Used
Zhang [37]	256	4	No	Yes	DT	No. of Constellation Points
Câmara [43]	32	5	No	Yes	DT	Fractional Lower-Order Cyclic Autocorrelation Function
Saharia [46]	64	11	Yes	Yes	RF	Spectral Density A/P Standard Deviations A/P/F Skewness Kurtosis A
Swami [47]	16	4	No	Yes	DT	High and Low-Order Cumulants Variance of 4th-Order Cumulants
Zhou [48]	64	5	No	Yes	SVM	C_{42} C_{61} C_{62} C_{63} C_{80}
Wong [49]	64	4	No	Yes	Naïve Bayes	C_{42} C_{42} C_{60} C_{63}
Alarabi [50]	64	4	No	Yes	ANN	Variance A/P Kurtosis A/P Entropy A/P Skewness A
Wang [51]	256	6	No	Yes	DT	No. of Constellation Points

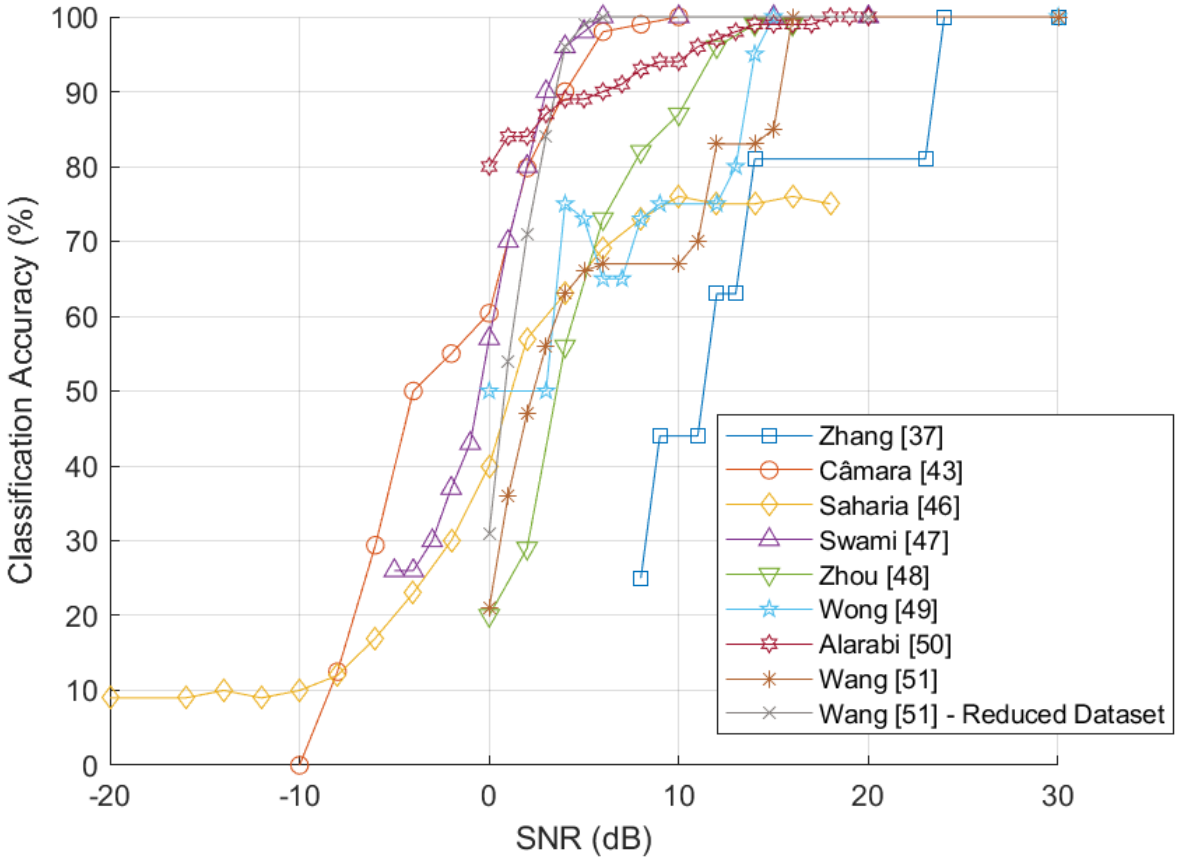


Figure 3.5: *Classification Accuracy (%) Against SNR (dB) Curves for Various Feature-Based Classifiers*

Swami et al. [47] used a DT with cumulant features to obtain the classification accuracy curve which appears to show the strongest performance of any in the comparison. However, Table 3.1 shows that the results were obtained with a limited dataset of 4 low-order modulation schemes. The results do demonstrate that cumulants alongside a DT can provide 100% classification accuracy at SNRs as low as 6dB.

Câmara et. al [43] achieve a similar performance trend to that of Swami et al. by again utilising a DT but making use of the CAF as a feature extractor. However, similar to Swami et al. the maximum modulation order on which classification is demonstrated is low, a maximum of only order 32. Therefore, while strong discriminatory power and SNR robustness is demonstrated, the fact that only low-order modulation schemes are employed in the testing data means that further investigation is required to confirm the performance of this classifier/feature combination on higher-order modulation schemes.

The remaining results were all obtained using a maximum modulation order of at least 64, making comparisons considerably more apt. A trend may be seen where the majority of systems classify with 100% accuracy above an SNR in the region of 15-18dB, below this SNR performance tends to degrade at a comparable rate.

Alarabi et al. [50] used an ANN in combination with low-order moments derived from both amplitude and phase data. The reported results achieve 100% accuracy above 18dB SNR,

but 99% accuracy is maintained down to 14dB SNR. The accuracy degrades at a reduced rate in comparison to many other systems. The accuracy is perhaps the strongest due to the superior robustness to noise, but the dataset is still limited only 4 different signals and no ability to distinguish between modulation schemes of equivalent order is demonstrated.

Zhou et al. [48] use a wide range of high-order cumulants as features to train their SVM model. 100% accuracy is achieved above 16dB SNR but the performance at lower SNRs is inferior to that of [50]. Furthermore, only 5 modulation schemes were used in the dataset with none of them being of equivalent order.

The results reported by Wong et al. [49] are once again similar to that of [50] and [48]. A small dataset of 4 different modulation schemes is used to achieve 100% accuracy above 15dB SNR, below this SNR the accuracy degrades at a quicker rate than the 2 similar systems.

The system proposed by Saharia et al. [46] is the final system which utilises a dataset with a maximum modulation order of 64. In this case the dataset is extensive as they use the opensource dataset RadioML.2016.10A [41]. The dataset consists of 11 different modulation schemes, including some of equivalent order. Additionally, this dataset includes additional impairments such as LO drift and fading, resulting in a dataset which imposes a larger challenge. The results obtained by the system perhaps reflect the usage of the more challenging dataset where a maximum classification accuracy of 75% is obtained. Despite the lower peak accuracy, it is maintained to a lower SNR than seen across all other systems that used a maximum order of 64 to obtain their results, although below 5dB SNR the rate of accuracy degradation is similar. It is impossible to predict the degree to which the peak accuracy was affected by employing the RadioML.2016.10A dataset, although inspection of the provided confusion matrix at an SNR of 16dB shows that the system mostly struggled with distinguishing 16QAM/64QAM and QPSK/8PSK, analogue signals tend to provide an overall increase in classification accuracy at this SNR. In summary, comparing this system to others is difficult as the datasets are vastly different, it may be the selection of features which result in inferior accuracy, it may be the dataset, or it may be a combination of the 2. It is unlikely that the classifier is a contributing factor as Swami et al. [47] and the 2 systems yet to be discussed used a DT model and obtained strong results, the work does show that the RF model results in a 15% increase in peak classification accuracy over using a DT.

The final 2 systems to discuss each use clustering algorithms to obtain the number of constellation points for use as features, each work also uses a DT classifier and a dataset consisting of QAM signals up to 256QAM. A major limitation of each system is that neither demonstrate the ability to classify any modulation formats other than QAM. Another inherent limitation with each system is the utilised feature of the number of constellation points, it is impossible to distinguish different modulation schemes of the same order with this feature as they would both result in the same feature values.

Zhang et al. [37] used the DBSCAN algorithm to extract features from a dataset consisting of 4 different orders of QAM. Peak accuracy is lost at the highest SNR of any system in the comparison, although this is wholly due to the inclusion of 256QAM within the dataset. Should 256QAM accuracy be discounted this method of feature extraction would maintain the peak accuracy to 14dB SNR, which is comparable to other systems which utilised 64QAM as the highest order signal. Despite this, accuracy degradation occurs at a faster rate than seen across all other systems, reaching equivalency to a random guess at an SNR of 8dB, an SNR at which all other systems still achieve accuracy rates of over 60%. DBSCAN is an algorithm

which can be difficult to optimise, as this algorithm was utilised in the proposed work in this thesis the lengthy and multivariate optimisation problem is well known to the author and discussed at length in Chapter 6. As Zhang et al. do not provide in depth information about the optimisation process it is difficult to know how the hyperparameters were selected and which modulation schemes they were optimised for. Poor hyperparameter optimisation may explain the reduced performance, especially as the other clustering-based feature extraction algorithm to be discussed achieved stronger accuracy with a greater number of modulation schemes included in the dataset.

Wang et al. [51] used the subtractive clustering algorithm to obtain the number of constellation points, which was then used a feature for classification with a DT. 6 different QAM orders were used in the dataset, 256QAM was the highest order scheme. Even with the inclusion of both 128QAM and 256QAM the peak accuracy of 100% is maintained as low as 16dB SNR, a comparable SNR to where the non-clustering feature-based classifiers lose perfect accuracy. The degradation in accuracy below 16dB also follows a similar trajectory to other systems. Inspection of Figure 5 within the text shows that should 32QAM and 128QAM be discounted from the dataset, 100% accuracy would be maintained as low as 6dB SNR which greatly surpasses the minimum SNR at which other feature-based classifiers degrade from 100% accuracy with 64QAM included in the dataset, as well as achieving 100% accuracy to the same SNR at which the low-order classifiers do [43,47]. While the discounting of 32QAM and 128QAM may seem arbitrary, the employed set of modulation schemes following the removal is identical to what is currently employed by 5G networks [1], this shows that via tailoring the set of utilised modulation schemes to the limitations of the AMC technology, the performance of the system may be maximised. As this method is demonstrated to achieve comparable accuracy to other feature-based systems even with the inclusion of higher-order signals in the dataset, and is demonstrated to lose 100% accuracy at a far lower SNR by discounting 2 modulation schemes (256QAM remains in the dataset), it can be said that this method of AMC is the best performing feature-based technique which could be found in the literature. The only limitation being the inability to distinguish modulation schemes of equivalent order.

3.2.5 Feature-Based Methods Conclusion

Systems which included modulation schemes up to an order of 64 each achieved 100% accuracy above SNRs in the region of 14 to 18dB, the notable exception being [46] which failed to reach 100% when utilising a dataset with a greater number of classes. The Naïve-Bayes classifier [49] lost accuracy greater than 99% at the highest SNR despite using similar features to the SVM classification system [48] and including fewer classes in the dataset, suggesting that SVM is the superior classifier for this problem. The system which employed the ANN [50] used low-order moment features, it lost perfect classification accuracy at 17dB SNR but the reduction in accuracy as the SNR decreased was far less severe than seen across other systems. This may be due either to the classifier model or the choice of feature. Little information can be gained from the performance of the RF model as the employed dataset is vastly different from other systems in the comparison. What can be said is that for this multivariate classification problem the authors found that the more complex RF model provided a significant increase in performance over a DT.

The information which can be gained from this comparison is that RF can provide in-

creased accuracy when working with high dimensional feature space, SVM is a superior classifier to Naïve-Bayes when employing high-order cumulant features, and that either low-order moments as features compared to high-order cumulants or an ANN over an SVM may provide increased robustness to noise with similar peak accuracy.

The subtractive clustering algorithm [51] as a means of feature extraction was demonstrated to be superior in every performance metric to using DBSCAN [37], and indeed all other feature-based methods. Even when utilising a dataset with more classes than the DBSCAN system, 100% accuracy was maintained to an SNR which was 7dB lower and nearly matched the SNR at which other feature-based methods lost perfect accuracy even when including 128QAM and 256QAM in the dataset. Critically, this method was shown to be capable of maintaining 100% accuracy as low as 6dB SNR if 32QAM and 128QAM were discounted from the utilised dataset, exhibiting vastly superior noise robustness in comparison to all other feature-based classifiers. The only limitation of this system was found to be the lack of inter-order classification ability, although no other system bar [46] was demonstrated to have this capacity. Clustering-based feature extraction may therefore be judged to the superior method of performing feature-based AMC, in particular the subtractive clustering algorithm [51] was found to be the optimal method of executing this task.

3.3 DL AMC Methods

Having explored the landscape of feature-based AMC, this review now focuses attention on the impact of DL-based approaches. The primary advantage of leveraging DL is that this class of system can learn hierarchical representations directly from an unprocessed input (although in some cases minor preprocessing is utilised) [54]. This data-driven approach allows the network to learn features which are optimally suited for AMC and which may not be expressible with closed-form mathematical solutions [15, 16]. This enables classification performance which often surpasses what has been achieved by traditional feature-based methods. However, DL systems require large and highly complex structures to obtain strong performance, with the achieved performance often scaling with the model size and complexity [54]. This makes this class of technology difficult to optimise for systems which are constrained by available hardware and power.

This section will first discuss the variety of model structures which have been employed for AMC, including the Convolutional Neural Network (CNN), the Long Short-Term Memory (LSTM), and the Transformer. Following this overview, the classification performance which has been achieved by utilising each model structure will be compared amongst each other as well as with the strongest feature-based approaches.

3.3.1 The CNN Model Structure

Perhaps the most popular architecture for AMC purposes is the CNN which is designed to process data with a grid-like topology. This class of model is generally utilised for the processing of images as this data format suits the matrix input format of the CNN [54], as such these models see frequent usage with a constellation diagram input [57, 58], but some researchers opt to use a time-series input [16]. CNNs leverage convolution, which involves sliding a filter (or kernel) over the input data, performing element wise multiplications

and summing the results to produce a feature map. This process is repeated with multiple filters which each learn to detect different features from the input. CNNs are commonly used with multiple convolutional layers with each applying their own set of convolutional filters to learn further features [54]. MaxPool is another fundamental component of the CNN structure. It down-samples the feature maps produced by the convolutional layers by utilising a sliding window over the obtained features and selecting the maximum value within each window. This reduces the computational cost by reducing the feature dimensions and mitigates overfitting by focusing only on the most important features [59]. Figure 3.6 displays the general structure of a CNN classifier.

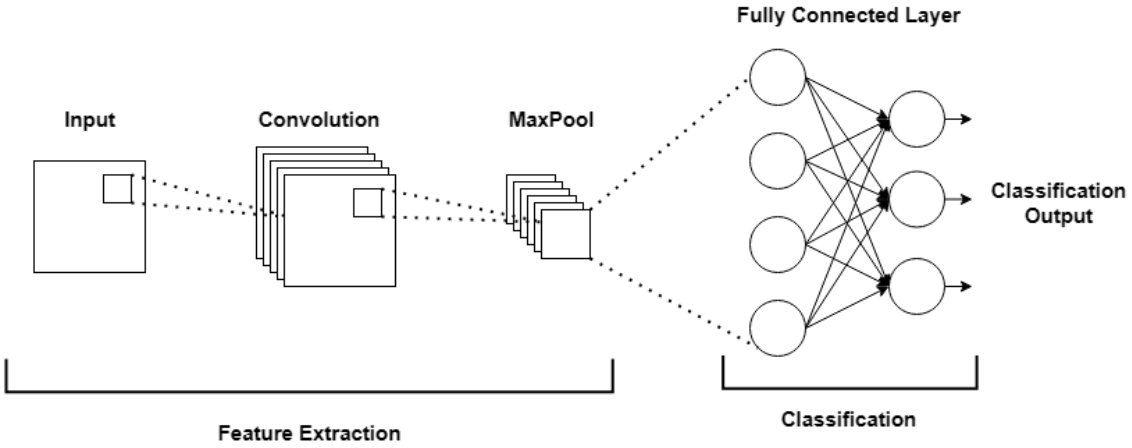


Figure 3.6: *The General Structure of a CNN Classifier, Reproduced from [119]*

Increasing the number of CNN layers generally increases the classification performance as it enables a greater number of features to be learned [54]. However, the relationship between complexity and performance does not scale linearly, beyond a certain level of complexity the increase in performance may begin to suffer diminishing returns, a decline in accuracy may even be observed [54]. This is due to the problems such as vanishing/exploding gradients or overfitting. Vanishing or exploding gradients are caused by the gradient of the loss function with respect to the network's weights becoming incredibly small or large, this makes the early weights in the network either incredibly difficult to update due to the small gradients or become unstable due to large gradients [54]. Techniques such as residual connections [60] have been utilised by AMC researchers to mitigate these gradient issues by allowing gradients to flow more easily within the network [16]. Batch normalisation (BatchNorm) is another tool which is frequently used to both mitigate gradient issues and provide slight regularisation, BatchNorm normalises the activation functions following each iteration (or mini-batch), ensuring that gradient size remains stable [61]. Overfitting is generally mitigated by L1/L2 regularisation or dropout [62], a technique which randomly sets network weights to 0 with each training iteration, encouraging the network to learn the feature set with sets of mini-networks and not overly rely on particular neurons [54]. The application of these techniques allows for the creation of deep networks which can learn a greater number of features from the input dataset and therefore obtain stronger classification accuracy.

Within the context of efficient hardware implementation, a larger network may not always

be advantageous. To reduce network implementation size techniques such as quantisation and pruning may be employed. Quantisation reduces the precision of the weights, activation functions, and inputs. Generally neural networks are trained with 32-bit floating-point precision, but some authors opt to employ 16-bit floating-points or use fixed-points with even lower precision. It has been found that 6-bit fixed-point precision can be used with minor losses in terms of classification accuracy [18, 39], ternary weights or 2-bit precision has also been proposed but significant reductions in classification accuracies have been found to occur with this strategy [17]. Pruning can also offer a means of reducing model implementation size, this technique removes neurons with weight values which are close to 0 as they contribute little to the classification network. Pruning has been found to be effective at minimising implementation size with a minimal loss in performance as the pruned neurons already provided little benefit [39].

3.3.2 The LSTM Model Structure

The LSTM is a development to the RNN model structure, as a class of RNN it is therefore designed specifically for a time-series input, making them particularly applicable for classification of a time-series I/Q data input [54]. The building block of the LSTM is the cell, a diagram of a cell can be found in Figure 3.7.

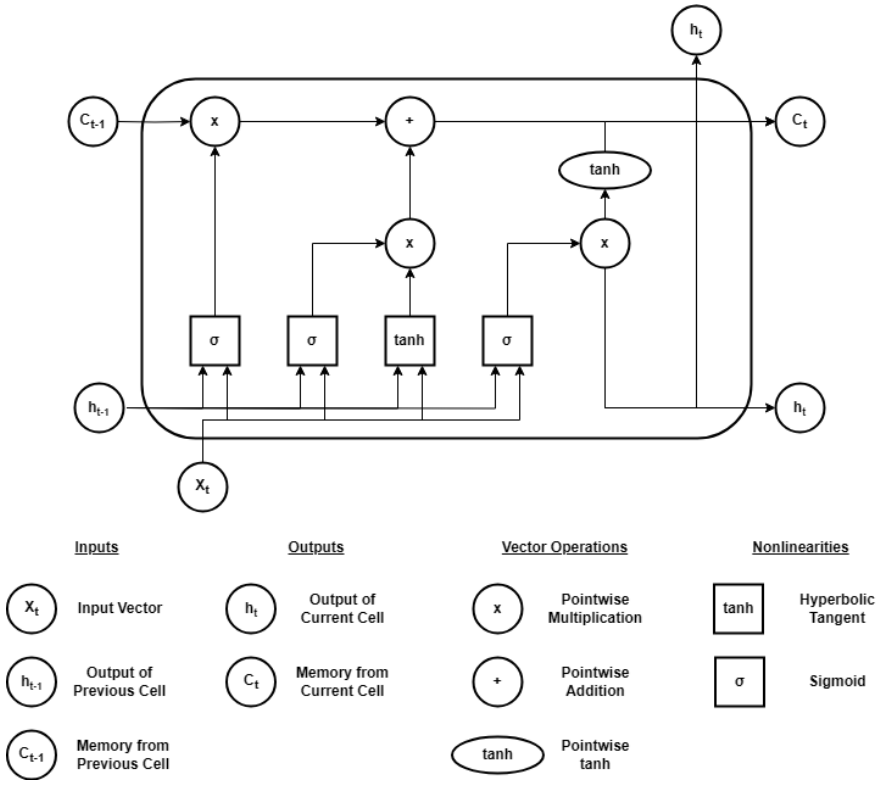


Figure 3.7: *The General Structure of an LSTM Cell, Reproduced from [120]*

The cell provides memory to the LSTM structure, allowing information to be conserved across time steps [54]. The cell state C_t is the component which enables memory, it acts

a conveyor belt between cells, carrying information across time steps. There are 3 gates within a cell which each perform a particular function. The forget gate is comprised of the leftmost sigmoid activation function, it decides information to discard by inspecting the input hidden state h_{t-1} and current input X_t, it outputs a 0 or 1 to signal if information should be forgotten or kept. The input is formed from the middle sigmoid and hyperbolic tangent, it decides which new information should be learned. The final gate is the output gate, formed from the rightmost sigmoid, it decides which information should be passed to the next cell [54]. Each LSTM cell processes a particular time step in a time-series sequence of data, as such they are arranged into layers of a length equivalent to the number of time steps in the input data sequence. Just as with the CNN, additional layers can facilitate the network to learn hierarchical representations of the input sequence. Early layers tend to learn features which capture short-term dependencies, whereas later layers learn more abstract and long-term relationships. There are also challenges when scaling the network size such as vanishing/exploding gradients and overfitting [54]. Once again techniques such as BatchNorm, dropout, and residual connections may be used to mitigate these issues, although more care must be taken when implementing residual connects as the temporal relationships between cells must be preserved [54]. Figure 3.8 shows a typical 3-layer LSTM classification structure.

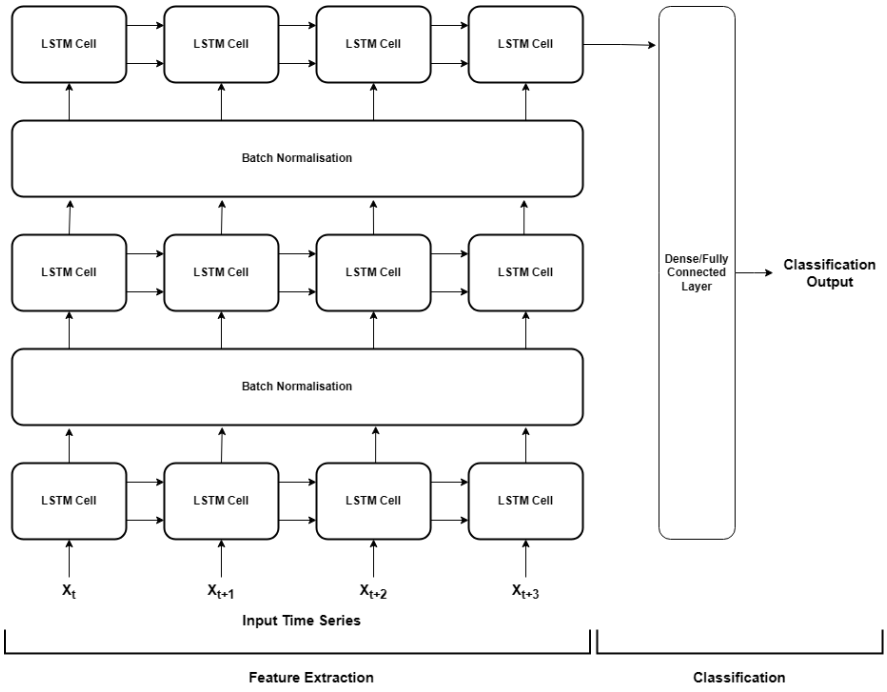


Figure 3.8: *The General Structure of an LSTM Classifier, Reproduced from [118]*

3.3.3 The Transformer Model Structure

The Transformer is a model which has risen to prominence in recent years, the core feature extraction mechanism of the Transformer is Attention, or more precisely Self-Attention [63]. Similarly to the LSTM, the Transformer is designed to operate on sequences of data. Com-

monly sequences of text are used such as with the large language models ChatGPT [33] and Gemini [64], but recently the ability of Transformers to classify sequences of numerical data has been investigated [65]. The traditional Transformer model employs an encoder-decoder structure but for time-series classification tasks the encoder is generally only used [66, 67]. To begin, the dataset must first be normalised and arranged into fixed length inputs. Next, positional encodings are added to the input data to provide information about the relative positioning of each number in the sequence. Following this, the Attention mechanism begins. Multi-Head Self-Attention uses matrix multiplication to attend each time step in the sequence to every other time step [63, 66]. The mechanism of Self-Attention involves operation on 3 learned matrices:

- **Query (Q)**: Represents the current time step i.
- **Key (K)**: Represents all time steps j.
- **Value (V)**: Represents the information from all time steps j.

The Attention weight between time step i and all time steps j is given by Equation 3.1:

$$\text{Attention}(Q_i, K_j, V_j) = \text{softmax}\left(\frac{Q_i \cdot K_j}{\sqrt{d_k}}\right) \cdot V_j \tag{3.1}$$

Where $\cdot$ is the dot product, and d_k is the dimensionality of K_j. Multi-Head refers to the Transformer performing this process multiple times per layer, each head can attend to different aspects of the relationship between elements. For example, one head may learn long-term relationships, another short-term, and a final head may learn to identify a specific pattern. The outputs of each head are concatenated and applied to a feed-forward neural network which identifies which learned features contribute most strongly to the output [63].

The combination of Multi-Head Self-Attention and a feed-forward network is defined as a Transformer layer [63]. Just as seen with the CNN and LSTM, additional layers allow for the learning of hierarchical features which can increase performance but introduces the typical overfitting and vanishing gradient problems. Again, residual connections, dropout, and normalisation are employed to mitigate these issues [54, 63].

At the output of an N-layer chain of Transformer units there is the typical linear layer followed by a SoftMax activation layer to provide classification functionality. Combining all the discussed components results in the network structure shown in Figure 3.9.

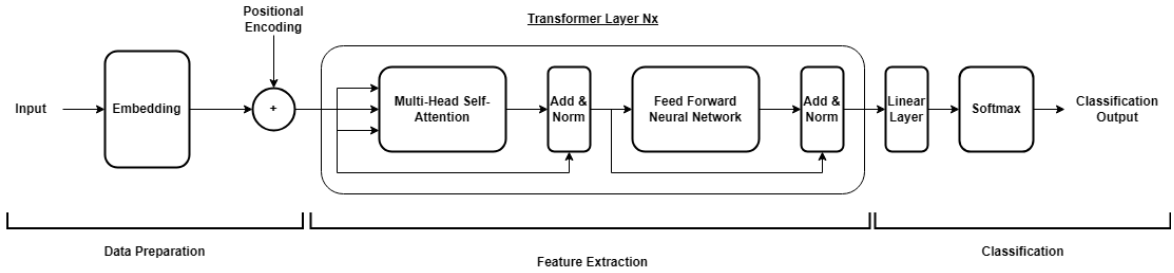


Figure 3.9: *The General Structure of a Transformer Encoder, Diagram Based Upon [63]*

Transformers process sequences of data much like the LSTM but have been found to have various advantages. Firstly, the Attention mechanism takes advantage of parallel processing, allowing Transformers to process all parts of the input sequence at once, speeding up training times significantly, particularly on hardware which performs matrix multiplications efficiently such as GPUs [63]. Additionally, the Attention mechanism allows for the model to directly attend any part of a sequence to any other which allows for the capturing of long-range dependencies, unlike the LSTM which struggles in this regard [63]. However, it has been shown that LSTMs may be better suited when the size of the available dataset is small, Transformers which reach state-of-the-art performance typically do so when trained on extremely large datasets [67].

3.3.4 Combinations of Model Structures

Each model structure which has been discussed has a particular data format which it is designed to operate upon. The CNN is designed to accept data in a matrix format such as images, whereas the LSTM and Transformer accept sequences [54]. Furthermore, as the feature extraction mechanism is different between models, employing various extraction mechanisms may allow for a greater depth of features to be learned, therefore increasing performance [19].

Some works opt to utilise combinations of CNNs, LSTMs, and Attention [19, 68, 69]. This is in an attempt first provide the classification system with data in multiple formats which leads to a greater depth of information to learn features from, but to also exploit the differing feature extraction mechanisms. This can be thought of as similar to utilising different algorithmic features in feature-based classifiers.

However, DL systems already suffer from complex implementations [54]. Including multiple DL models within one design naturally imposes a further increase in implementation complexity.

3.3.5 Input Data Formats

When employing DL networks for AMC it is advantageous to provide the classifier with as much data as possible and allow it to learn its own features. It is therefore generally the case that raw I/Q waveform data is used as an input. I/Q data is arranged into blocks of samples of lengths determined by the author, sample lengths may be as low as 128 [15] or as high as 2048 [65]. This input format has been utilised for all model structures, including combinations of each structure.

Constellation diagram inputs have also been used as the input to CNN classifiers. Constellation diagram data is formatted into images, some authors use only images of the pure constellation diagram [57, 70], whereas others opt to preprocess the data and apply information about density to the constellation diagram image with colour [58, 71]. The aim of adding colour to the diagram is to provide further information to the classifier with which to learn from.

Finally, one example uses both I/Q waveforms and constellation diagram images with density gradient colours with the aim of providing as much information as possible [72].

3.3.6 DL Performance Comparisons

The variety of model structures, input data formats, and dataset configurations has now been discussed. This section aims to provide a review of the level of performance achieved by each strategy within the context of the dataset used by each work. As discussed in Section 3.1, the achieved system performance is in part due to the set of modulation schemes employed for training and testing. In general, favourable judgements will be given to AMC systems which demonstrate the ability classify high-order signals, modulation schemes of equivalent order, and modulation schemes of the same format. Table 3.2 summarises this information for ease of comparison.

Table 3.2: *Comparison of AMC Model Architectures and Input Data Formats and Included Modulation Schemes*

Author	Model	Max Order	No. of MS	Inter-Order	Inter-Format	Input Type
Hermawan [73]	IC-AMCNet (CNN)	64	10	Yes	Yes	I/Q
Rajendran [15]	LSTM	64	11	Yes	Yes	I/Q
Ke [74]	DAE-LSTM	64	11	Yes	Yes	I/Q
Hamidi-Rad [65]	Transformer	64	10	Yes	Yes	I/Q
Liu [75]	CLDNN (Densenet CNN & LSTM)	64	11	Yes	Yes	I/Q
Liu [75]	DenseNet (CNN)	64	11	Yes	Yes	I/Q
O'Shea [16]	ResNet (CNN)	64	11	Yes	Yes	I/Q
Zhang [69]	LSTM & CNN	64	11	Yes	Yes	I/Q
Alarabi [50]	ANN	64	4	No	Yes	I/Q - Features
O'Shea [16]	ResNet-256	256	24	Yes	Yes	I/Q
Kumar [19]	CNN & LSTM & Attention	1024	9	No	Yes	I/Q
Doan [57]	FiF-Net (CNN)	64	8	Yes	Yes	CD
Kumar [70]	ModNet	64	6	No	Yes	CD
Huang [58]	M-CNN (CNN)	256	7	No	Yes	CD & Density
Kumar [71]	Inception ResNet V2 (CNN)	64	8	Yes	Yes	CD & Density
Wang [72]	DrCNN (CNN)	64	8	No	Yes	I/Q & CD & Density
Wang [51]	DT	256	6	No	Yes	Cluster No.

For clarity, comparison graphs will be split between methods which use high-order modulation scheme data inputs and those which do not. The best performing low-order input system will be included in the high-order input figure for comparison, the strongest feature-based classifier is also included in both figures for the same reason. Figure 3.10 shows the average classification accuracy against SNR achieved by each DL system which accepts data with a maximum order of 64 as an input, Figure 3.11 displays the same chart but magnified to better display high SNR performance.

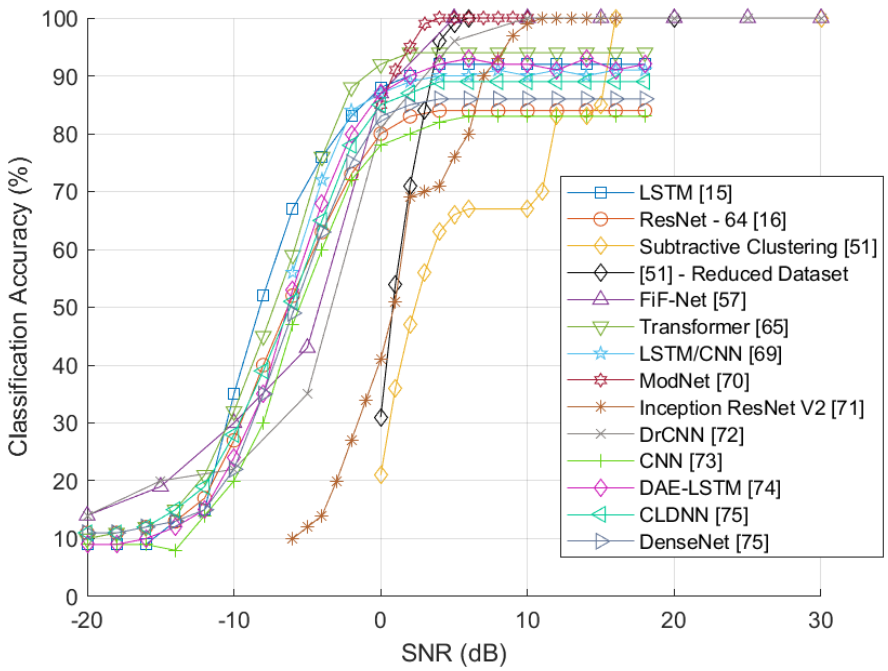


Figure 3.10: *Average Classification Accuracy (%) Against SNR (dB) for Various DL Models Tested with a Maximum Modulation Order of 64*

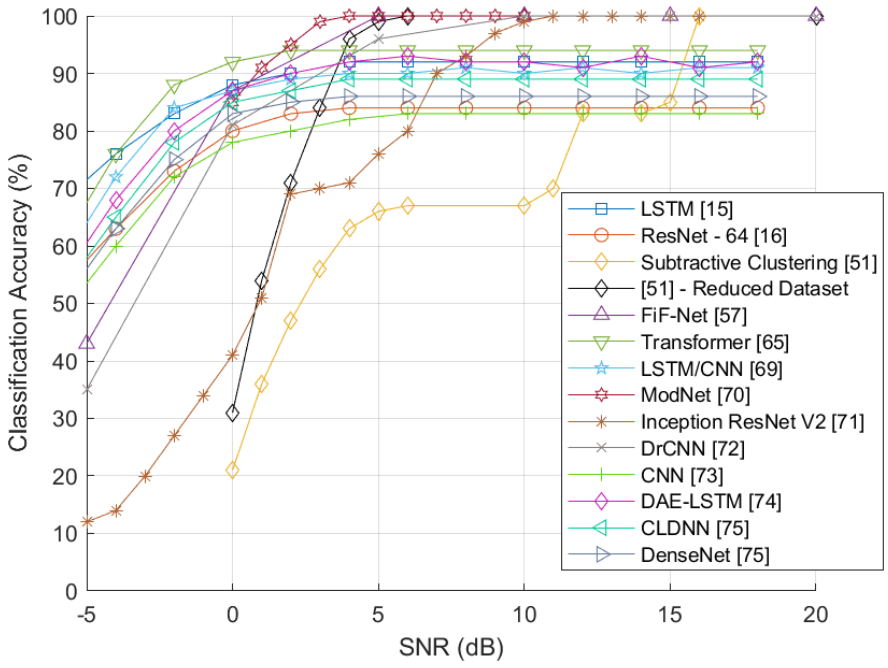


Figure 3.11: *Figure 3.10, Magnified to Only Display SNRs Between -5dB and 20dB*

Immediately when inspecting Figure 3.10 and 3.11 it can be seen that the performance curves

seem to follow 2 broad trends. The classifiers which accept a constellation diagram input all reach 100% accuracy, those which accept only an I/Q waveform fail to reach this performance level as they all reach a peak accuracy between 94% and 83%. However, although the I/Q classifiers fail to reach 100%, the peak performance differential is less severe than it appears. All I/Q accepting models shown in the comparison use the RadioML.2016.10A [41] dataset which includes numerous analogue signals, including WBFM and Amplitude Modulation - Double Side Band - Suppressed Carrier (AM-DSB-SC). It was said in Section 3.1 that every model which uses this dataset cannot distinguish between these 2 analogue modulation schemes. As this thesis is solely concerned with digital modulation, the performance on analogue modulation schemes may be ignored, the obtained peak accuracy by the LSTM would therefore be 96.2% and could potentially be higher if the models were retrained to classify only the set of digital signals. Despite this improvement, the peak accuracy of the I/Q accepting models would not reach 100% as these models all exhibit some degree of inter-format misclassification at high SNRs.

Beginning with the performance characteristics of the I/Q accepting models, regardless of the level of peak accuracy which is achieved, every model follows a similar trend of performance degradation. The trend is as follows, between 5dB and 0dB SNR the accuracy is slightly reduced but remains above 75% in all cases, between 0dB and -3dB SNR the gradient of accuracy degradation severely increases, between -3dB and -10dB SNR each system sees a linear decline in accuracy until at -12dB SNR the accuracy reaches levels close to that of a random guess, at -15dB SNR all but 2 models classify with equivalent rates to a random guess. Contrary to the level of peak accuracy achieved, models which accept an I/Q waveform input show stronger robustness to noise than those which accept constellation diagrams. Additionally, the performance differential between I/Q accepting models is in general maintained across the majority of SNR values, for example the Transformer [65] and LSTM [15] both achieve the highest peak accuracy and maintain the highest accuracy with respect to other I/Q-based models across all SNRs greater than -10dB. Conversely, the CNN [73] achieves the lowest peak accuracy of any model and maintains this performance differential at all SNRs. Admittedly, this trend is not followed in all cases as there is some degree of crossing, particularly by models with middling performance, but in general the stated trend is followed. Each I/Q accepting model seems to produce the same sigmoid performance trend, but each is shifted upwards by a constant, therefore evaluating the peak accuracy of the I/Q accepting models can provide a strong description of the overall performance.

When evaluating the peak accuracy achieved by the I/Q accepting models another clear trend can be seen. The model structures which are designed primarily to learn from sequences of data perform better than those which are not. For example, the model with the lowest peak accuracy is the traditional CNN [73], ResNet [16] which introduces residual connections to the CNN enables an accuracy gain of 1%, DenseNet [75] which uses a densely connected CNN provides a 2% performance gain over ResNet. The best performing I/Q accepting CNNs both incorporate LSTM layers into their model structure, CLDNN [75] obtains a 3% performance gain over DenseNet and the LSTM/CNN [69] achieves a 2% performance gain over CLDNN. The best performing models include no CNN functionality, the standard 2-layer LSTM [15] and the LSTM in an autoencoder configuration [74] both achieve an average of 92% peak accuracy, with the accuracy of the autoencoder being more variable than that of the LSTM, this amounts to a 1% increase in accuracy being obtained by not

utilising convolutional layers. The best performing I/Q accepting model is the Transformer [65] which achieves a peak accuracy of 94%, 2% higher than both LSTMs. The analogue modulation scheme AM-SSB was not included in the Transformers dataset despite the authors using the RadioML2016.10A dataset [41]. Both LSTMs performed poorly on this scheme, at high SNRs it was classified correctly with an accuracy of only 94% and 96% with the LSTM and autoencoder respectively, there was also a significant degree of misclassification of other schemes as AM-SSB. The removal of this particular modulation scheme from the Transformer dataset would have to have been performed manually and the authors do not provide reasoning as to why it was done. Using the confusion matrices provided for the LSTM and autoencoder the peak accuracy was recalculated assuming that AM-SSB was also discounted, the obtained peak accuracies were found to be 94% and 93.4% respectively, bringing them to equivalency with the transformer.

As all I/Q accepting models broadly use the same dataset, definitive conclusions can be drawn. It is clear that the performance of the system is primarily dependent upon the model structure. Furthermore, model structures which are designed to process input sequences outperform those that are not. Despite the enhancements made to the CNN structure such as residual and dense connections, the performance only approached that of the LSTM [15] and Transformer [65] when elements from these model structures were incorporated. The performance differential between the LSTM and Transformer is minor, although the LSTM in an autoencoder configuration [74] provided little benefit and seemed to make the accuracy more variable.

Evaluating the performance of the constellation diagram accepting models is more difficult as each system was trained and tested with a different dataset. Every system in the comparison employs a CNN for its image recognition abilities, unlike the I/Q accepting CNNs this is the input format which the CNN was designed to accept [54]. The applicability of the CNN to this task is made clear as these models all reach 100% accuracy at higher SNRs, however they do suffer from reduced robustness to noise in comparison to the I/Q accepting models.

The system which maintains 100% accuracy to the lowest SNR is ModNet [70]. This model was trained with a dataset of 6 different QAM and PSK modulation schemes, demonstrating inter-format classification, no inter-order classification ability was demonstrated. A discrepancy was noticed in the provided images in Figure 3 of [70]. This discrepancy was that the provided constellation diagram images do not seem to represent the stated SNR levels to which the authors claim they do. To illustrate, 3 diagrams of 64QAM from [70], [57], and the work conducted in this thesis are shown in Figure 3.12.

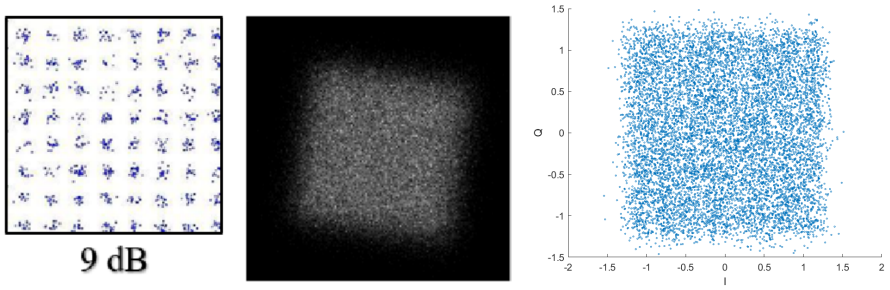


Figure 3.12: *64QAM at an SNR of Approximately 10dB from: (a) [70], (b) [57], (c) This Thesis*

Figure 3.12 (a) shows the reported 64QAM constellation diagram at 9dB SNR from [70], (b) is 64QAM at 10dB SNR from [57], and (c) is 64QAM at 10dB SNR from the dataset used in the work conducted in this thesis. It is clear that (b) and (c) have a similar appearance, in each case the clearly defined constellation point structure of 64QAM is lost and instead a noisy square of data points may be seen. Conversely, in (a) the constellation points are still clearly defined even despite the stated SNR being 1dB lower. This same trend is seen across all the provided constellation diagram images in [70] casting doubt upon the validity of the reported results. It may be the case that the error was introduced during the writing process, but it is impossible to know, therefore the results obtained from [70] will be discounted.

Doan et al. [57] also use constellation images to achieve AMC with their proposed FiF-Net system. In this case no dataset discrepancy could be found, although it was thought that the employed set of modulation schemes may have been utilised to provide vastly different constellation diagram appearances at lower SNRs and thus the system could perhaps more be trained on the appearance of shapes rather than constellation diagrams. For example, 64QAM results in a blurry square as shown in Figure 3.12 (b), 64PSK a blurry circle, and notably 16PAM was included in the dataset which is arranged in a line format. However, the authors do include 2 modulation orders per modulation format and the system distinguishes schemes of the same format with a high degree of accuracy, demonstrating strong inter-format classification abilities. Therefore, the model performs well overall, 100% accuracy is achieved, and this accuracy is maintained to 5dB, the lowest SNR of any constellation diagram classifier which could be found in the literature.

The remaining 2 constellation diagram classifiers pre-process the images to mark regions with colours based upon the density. Kumar et al. [71] propose the usage of 2 CNN model structures, InceptionNet V2 [76] and ResNet-50 [60]. Of the 2 structures ResNet-50 performs the strongest across all SNRs greater than 0dB. Despite the system being demonstrated to be capable of achieving 100% accuracy, this performance level is lost at an SNR 10dB and suffers from rapid performance deterioration. Accuracy equivalent to a random guess is reached at -6dB SNR. Therefore, peak accuracy is lost and equivalency to a random guess is reached at the highest SNR of any system in the comparison.

DrCNN proposed by Wang et al. [72] uses 2 CNNs in series, the first uses I/Q data to differentiate between orders of PSK, FSK, and PAM, whereas the second uses constellation diagrams with dense regions marked in yellow to distinguish 16QAM and 64QAM. While this strategy achieves 100% accuracy, this level of performance is lost below an SNR of 10dB,

however in this case the accuracy degradation is far less severe than seen with ResNet-50 [60], with the trend of degradation being more similar to that of FiF-Net. Despite reaching 100% accuracy, the usage of 2 CNNs discounts the usage of this method for efficient hardware implementation, as a single CNN already leads to a large utilisation.

To summarise constellation input DL systems, they are demonstrated to be capable of achieving 100% accuracy, albeit when using fewer classes than seen with the I/Q systems. Yet the I/Q systems all struggled to distinguish between 16QAM and 64QAM even at high SNRs, a task which the constellation classifiers performed well on. The strategy of marking dense regions on the constellation diagrams was not shown to provide any improvements in performance as FiF-Net which did not require this pre-processing step was shown to maintain perfect accuracy to a lower SNR than the 2 systems which incorporated density into the images.

3.3.7 High-Order DL Classifier Performance

Following the discussion of the systems trained with a maximum modulation order of 64, the systems which employed a dataset with higher-order modulation schemes may be explored. There is a limited number of systems which opt to provide results on signals of orders greater than 64, the results provided in Figure 3.13 are the only examples of works which do so, the strongest feature-based classifier is also included for comparison.

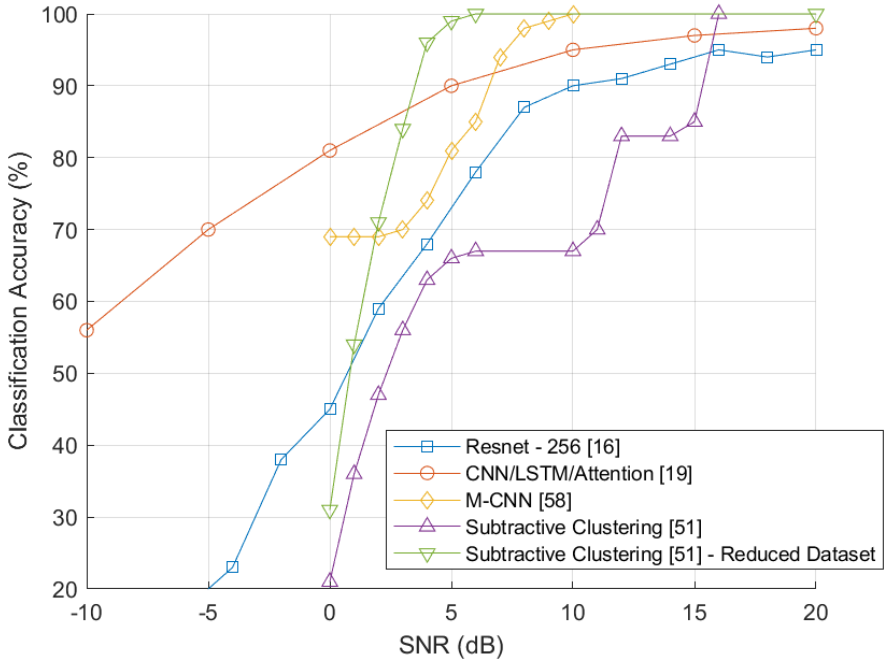


Figure 3.13: *Average Classification Accuracy (%) Against SNR (dB) for Various AMC Models Tested with a Dataset Including High-Order Modulation Schemes*

There are 3 DL models to compare, 2 are I/Q accepting [16, 19] and 1 accepts constellation diagrams with density marked by colour [58]. M-CNN proposed by Huang et al. [58] is the only system of the 3 to achieve 100% classification accuracy, which is consistent with the previous

discussion where the constellation diagram accepting models were shown to be capable of reaching this level of performance. However, once perfect classification accuracy is lost the degradation in performance is drastic, again this matches what was observed in Figure 3.10 but in this case the rate of decrease is even more pronounced. This may be explained by the fact that this system is trained solely on various orders of QAM from 4 to 256, once the SNR is low enough to eliminate any separation between constellation points, the appearance of each constellation diagram becomes 1 cluster of points in a square formation. This is contrary to the dataset used by FiF-Net where wholly different modulation formats could be distinguished due to the overarching structure of the modulation format, e.g. a line for 16PAM, a circle for 64ASPK, a square for 64QAM. Therefore, despite the severe performance decrease, reaching 100% accuracy implies strong inter-format classification capabilities, although no inter-order classification ability is demonstrated.

The ResNet system proposed by O'Shea et al. [16] achieves a higher peak accuracy than any other I/Q accepting DL system in the previous lower-order comparison, although this peak is reached at 12dB SNR and not 5dB as was the case for all systems compared in Figure 3.10. This performance level is reached using a dataset which is extensive. The dataset is the RadioML.2018.01A opensource dataset [41], it includes 24 modulation schemes of formats such as QAM, APSK, and PSK, as well as every modulation scheme included in the previously discussed RadioML.2016.10A dataset [41]. Both datasets are published by O'Shea et al. The improved peak accuracy despite the more difficult dataset could be explained by the number of examples included in each dataset. Figure 16 in [16] shows how the performance of the system improves depending upon the training set size, the results shown in Figure 3.13 are taken from the accuracy obtained when 2 million examples were used for training, RadioML.2016.10A only provides 220,000 examples. Inspection of Figure 16 in [16] shows that if an equivalent number of examples were used to train the high-order ResNet classifier then a peak accuracy of 87% would be reached at 14dB SNR. This also implies that the I/Q classifier results shown in Figure 3.10 could also be improved by using an expanded number of training examples. The RadioML.2018.01A dataset includes 10 analogue signals which are not of a concern to the work conducted in this thesis, fortunately [16] provides figures which demonstrate the accuracy achieved across each modulation scheme. Recalculating the peak average accuracy from only PSK, APSK, and QAM individual accuracies provides a value of 95.8%, which is 0.8% higher than the peak accuracy the authors state was achieved in the text.

When compared to the results achieved by M-CNN, there is no SNR at which the ResNet classifier achieves a higher average accuracy, although the accuracy of M-CNN below 0dB is not provided. Despite the reduction in performance in comparison to M-CNN, the ResNet classifier is shown to have the capability to classify a wider range of modulation schemes, as inter-order and inter-format classification is demonstrated across a range of high-order schemes. The ResNet system also achieves these results without the requirement for a significant pre-processing step, whereas M-CNN requires constellation diagram densities to be calculated and subsequently images to be created which include this information, necessitating an increase in latency and hardware requirements. In the previous comparison between low-order classifiers it was found that LSTMs and Transformers achieve a higher peak accuracy than CNN-based classifiers, no examples of pure LSTMs or Transformers being tested on high-order datasets were found in the literature. If the identified trend continues when work-

ing with high-order datasets there is scope for a pure LSTM or Transformer model designed to accept I/Q data to achieve a greater level of performance than seen with ResNet.

The final high-order classifier to discuss is the CNN/LSTM with attention, proposed by Kumar et al. [19]. This model employs layers from each DL structure which has been discussed throughout this literature review, providing the opportunity to examine if LSTM and Transformer layers can provide increased performance over the pure CNN on high-order signals. The employed dataset consists of QAM signals from 16 to 1024, PSK of orders 2 to 8, and a frequency modulated scheme MSK which will be ignored. This is the only system to be tested with 512QAM and 1024QAM which makes for a less direct comparison with ResNet but does make this the only system to demonstrate classification performance with modulation schemes of orders this high.

As suggested by the trend identified from Figure 3.10, the addition of LSTM and Transformer layers to the CNN results in an increase in peak accuracy. At an SNR of 20dB a peak accuracy of 98% is obtained, a 3% increase over what was achieved by ResNet. However, the accuracy is still 2% lower than the 100% achieved by M-CNN, although this difference could potentially be explained by the inclusion of 512QAM and 1024QAM. Similarly to the previous comparison, the rate of accuracy loss as the SNR decreases is lower than that of the 2 CNN-based systems, interestingly this rate is far lower than seen with I/Q accepting models in Figure 3.10. At -10dB all models classified with an accuracy below 30%, whereas the model in question remains at 55% accuracy at this SNR. It is unknown what provides such strong robustness to noise in this case, a likely explanation is that the system is particularly effective when classifying the low-order PSK and MSK signals included in the test dataset. Figure 2 (a) in [19] provides results for the system when the attention layer is not employed, it is shown that the inclusion of the attention layer provides a 3% increase in peak accuracy.

It is difficult to draw definitive conclusions based on the results shown in Figure 3.13 due to the differing datasets. What can be concluded is that trends identified in the low-order discussion hold when high-order signals are employed. There are no examples of I/Q accepting classifiers which obtain 100% average accuracy. While constellation diagram accepting classifiers are shown to be capable of reaching perfect classification, they suffer from reduced robustness to the effects of noise compared to I/Q accepting models. The performance obtained by I/Q accepting models was again found to be dependent upon the feature extraction technology employed as the CNN was found to be less apt in comparison to system featuring LSTM and Transformer layers.

3.3.8 Comparisons between Feature-Based and DL Methods

Section 3.2.5 concluded that the best performing feature-based classifier was the subtractive clustering-based model proposed by Wang et al. [51]. Included in Figure 3.13 were curves displaying the results of this technique on the full dataset as well as the results with the 32QAM and 128QAM modulation schemes removed.

When comparing with the classifier results shown in Figure 3.5 it must be taken into account that the clustering classifier includes both 128QAM and 256QAM in the test dataset, which no other models in the comparison includes. Without discounting any signals, the system in question achieves 100% classification accuracy which is not achieved by any I/Q accepting DL system. This level of peak performance is achieved by the constellation diagram accepting models, however the SNR at which the clustering classifier loses peak accuracy is

10dB higher than what was achieved by the strongest DL classifiers.

Inspection of the results with 32QAM and 128QAM results removed shows that perfect accuracy is maintained to as low an SNR as the strongest DL image classifier. The SNR at which peak performance is lost is also similar to the I/Q accepting models. The major limitation of this system is that the robustness to noise is weaker than every DL system bar [71].

Comparisons to the results shown in Figure 3.13 are perhaps more apt due to the similarities between the datasets. Without removing any results, the system achieves 100% accuracy which both I/Q accepting models fail to reach, the SNR at which peak accuracy is lost is also similar to both [16] and [19]. M-CNN maintained 100% accuracy to a lower SNR than [51] on the full dataset.

The results when the 2 QAM orders are removed show that 100% accuracy is maintained to an SNR which is 5dB lower than what is achieved by M-CNN. At all SNRs greater than 3dB, the clustering-based classifier achieves a higher classification accuracy than both I/Q accepting DL systems.

This comparison shows that without careful selection of employed modulation schemes, clustering-based classifiers can be at least somewhat competitive in terms of peak accuracy with the state-of-the-art DL models. With careful selection of the set of utilised modulation schemes, a feature-based classifier may be created which can achieve perfect classification accuracy, and maintain this level of performance to as low an SNR as has been achieved by the state-of-the-art.

The goal of this thesis is to create a system which can match or improve upon the results obtained by the state-of-the-art classifiers whilst also achieving significant reductions in terms of utilisation requirements. The results shown here demonstrate that the performance of the relatively complex and power inefficient DL models can be matched by feature-based models, thus creating a potential path towards a truly efficient AMC classifier. To provide a clearer picture of the landscape of hardware-implemented classifiers, the following section compares the results obtained by classifiers which have been implemented in hardware in terms of both accuracy against SNR and utilisation statistics.

3.4 AMC Hardware Comparison

The research conducted in the field of AMC is focused primarily on maximising the performance of software-based models, therefore there is only a limited number of works which provide statistics for hardware implementations. Table 3.3 shows the implementation statistics of works which provide them.

Table 3.3: *FPGA Resource Utilisation, Operating Frequency, Power Consumption, and Latency of Various AMC Implementations*

System	FFs	LUTs	DSP	RAM	F (MHz)	Power (W)	Time (µs)	Platform
Feature-Based DT [21]	16746	7933	180	14	100	-	15.79	XC7Z020-CLG484
RUNet [39]	21357	34563	0	40	-	-	7.5	ZCU111
QMCNet [39]	40476	61364	0	57	-	-	-	ZCU111
BaselineCNN [39]	54483	85151	0	70	-	-	-	ZCU111
TW-96 CNN [17]	369000	232000	1207	524	250	-	8	ZCU111
TW-BA-128 [17]	333000	234000	1408	523	250	-	8	ZCU111
TW-INCRA-128 [17]	324000	211000	1407	512.2	250	-	8	ZCU111
MobileNetV3 [18]	25800	31200	162	22	250	4.2	-	ASIC
Histo-SVM [20]	462	-	-	6144	-	-	20	Altera Cyclone II

There is not a large variety of model structures shown in Table 3.3, only 2 feature-based classifiers and 7 CNNs could be found in the literature. Furthermore, all CNN classifiers require an I/Q waveform input, which is unfortunate as the distinction between input formats may result in major differences in implementation size due to the difference in the required layer dimensionality. It is also unfortunate that no LSTM or Transformer implementation could be found as these models were identified as the best performing of any I/Q accepting system. Figure 3.14 shows the accuracy against SNR curves for each system in this hardware comparison. Table 3.4 displays information about the employed datasets used to obtain the results shown in Figure 3.14.

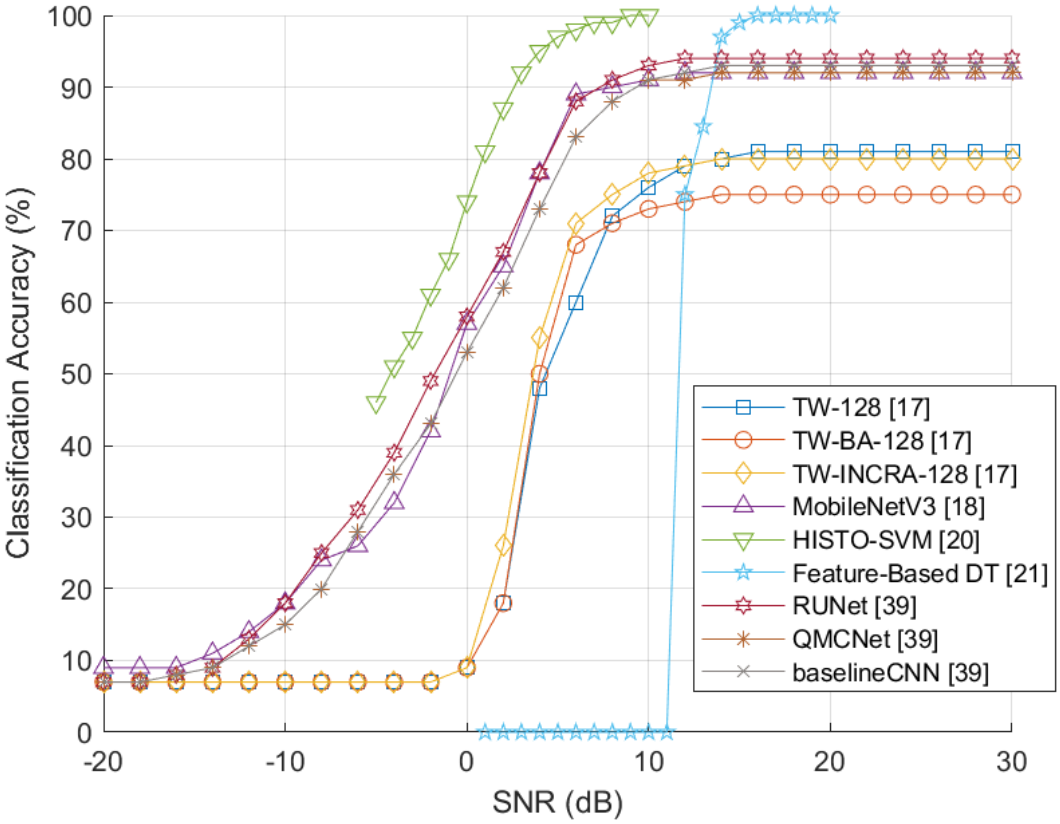


Figure 3.14: *Average Classification Accuracy (%) Against SNR (dB) for Various Hardware Implemented AMC Models*

Table 3.4: *Summary of Hardware Implemented AMC Model Datasets*

Model	Max Order	No. of MS	Inter-Order	Inter-Format	Input Type
Feature-based DT [21]	16	4	No	Yes	I/Q (Features)
RUNet [39]	256	24	Yes	Yes	I/Q
QMCNet [39]	256	24	Yes	Yes	I/Q
Baseline CNN [39]	256	24	Yes	Yes	I/Q
TW-96 CNN [17]	256	24	Yes	Yes	I/Q
TW-BA-128 [17]	256	24	Yes	Yes	I/Q
TW-INCRA-128 [17]	256	24	Yes	Yes	I/Q
MobileNetv3 [18]	256	24	Yes	Yes	I/Q
HISTO-SVM [20]	16	4	No	Yes	I/Q (Features)

3.4.1 Feature-Based Hardware Implementation Discussion

Beginning with the feature-based classifiers, Zhang et al. [21] propose a classifier which uses high-order cumulant features in conjunction with a DT. Cardoso et al. [20] use a linear SVM classifier in conjunction with a feature extraction mechanism which uses histograms to estimate the probability mass function of the input signal. 2 histograms estimate the amplitude and phase of a received sequence, they are then concatenated to create a signature which is then used as a feature for classification.

Both feature-based systems require fewer FPGA resources than the CNNs in the majority of cases. In terms of FFs, both the DT [21] and HISTO-SVM [20] require fewer FFs than every other system in the comparison. Similarly, the DT requires the fewest LUTs by 74.6%, HISTO-SVM either requires no LUTs or does not provide a value, it is unclear which is the case. The DT requires the fewest DSP slices of works which make use of this logical element, requiring 36.4% fewer than the next largest. Again, HISTO-SVM either requires either none or neglects to mention the utilisation of this element. The DT requires the least RAM of any system in the comparison, whereas HISTO-SVM requires the second most. The high RAM utilisation of HISTO-SVM is due to the histogram functionality primarily making use of RAM elements rather than FPGA logic. Each system requires fewer FPGA elements than every CNN implementation in 3 out of 4 categories.In general, this class of system may be implemented in hardware with a drastically reduced hardware cost. Neither system provide statistics for power consumption but based upon the utilisation it would be expected to be lower than that of the CNNs. Each system requires a greater time to execute than most of the CNN classifiers, with the feature-based DT being twice as slow as the quickest CNN, HISTO-SVM is nearly 3 times slower. Due to the drastically different datasets employed by each feature-based system in comparison to the CNNs, there is little to be gained from a classification accuracy comparison.

3.4.2 RUNet and QMCNet Hardware Implementation Discussion

Each CNN implementation uses various strategies to reduce the required hardware utilisation. RUNet and QMCNet [39] both employ quantisation and pruning to various degrees on the input, weights, and activation functions. MobileNetV3 [18] and the various CNNs proposed in [17] also employ quantisation but do not make use of pruning.

Kumar et al. [39] propose RUNet and QMCNet, they also provide utilisation statistics for an unpruned yet quantised baseline CNN. Each model has the input, activation function, and weights quantised to varying degrees, RUNet quantises each element to 6-bits, QMCNet to 4, 5, and 6-bits respectively, the baseline CNN quantises each element to 8-bits. Statistics for an unquantised RUNet are not provided, but when quantising QMCNet a reduction in terms of required bit operations and weight bits of 98.4% and 88.2% respectively was achieved. Similarly, with the baseline CNN a respective reduction of 93.9% and 75% was achieved, although in this case the values were quantised to 8-bits.

The authors describe QMCNet as a highly quantised implementation of the VGG10 [77] CNN structure, it has 6 convolutional layers. RUNet is a custom CNN featuring residual connections, it is described as a deeper network as it has 12 convolutional layers. Inspection of Table 2 in the text shows that RUNet requires 75% fewer operations and 93% fewer weight bits in comparison to QMCNet. It can also be seen from Table 3.3 in this thesis

that RUNet requires fewer FPGA resources than QMCNet in all utilisation categories. As RUNet is described by the authors to be a deeper network, is explicitly shown to feature double the number of convolutional layers, and is quantised to a lesser degree, it is initially counterintuitive that RUNet has such a dramatically smaller implementation size.

The answer to why this is the case may lie with the architecture of the convolutional layers themselves. As described in Section 3.3.1, CNNs learn features by performing convolutions via sliding N-dimensional kernels over the input, it was also said that per layer there may be numerous kernels employed which each capture different features. Each learned feature requires its own output channel from the layer which has its own associated weights. VGG10 which QMCNet is based upon uses 3x3 kernels. The number of kernels, and therefore the required output channels, scales from the input of the system to the output, early layers may feature 64 output channels, later layers may feature up to 512 output channels [54, 77]. Conversely RUNet has a maximum number of outputs channels of 48 and employs larger kernels with a maximum dimensionality of 36 [39]. The smaller number of output channels therefore requires fewer convolution operations as well as fewer weights. However, this strategy also means that fewer features are learned. The increased kernel size seems to be a strategy to maximise the learned information contained within the limited number of features. Larger kernel sizes do not necessarily result in better features being learned but do enable longer range dependencies between datapoints to be captured [54]. The system makes use of kernels of size 27, 9, and 1. The strategy seems to be to minimise the number of learned features but to maximise the information learned by each feature. The final point to make in this regard is that RUNet does not employ MaxPool layers after each convolutional layer, unlike QMCNet. MaxPool layers reduce feature dimensionality and therefore some information is lost, not including MaxPool layers appears to be a strategy to maintain the information which is learned by the limited set of features, with the added benefit of not requiring hardware to perform this operation. It must be said that this strategy is not explicitly outlined within the text, but it is hinted at. The strategy discussed here is therefore largely conjecture based upon the information which is provided.

With the unpruned implementation sizes explored, the benefits of pruning may be examined. For QMCNet the pruning process resulted in a respective reduction in bit operations and weight bits of 58.3% and 58.6%, 52.3% and 50% was achieved for RUNet. Therefore, at least a 50% reduction in the required operations and weight values was obtained in all cases. It may be the case that QMCNet was pruned to a larger degree as RUNet's operations and weights were found to be more necessary due to the reduced number of output channels.

Table 3.3 shows that despite being quantised to a greater degree, QMCNet has a greater utilisation than RUNet. Furthermore, while both systems require no DSP slices, RUNet requires the fewest FFs and DSP slices as well as the second fewest LUTs, and RAM of any CNN. The only other system which matches RUNet's aggregate hardware cost is MobileNetV3 [18] which requires fewer FFs and RAM elements but more LUTs and DSP slices.

In terms of performance RUNet achieves the highest peak classification accuracy of any DL-based system in the comparison, achieving a peak of 94.46% on the expansive RadioML.2018.01A dataset [41]. As QMCNet only achieves a peak accuracy of 90.58%, the strategy employed by RUNet is shown to result in performance gain of nearly 4%. The authors state that the deeper network employed by RUNet resulted in decreased accuracy loss when quantising in comparison to the comparatively shallower QMCNet.

3.4.3 MobileNetV3 Hardware Implementation Discussion

The MobileNetV3 [78] architecture was employed by Woo et al. [18] to achieve an implementation size which is comparable to that of RUNet. The MobileNetV3 architecture also includes fewer output channels than generally utilised by traditional CNN implementations such as VGG10 [77, 78]. The authors employ quantisation of the inputs and activation functions to 16-bits, whereas the weights are quantised to 2-bits, henceforth referred to as ternary weights. While this architecture is not pruned, a similar technique called Common Subexpression Elimination (CSE) is employed. CSE identifies mathematical operations which are calculated multiple times within the architecture, it then replaces each instance of that calculation with a single operation and stores the result in memory for reuse. An example of where this may be employed within a CNN is when large kernels are employed, as the kernel moves across an input there may be overlap between the regions processed by the kernel, this means that some input values are multiplied by the same weights multiple times, CSE may identify these redundant operations and perform them only once. CSE is demonstrated to result in the reduction of Multiply and Accumulate (MAC) operations performed by an average of 46.2%.

The final novel contribution from [18] is the introduction of decaying weight training, this is a technique which sets weight values to either ± 1 or $\pm 1/2j$, where the parameter j increases with each training iteration, and therefore approximates 0 towards the end of the training process. The authors state that this novel training method allows for proper calculation of the gradient descent algorithm as only non-zero values are used, therefore leading to stronger weight values.

The design strategy of the MobileNetV3 model results in an implementation size which is similar to that of RUNet. 20% and 162 more FFs and DSP slices are required, but a reduction of 9.7% and 45% is achieved in terms of LUTs and RAM. The average peak classification accuracy is however reduced in comparison to RUNet, MobileNetV3 only obtains a peak of 91.3%, 3.2% lower than what was achieved by RUNet. This was found to be due primarily to the choice to use ternary weights.

3.4.4 Ternary Weight Hardware Implementation Discussion

The final set of models to discuss are the ternary weight networks proposed by Tridgell et al. [17, 79]. These models are based on the VGG10 [77] and ResNet33 [60] architectures. Both model structures feature output channel numbers from 64 to 512. Similarly to MobileNetV3, ternary weights are employed, although in this case the inputs were quantised to only 16-bits. In Table 3.3 and Figure 3.14 it can be seen that a range of models from this work are included, the naming convention of each model is indicative of various features. TW refers to the model employing ternary weights, BA indicates binary activation functions, INCRA indicates activation functions which double in precision as the layer number increases, from 1 to 16, the number at the end of each name refers to the dimensionality of the convolution layers. Therefore, TW-BA-128 uses ternary weights, binary activation functions, and has 1x128 dimensional convolutional layers. None of the proposed models employ pruning, but CSE is incorporated into the design.

Comparing the utilisation of these ternary weight models to MobileNetV3 and RUNet shows that the design methodology has not resulted in an implementation of comparable size.

In every category, every model proposed by Tridgell et al. is at least an order of magnitude larger than achieved by the previously discussed CNNs. This huge implementation is obtained even despite quantising to a higher degree, performing CSE, and utilising fewer layers. The comparatively large FPGA utilisation is likely to be due to the number of output channels after each layer. The complexity of an 8-bit quantised VGG10 model was provided in [39], in this case the required number of bit operations and weight bits was also orders of magnitude greater than that of RUNet and QMCNet.

Furthermore, it can be seen in Figure 3.14 that the peak accuracy achieved by this class of system is significantly lower, with TW-96 achieving a peak accuracy of 82.4% which is 12.4% lower than RUNet and 9.2% lower than MobileNetV3. The difference in performance may primarily be attributed to the usage of ternary weights without utilising decaying weight training as was the case with MobileNetV3. Kumar et al. [39] also stated that the deeper network of RUNet provided reduced accuracy loss due to quantisation, as these ternary weight networks are comparatively shallower this may also be a contributing factor.

3.4.5 Comparison with a Hardware Optimised LSTM

No hardware implemented LSTM AMC system could be found in the literature. To investigate if an LSTM implementation could be advantageous, a quick comparison to the smallest general LSTM structure which could be found in the literature is provided.

The smallest general LSTM implementation which could be found in the literature is the V-LSTM proposed by Kim et al. [80]. The system is pruned with a scheme known as Viterbi-pruning and quantised to 4-bit weights, 1-bit indexes, and 16-bit activation functions.

The optimisation strategy achieved an implementation which requires 172413 FFs, 350372 LUTs, 113 BRAM blocks, and 24 DSP slices. The results provided in the text are given in terms of % utilisation of the VC709 board which features the Virtex-7 XC7VX690T FPGA [81]. The datasheet of the XC7VX690T was utilised to convert the percentage values to the raw utilisation.

These implementation statistics place the LSTM among the largest in terms of utilisation of any system shown in Table 3.3. The number of required FFs and LUTs are respectively 707.3% and 913.7% greater than required by RUNet, whereas the RAM and DSP utilisation is more similar to what was required by the most optimised CNN implementations. Therefore, even the most highly optimised LSTM configuration is still multiple times larger than what has been achieved by efficient CNN implementations.

3.4.6 Hardware Comparison Conclusion

The comparison between hardware implemented AMC systems has provided several insights. Firstly, the utilisation of both feature-based systems is smaller than that of even the most highly optimised DL-based models in all but 2 cases, these cases being the required number of DSPs for [39] , and the RAM utilisation of [18]. Conversely, the DL classifiers were found to have approximately half the required latency. However, neither feature-based model was shown to be capable of classifying the extensive set of modulation schemes of the DL models. It is the case that a more complex feature-based classifier would be required to accept such a broad range of inputs, the authors of HISTO-SVM state that the number of binary SVMs

required B to classify a number of classes P scales with the relationship shown in Equation 3.2.

$$B = \frac{P(P-1)}{2} \tag{3.2}$$

A 24-class dataset as was used to train and test the DL models would necessitate 276 binary SVM classifiers rather than the 6 which were employed. Although in this case it would be advantageous to select a classifier structure with greater flexibility such as an MLP.

When evaluating the optimisation strategies of the CNN-based classifiers, quantisation, pruning, and CSE were each found to be effective at reducing the implementation requirements. QMCNet saw a reduction in terms of required bit operations and weight bits of 98.4% and 88.2% respectively with quantisation, pruning provided a further gain of 58.3% and 58.6% respectively. For MobileNetV3 CSE was found to result in an average reduction in MAC operations required per layer in by 46.3%. However, the smallest CNN implementation sizes were obtained when the structure of the CNN itself was optimised to minimise the number of output channels per layer. Both MobileNetV3 and RUNet employed a range of larger kernel sizes and fewer output channels compared to the range of ternary weight networks proposed by Tridgell et al. [17], the strategy resulted in RUNet respectively requiring 93.4% and 83.6% fewer FFs and LUTs compared to TW-INCRA-128. It seems that while quantisation, pruning, and CSE can provide significant reductions in utilisation, beginning with a smaller utilisation before these optimisation mechanisms are applied is also required to achieve the smallest possible network.

In terms of model performance, quantisation was found to impose a penalty depending upon the degree to which it was performed. Tridgell et al. [17] saw a large decrease in peak accuracy by 12.4% via employing ternary weights. However, Woo et al. [18] proposed a training strategy called decaying weight networks which minimised this accuracy difference to only 3.2%. Kumar et al. [39] found that a deeper network also minimised the loss in accuracy incurred by employing quantisation. Pruning was found to not result in drastic accuracy losses as only network connections which did not make a significant contribution were pruned.

3.5 AMC Literature Review: Section Conclusion

This section of the literature review has focused on AMC techniques and explored the state-of-the-art of both software and hardware implementations. Comparisons between feature-based methods revealed that clustering methods are capable of providing equivalent peak accuracy to statistical I/Q based methods, even when higher-order modulation schemes were included in the dataset. The Subtractive Clustering method [51] maintained 100% accuracy to an SNR as low as 16dB on a dataset which included 128QAM and 256QAM. In contrast, the Cumulant Classifier by Alarabi et al. [50] was perhaps the strongest statistical method and maintained 100% accuracy to only 18dB SNR.

Comparisons between DL models revealed that systems which accept constellation diagrm inputs can achieve 100% accuracy whereas those which accept I/Q waveforms cannot. This was found to be the case when comparing systems which use datasets with a maximum modulation order of both 64 and 1024. DL Models in general display greater robustness

to SNR than feature-based methods, maintaining peak accuracy to as low as 3dB SNR, where feature-based models generally lost peak accuracy between an SNR of aproximately 14dB to 18dB. I/Q accepting models were found to exhibit greater SNR robustness than constellation diagram accepting models. The exception to these findings was found to be when the Subtractive Clustering algorithm [51] was tested on a reduced dataset of only 4QAM, 16QAM, 64QAM, and 256QAM, in this case 100% accuracy was maintained to a lower SNR than all high-order DL classifiers [16, 19, 58].

There was not a large range of hardware implemented AMC models found in the literature, only 2 feature-based and 7 I/Q accepting CNNs were found. Techniques such as CSE, pruning, and quantisation were found to be effective strategies for minimising CNN implementation sizes but it was thought that minimising the number of output channels per layer was the optimum method of reducing implementation sizes. The 2 feature-based methods [20,21] were found to have smaller implementations than even the most optimised CNNs [18,39]. None of the hardware implemented CNNs achieved 100% accuracy, primarily due to employing I/Q accepting structures, based on earlier findings a constellation diagram accepting model would be required to achieve this level of performance, yet such an implementation may require a larger implementation size due to the dataset dimensions [54].

While the discussed optimisation strategies may provide CNN implementation sizes which approach the utilisation of the feature-based approaches, they did not result in smaller implementations. They also failed to reach the 100% accuracy achieved by constellation diagram accepting CNNs [58,70] and feature-based classifiers [50,51]. Based on these findings, promising candidates for the realisation of accurate and hardware-efficient AMC systems are therefore hardware implemented image classifiers (which may result in larger implementations) or clustering-based feature extraction classifiers.

While accurate and efficient AMC is a critical function for CR development, it was discussed in Chapter 1 that accurate SNR estimation is also a primary requirement. This literature review will now explore the field of NDA SNR estimation methods and evaluate their suitability for integration with AMC systems within a complete CR system.

3.6 NDA SNR Estimation Literature Review

This section of the literature review explores the various techniques which have been proposed to achieve NDA SNR estimation. Particular attention is paid towards technologies which are most suitable for deployment alongside an AMC system in a CR enabled communications system. The NDA SNR estimation system should therefore be capable of operating with a high accuracy, across a large SNR range, and on numerous different modulation schemes. A system which possesses these characteristics is necessary for accurate SNR estimation in a CR system which employs dynamic modulation scheme adaptation.

The following section provides an overview of the various NDA SNR estimation methods which have been proposed. Following this overview, the performance of each method is evaluated. Finally, limitations of the state-of-the-art systems are identified, and promising avenues of improvement are outlined.

3.6.1 Algorithmic NDA SNR Estimation Methods

Blind or NDA SNR estimation technologies may be categorised into two disparate approaches; those which leverage DL and those which are purely algorithmic. This discussion begins with the algorithmic approaches.

Much like the feature-based approaches which were discussed for AMC, statistical cumulants and moments may be employed to achieve a measure of a transmission's SNR. Pauluzzi et al. published two works at the end of the 20th century which provided a comparison between a collection of the most promising techniques at that time [22, 25]. The first algorithm to discuss was proposed by Gilchriest [82] and called the SMV. As the name implies the method arrives at an estimate of the SNR by finding the mean and variance of the absolute value of a received signal sample $y(n)$ of length N, the square of the mean is then divided by the difference between the variance and the square of the mean, the formula is expressed in Equation 3.3.

$$SNR = \frac{\left(\frac{1}{N}\sum_{n=1}^{N}|y(n)|\right)^2}{\left(\left(\frac{1}{N}\sum_{n=1}^{N}|y(n)|^2\right) - \left(\frac{1}{N}\sum_{n=1}^{N}|y(n)|\right)^2\right)} \tag{3.3}$$

A similar method proposed by Benedict et al. [83] uses the second and fourth-order moments M_2 and M_4, it is known as the M_2M_4 method, the formula for this method is shown in Equation 3.4.

$$SNR = \frac{\frac{1}{2}\sqrt{6M_2^2 - 2M_4}}{M_2 - \frac{1}{2}\sqrt{6M_2^2 - 2M_4}} \tag{3.4}$$

The Split-Symbol Moments Estimator (SSME) was first proposed by Shah et al. [84]. the method relies upon the outputs of two accumulators which each operate on separate halves of a single symbol, the product of the outputs provides an estimate of the signal power, an estimate of the total power is obtained by summing the accumulator outputs. Then, by integrating over the entire symbol, squaring the sum, and averaging the result of this process over a number of symbols, an estimate of the SNR is obtained.

The final Method discussed in these comparison articles is the Maximum Likelihood approach proposed by Thomas [85]. By taking the partial derivatives of the Maximum Likelihood function expressed in terms of the received signal S and noise N, setting the results to 0, and solving for S and N, Equation 3.5 is obtained.

$$SNR = \frac{\frac{1}{P^2}\left(\frac{1}{M}\sum_{k=1}^{M} r_k m_k^*\right)^2}{\frac{1}{M}\sum_{k=1}^{M}\left(r_k^2 - \frac{1}{P}\left(\frac{1}{M}\sum_{k=1}^{M} r_k m_k^*\right)^2\right)} \tag{3.5}$$

Where M is the number of samples, $P = \left(\frac{1}{M}\sum_{k=1}^{M} m_k^*\right)^2$, r_k the received signal, and m_k^* represents the remodulated data sequence formed at the receiver. Pauluzzi et al. emphasise the similarity between equations 3.5 and 3.3, demonstrating how two separate methods arrive at similar expressions for SNR estimation.

Z. Zuo et al. propose an SNR estimation algorithm based on the signal envelope [26]. This method may be seen as an extension of the SMV algorithm which enables operation on differing orders of QAM. The proposed formula is shown in Equation 3.6.

$$\hat{\rho}_{MQAM} = \frac{-(\lambda_{MQAM} - 1) + \sqrt{\frac{a}{a+b}(1 - \lambda_{MQAM})}}{\lambda_{MQAM} - \frac{b}{a+b}} \tag{3.6}$$

The variable a and b are precalculated values which differ between the various orders of QAM, they are therefore used to tune the algorithm based upon the order which is to have its SNR estimated. λ_{MQAM} is a variable which is obtained with the formula provided in Equation 3.7.

$$\lambda_{MQAM} = \frac{\mathrm{Var}(\gamma)}{(E[\gamma])^2} \tag{3.7}$$

Which is equivalent to the variance of the signal envelope γ divided by the square of the mean of the signal envelope γ, which resembles the formula for the SMV estimation algorithm.

W. Wang et al. propose an Empirical Distribution Function (EDF)-based SNR estimation technique [86]. It operates on the principle of comparing the EDF of the received signal to a set of theoretical Cumulative Distribution Functions (CDF) calculated for various hypothesized SNR values. The SNR hypothesis corresponding to the CDF that best matches the EDF is then selected as the estimated SNR. However, instead of performing a full comparison across all possible values of the signal envelope (as in the standard Kolmogorov-Smirnov test [87]), the authors introduce a simplified matching process. They propose a new test statistic that reduces the comparison from two dimensions to one, significantly lowering the computational complexity.

H. Xu et al. [27] propose using the mean value of the square of sampled signal S over the mean value of the absolute values of the sampled signal S squared, the formula is shown in Equation 3.8.

$$Z = \frac{\mathrm{mean}(S^2)}{(\mathrm{mean}(|S|))^2} \tag{3.8}$$

The authors state that there is a relationship between the value of Z and the SNR of the signal which can be described by a 5th order polynomial. The relationship between Z and the SNR is different for each modulation scheme, the authors therefore provide different polynomial weights (C_0 to C_5) which may be modified to allow for accurate estimation of a range of QAM modulation formats. The mapping of Z to the SNR is shown in Equation 3.9.

$$SNR = C_5 Z^5 + C_4 Z^4 + C_3 Z^3 + C_2 Z^2 + C_1 Z + C_0 \tag{3.9}$$

Another similar technique was proposed by M. Álvarez-Díaz et al. [28]. This method is known as the Eighth-Order Statistics (EOS)-based method. In this case the ratios of various even-ordered moments multiplied by constants are transformed into a 4th order polynomial, the roots of the polynomial then result in an estimate of the SNR. The first step is to calculate the even ordered moments of the received signal M_2, M_4, M_6, and M_8, and arrange them in the form shown in Equation 3.10.

$$f_{EOS}(\rho) = \beta\frac{M_4}{M_2^2} + \gamma\frac{M_6}{M_2^3} + \delta\frac{M_4^2}{M_2^4} + \epsilon\frac{M_8}{M_2^4} \tag{3.10}$$

Where ρ is equal to the SNR. By applying the variable change of $\rho = z/(z-1)$, the authors state that the polynomial shown in Equation 3.11 is obtained.

$$F_{EOS}(z) = F_4 z^4 + F_3 z^3 + F_2 z^2 + F_0 \tag{3.11}$$

Where the coefficients F_k are linear and are given by Equations 3.12 to 3.15:

$$F_4 = \delta(c_4 - 2)^2 + \epsilon[72(c_4 - 1) - 16c_6 + c_8] \tag{3.12}$$

$$F_3 = (\gamma + 16\epsilon)(1 - 9c_4 + c_6) \tag{3.13}$$

$$F_2 = (\beta + 9\gamma + 4\delta + 72\epsilon)(c_4 - 2) \tag{3.14}$$

$$F_0 = 2(\beta + 3\gamma + 2\delta + 12\epsilon) \tag{3.15}$$

Where c_4, c_6, and c_8 represent the moment ratios given in Equation 3.10. By finding the roots of Z, a measure of the SNR is obtained. The values of β, γ, δ, and ε are all tuned to suit the modulation scheme of the signal; via this tuning, adaptation to a range of schemes is achieved.

The final polynomial fitting method to discuss utilizes the DBSCAN algorithm [38]. This method uses the ratio R of core points $|\Omega|$ to total cluster points $|D|$ found by the DBSCAN algorithm on a particular constellation diagram. The ratio R is defined in Equation 3.16.

$$R = \frac{|\Omega|}{|D|} \tag{3.16}$$

The concept of core points and cluster points is explained in greater detail in the following chapter of this thesis. The key to understanding this method is that this ratio is used as a proxy for describing the density of the constellation diagram. As the SNR falls the constellation points increase in size and therefore reduce in density, the DBSCAN algorithm thus finds fewer core points at lower SNRs and more core points at higher SNRs. Figure 3.15 illustrates an example of the number of core points found by DBSCAN when operating on 16QAM at an SNR of 20dB and 10dB.

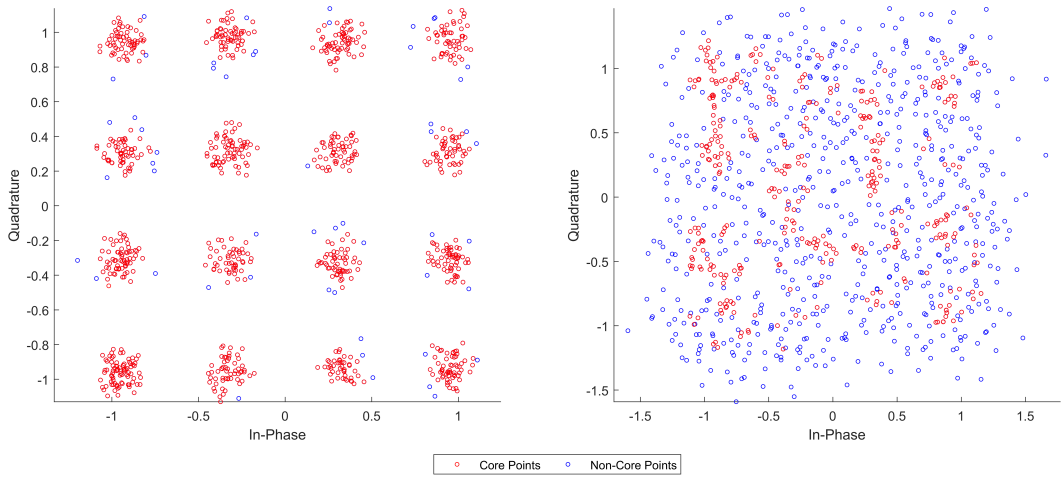


Figure 3.15: *An Example of the Ratio of Core Points (Red) to Non-Core Points (Blue) by the DBSCAN Algorithm on 16QAM Data at an SNR of 20dB (Left) and 10dB (Right)*

How the value of R varies with respect to the SNR differs between modulation schemes but may be described by a fitted 4th order polynomial such as shown in Equation 3.17.

$$f_4(x, w) = w_0 + w_1 x + w_2 x^2 + w_3 x^3 + w_4 x^4 \tag{3.17}$$

Similarly to the previous polynomial fitting techniques, the constants w_0 to w_4 may be varied according to the signal's modulation scheme.

3.6.2 DL NDA SNR Estimation Methods

The application of DL to NDA SNR estimation has emerged as a competitive approach in recent years. Similarly to the field of AMC, the CNN is the primary model structure which has been employed, with different works applying a variety of input formats with the aim of obtaining improvements in performance.

S. Zheng et al. [24] propose the use of ResNet with an input of raw I/Q data as well as I/Q data transformed to form both a periodogram and an average periodogram. Both periodogram formats provide an estimate of the spectral density of the signal by performing a Discrete Fourier Transform (DFT) and squaring the output of each frequency component, the average periodogram performs this by taking the average of multiple overlapping segments. The model features 3 convolutional layers with a single MaxPool layer between the final convolution layer and the output. K. Yang et al. [30] also propose an I/Q accepting CNN, in this case the authors opt for a 5-layer CNN with a MaxPool layer following each convolutional layer.

The constellation diagram has also been proposed as a viable CNN input. X. Xie et al. [29] preprocess the constellation diagrams with an exponential decay model, the resulting images which are used as inputs are constructed from 3 constellations derived from the same signal sample but have different exponential decay rates applied. This preprocessing operation results in clearer constellation diagram images, with a spectrum of colours representing regions

of varying constellation point density. They test the strategy with 3 different CNN model structures, AlexNet [59], InceptionV1 [76], and VGG16 [77]. All three models are highly complex implementations as they feature 8, 22, and 16 total layers respectively.

S. Chen et al. [23] propose the usage of a covariance matrix input. The covariance matrix contains the variance and covariance of an N-point I/Q sample, the diagonals of the matrix hold the variance values, the non-diagonal elements hold the covariance. The text states that the diagonal variance values show a strong correlation between various SNRs whereas the non-diagonal covariance values vary significantly between SNRs. It is implied therefore that the diagonals capture information about the signal power (which should strongly correlate across SNRs) and the non-diagonals capture noise power information (which should be uncorrelated). This difference in correlation structure is the information which the CNN learns from, enabling SNR estimation functionality to be achieved. The authors also state that the reduced dimensionality of the covariance matrix compared to a raw I/Q sample or constellation diagram enables a CNN with lower computational complexity to be employed.

All relevant NDA SNR estimation techniques how now been introduced, the following section provides a comparison between the performance of each technique with the ultimate aim of identifying the strengths and weaknesses of the various methods.

3.6.3 NDA SNR Estimation Performance Comparison

Results in this section will be provided in terms of MSE against SNR as well as Mean Absolute Error (MAE) against SNR in one case. Many works opt to provide their results in terms of Normalised Mean Square Error (NMSE), which is the MSE normalised by the square of the SNR. The NMSE metric seems to obfuscate the true performance of a system, for instance Pauluzzi et al. [22] claim that an unchanging NMSE value demonstrates a well-behaved system, however a stable NMSE value as the SNR increases implies an increase in MSE with respect to the SNR. Therefore, works which provide results in terms of NMSE will have their performance converted to MSE where possible as the metric has been found to provide a more intuitive means of comparing performance. One work opts to provide performance in terms of MAE, it is impossible to convert MAE to MSE without knowledge of the full test statistics, therefore this work cannot be compared with the other systems on the same figure. Comparisons to the proposed system in Chapter 8 will be performed using both MSE and MAE.

It was mentioned in the previous section that many estimation methods included reconfigurable parameters to enable accurate estimation of a range of modulation schemes, the algorithmic methods which do not include such functionality are therefore limited to operation on a single modulation scheme. This fact discounts the usage of these techniques in an adaptive modulation scenario, however the results obtained by these methods are provided to illustrate the performance differential between these methods and those which are reconfigurable.

Beginning by first comparing the 4 estimation techniques compared in the comparison article by Pauluzzi et al. [22, 25], the tests performed to obtain the following results were performed using the 2PAM modulation scheme, it is therefore an unfair comparison to utilise these results when comparing to other works which utilise more complex modulation schemes such as m-PSK and m-QAM. Figure 3.16 displays the reported MSE against SNR characteristics.

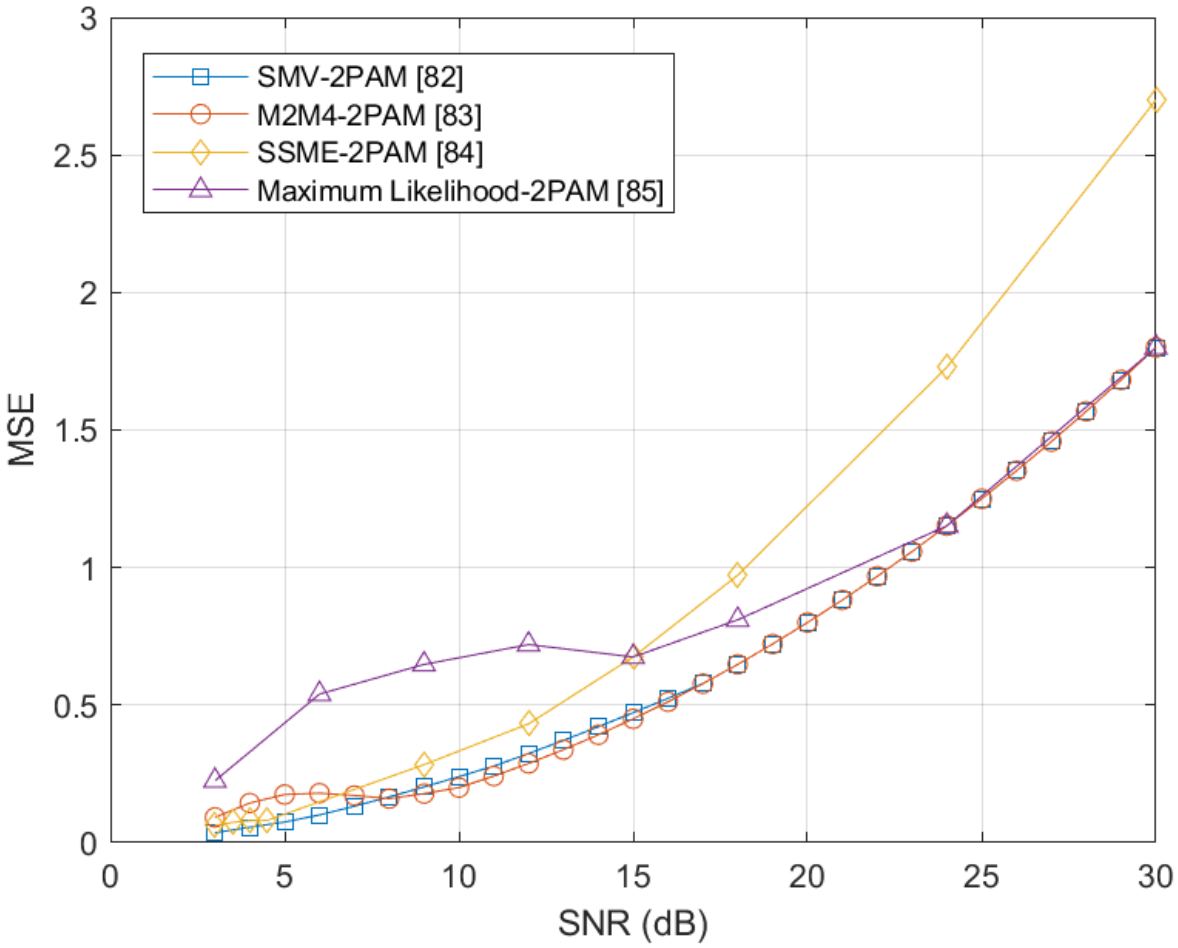


Figure 3.16: *MSE Against SNR (dB) Estimation Performance for the 2PAM Algorithmic Estimators Compared in [22]*

The MSE against SNR curves show that both the SMV and M_2M_4 methods consistently achieve a lower MSE than both the SSME and Maximum Likelihood methods. Maximum Likelihood has a weaker MSE between 0dB and 24dB, above which the performance matches that of SMV and M_2M_4, SSME begins with a comparable accuracy to the 2 strongest methods, but the MSE increases with a greater gradient as the SNR increases. As M_2M_4 and SMV both are shown to provide the strongest performance, they are typically used as a benchmark for comparison in later works, more commonly M_2M_4 is utilised for this role. Zheng et al. [24] provide MSE against SNR characteristics for the M_2M_4 method as they utilise this method as their benchmark, further instances of M_2M_4 results being provided in this literature review will utilise the M_2M_4 QPSK performance provided by Zheng et al. [24].

Next, M_2M_4 will be compared with the various CNN SNR estimators. All but 1 of the CNN systems only provide results when test data of QPSK was utilised, Figure 3.17 displays a comparison of the results.

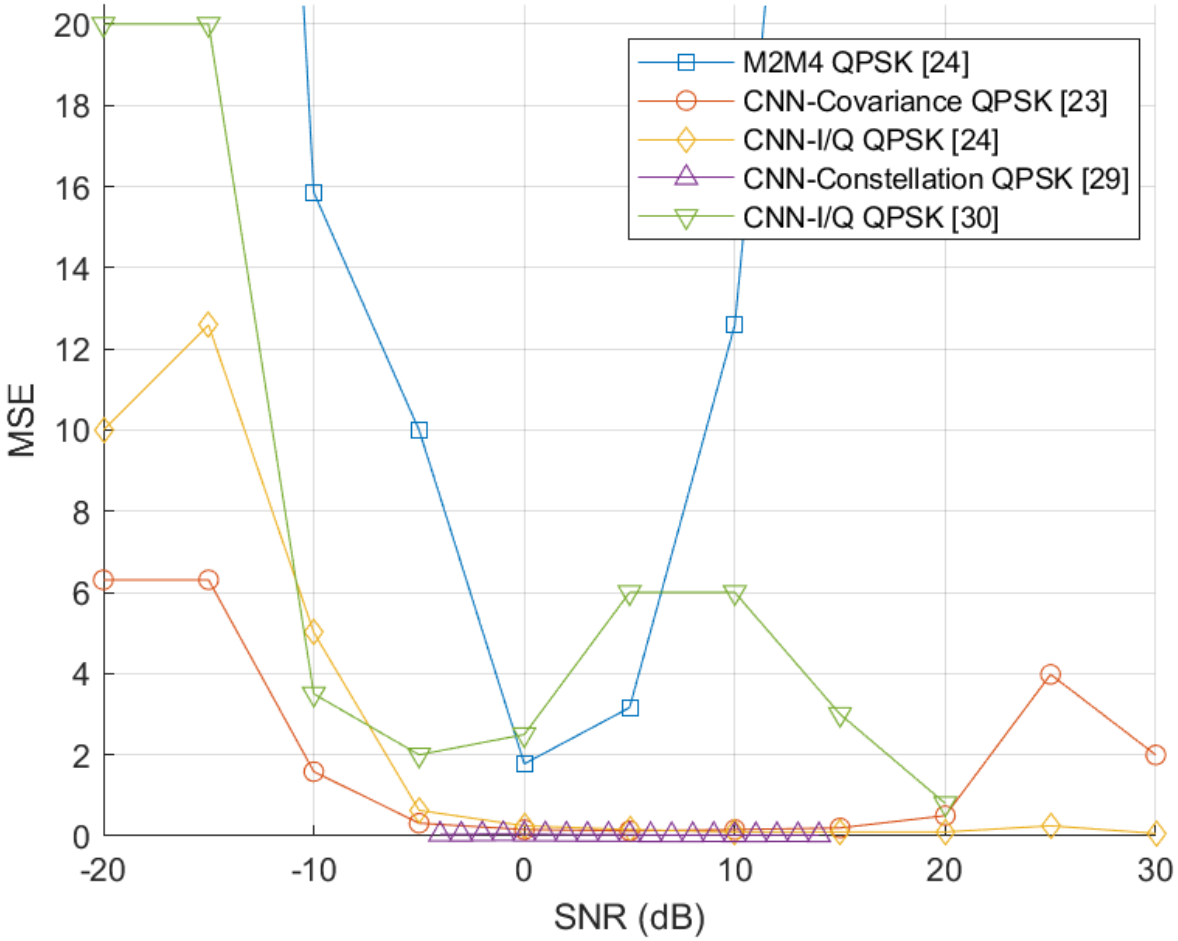


Figure 3.17: *MSE Against SNR (dB) Estimation Performance for the DL Estimators [23,24,29,30] and M_2M_4 with a QPSK Input [24]. M_2M_4 Data Outside of the $\pm$11dB Range Lies Outside of the Y-axis Scale but the Trend of MSE Increase Continues. The Omission is to Display CNN Accuracy More Clearly.*

Firstly, the change in M_2M_4 MSE against SNR performance is stark in comparison to the results obtained when 2PAM was employed as the test data. In this case a reasonably strong MSE is only achieved in the SNR range of 0dB to 5dB, in all other cases the MSE is significantly higher than that of the CNNs. Inspection of the estimated SNR against true SNR figure shown in [24] shows that the output of M_2M_4 algorithm asymptotes -5dB and 10dB at all SNRs below and above these values respectively.

While all 3 of the CNNs which provide data in the range of -20dB to -10dB SNR each show a large increase in MSE [23,24,30], the increase is less significant than that of M_2M_4. Between -5dB and 20dB SNR 3 of the CNN systems achieve an MSE close to 0 [23, 24, 29], implying a strong SNR estimation accuracy in this range. The CNN which accepts only a raw I/Q [30] input fails to match this performance, implying that the constellation diagram [29], covariance matrix [23], and periodogram [24] inputs enable the systems to reach a stronger level of performance by providing more discriminatory data. Above 20dB SNR the covariance matrix accepting CNN [23] sees a slight increase in MSE whereas the periodogram and constellation

accepting models maintain an MSE close to 0.

The results shown in Figure 3.17 demonstrate that DL methods can provide increased SNR estimation performance in comparison to early algorithmic SNR estimation techniques, stronger accuracy was obtained not only over the full SNR range but also at the SNRs at which M_2M_4 performed best.

Next, the polynomial fitting methods proposed by M. Álvarez-Díaz et al. [28] and Xu et al. [27] as well as the EDF based estimator by Wang et al. [86] are compared. Each work provides accuracy when 16QAM and 16APSK are used as the test data, Figure 3.18 displays the MSE against SNR characteristics.

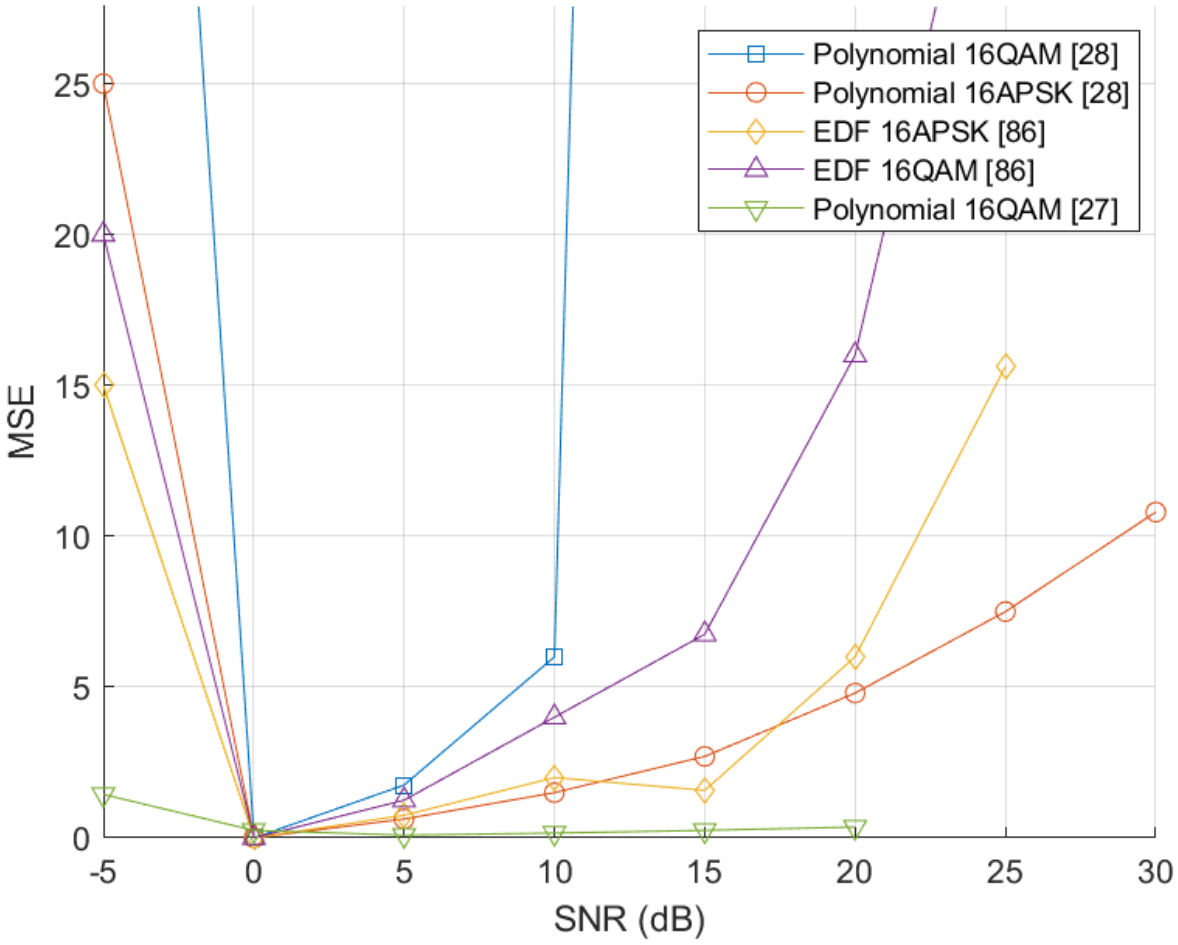


Figure 3.18: *MSE Against SNR (dB) Estimation Performance for the Polynomial Estimators [27,28] and EDF Estimator [86] with a 16QAM and 16APSK Input. Some Polynomial and EDF 16QAM Datapoints Lie Outside of the Y-axis Scale but the Trend of MSE Increase Continues, The Datapoints Were Omitted to Better Display the Trends Across other Curves.*

The results shown in Figure 3.18 demonstrate that these algorithmic estimators have a similar weakness to the M_2M_4 method in that optimum performance is obtained in the 0dB to 10dB SNR range, outside of this range estimation accuracy sees a significant reduction. The exception to this is the polynomial fitting method [27] which does exibit an MSE increase outside

of the 0dB to 10dB SNR range but the increase is significantly less severe in comparison to the 2 other techniques in the comparison.

Figure 3.19 compares the 32QAM, 32APSK, and 64QAM performance of the polynomial fitting method proposed by M. Álvarez-Díaz et al. [28], with the I/Q accepting CNN by Yang et al. [30], the envelope-based technique proposed by Zuo et al. [26], and the polynomial fitting method proposed by Xu et al. [27].

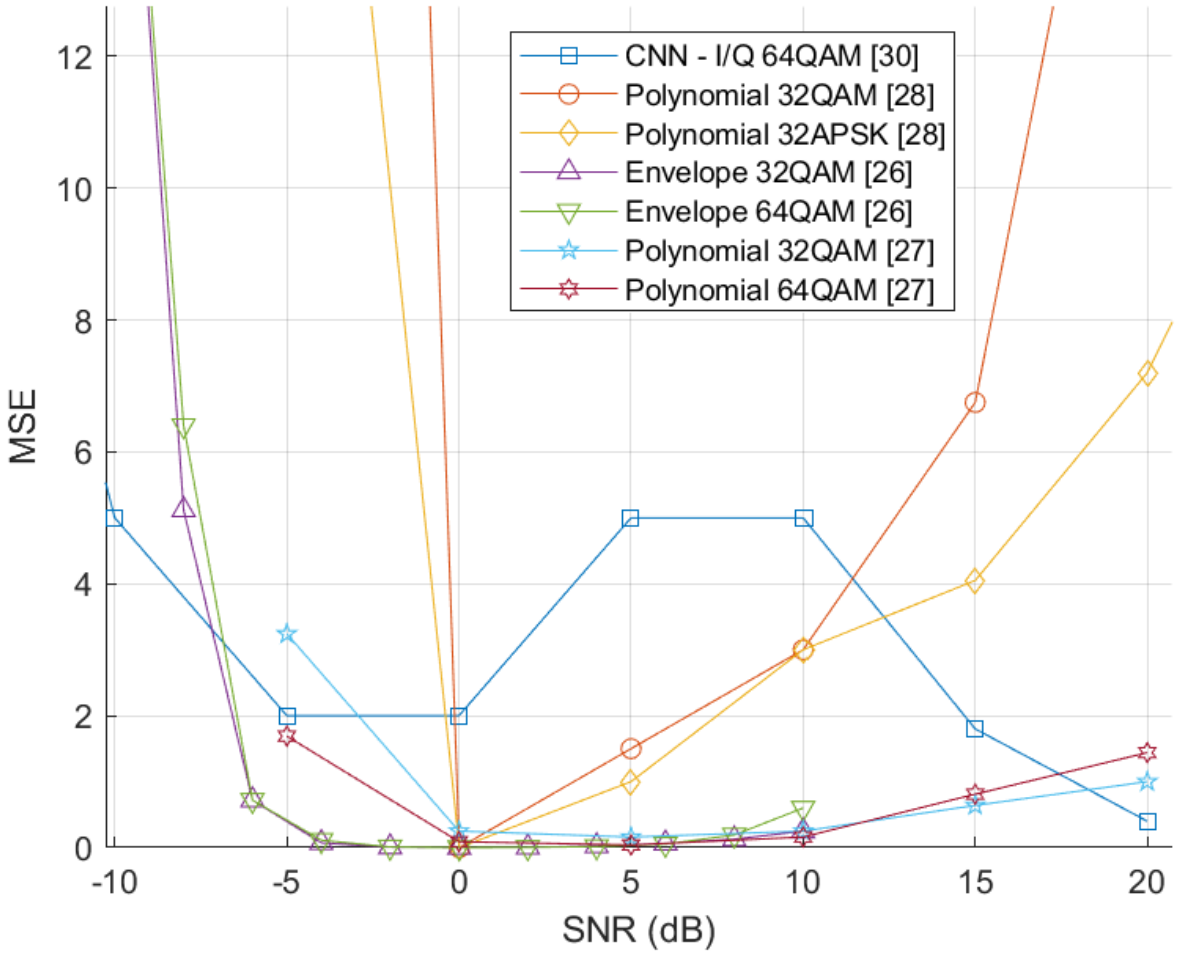


Figure 3.19: *MSE Against SNR (dB) Estimation Performance for the Polynomial Estimators [27,28], Envelope Estimator [26], and I/Q Accepting CNN [30] with a 32QAM, 32APSK, and 64QAM Input. Some Polynomial [28] 32QAM and 32APSK Datapoints Lie Outside the Scale of the Y-axis but the Trend of MSE Increase Continues. Datapoints Were Omitted to Better Display the Trends in Other Curves*

The polynomial fitting technique proposed by M. Álvarez-Díaz et al. [28] is once again shown to only provide strong performance in the 0dB to 10dB SNR range, in this case the MSE sees an even more severe increase at either end of this SNR range. Zuo et al. [26] do not report MSE statistics at an SNR greater than 10dB, judging by the slight increase in MSE seen between 9dB and 10dB SNR it is likely that a large increase in MSE would be observed

at higher SNRs, as has been seen to be the case for all other algorithmic techniques. Zuo's envelope method [26] does achieve a low MSE in the -4dB to 9dB SNR range, consistently achieving an MSE very close to 0, demonstrating that this technique achieves very low error within this range. While the CNN-based estimator [30] does not obtain as low an MSE as [26, 28] in the SNR ranges at which they are shown to be optimal, the obtained MSE against SNR performance demonstrates no asymptotic behaviour. This finding is consistent with those shown in Figure 3.17 where DL-based methods are shown to be more capable of accurate SNR estimation across a wider range of SNRs. Perhaps the strongest performing system is again shown to be the polynomial fitting method proposed by Xu et al. [27]. This method is shown to match the low MSE obtained by [26], consistently achieve a lower MSE than [30], and does not exhibit as large of an increase in MSE outside of the 0dB to 10dB SNR range as [28] and perhaps [26].

The final comparisons to draw are between the two works which provide results across a wide range of modulation schemes, demonstrating a necessary capability which an SNR estimator should possess as part of a CR enabled system. These methods are the polynomial fitting technique proposed by H. Xu et. al [27] and the DBSCAN SNR estimator proposed by Zhao et al. [38]. Unfortunately, these two works report estimation accuracy with different metrics, making comparisons less direct. There are however several insights which can be drawn from the following comparison. Figure 3.20 exhibits the MSE against SNR performance of the polynomial fitting method [27]. Figure 3.21 shows the MAE against SNR performance achieved by the DBSCAN SNR estimation system [38].

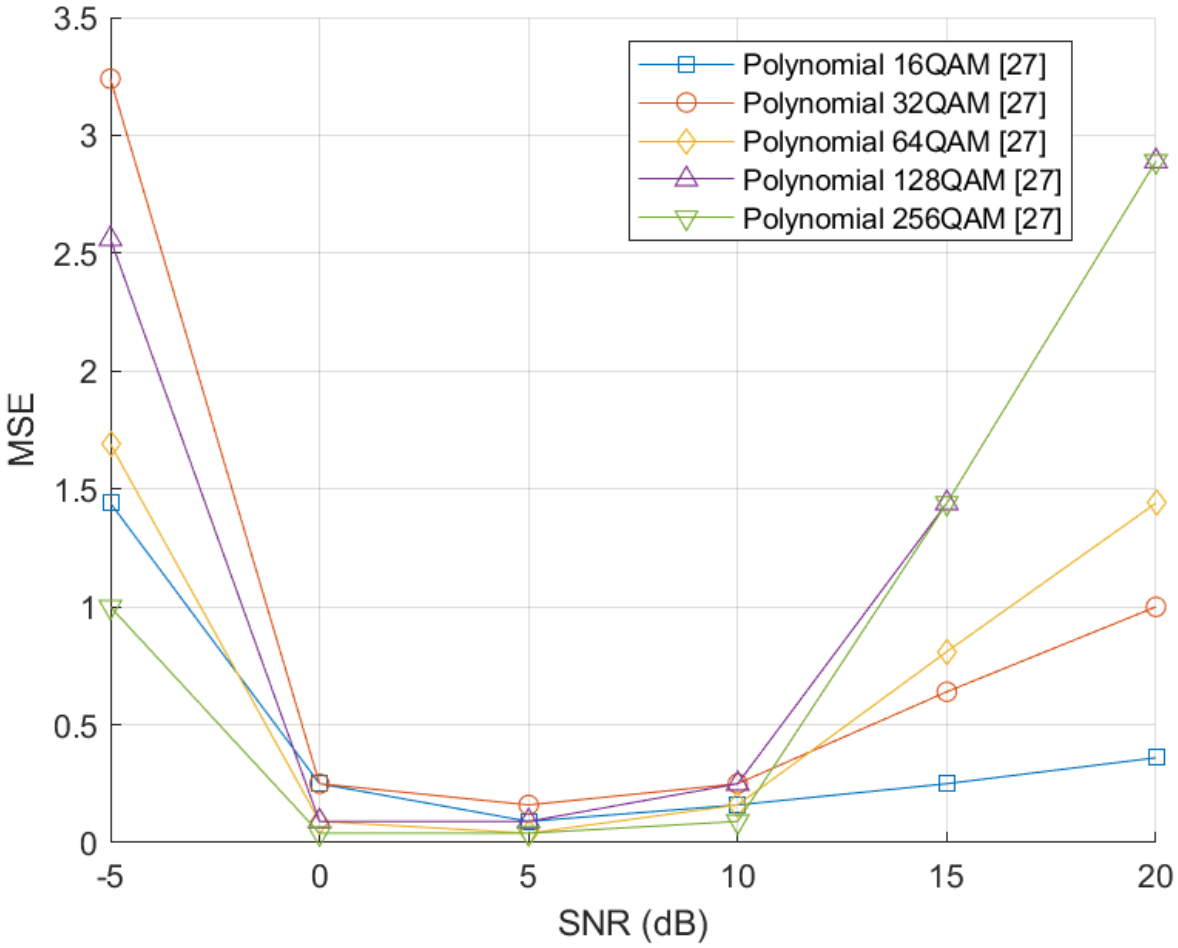


Figure 3.20: *MSE Against SNR (dB) Estimation Performance for the Polynomial Estimator [27] with a 16/32/64/128/256QAM Input*

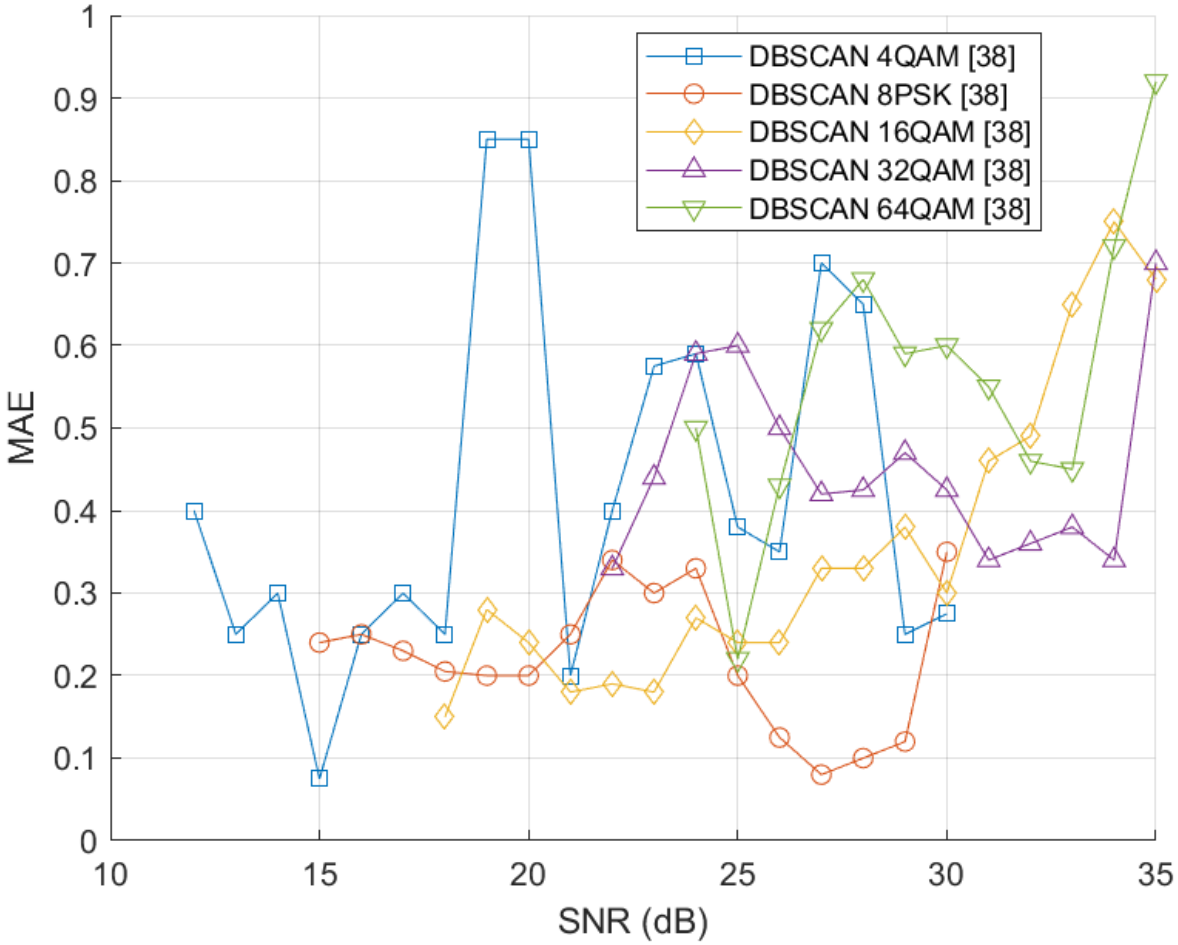


Figure 3.21: *MAE Against SNR (dB) Estimation Performance for the DBSCAN Estimator [38] with a 4/16/32/64QAM and 8PSK Input*

The first trend to identify is that the polynomial fitting method [27] once again demonstrates that the region of strongest performance lies between 0dB and 10dB SNR, continuing the trend of algorithmic estimators performing well only within this region. The performance of this method on 16QAM, 32QAM, and 64QAM was demonstrated to be superior to other algorithmic methods in Figures 3.18 and 3.19, particularly as the MSE increase outside of the 0dB to 10dB range was found to be less severe. However, in Figure 3.20 it can be seen that the MSE of 128QAM and 256QAM increases at a greater rate in comparison to the lower-order modulation schemes. As 128QAM and 256QAM would generally be expected to be employed at SNRs greater than 10dB in a CR system, it is unfortunate that the performance degrades to such a degree.

Conversely, the DBSCAN-based estimator [38] is shown to achieve optimum performance between an SNR of 12dB and 35dB. The DBSCAN method is perhaps more akin to the DL-based techniques as it relies on machine learning methods, requiring a feature extraction algorithm followed by a classifier, this is thought to be why it is shown to be capable of performing well outside of the 0dB to 10dB SNR range. The only trend which could be identified with the DBSCAN estimation accuracy performance is that the MAE is in general

higher at higher SNRs than it is at lower SNRs.

The range of SNRs for which the authors of the DBSCAN technique provide MAE results varies between employed modulation schemes, the reported SNR range is higher for the higher-order signals and conversely lower for the lower-order signals. It is thought that at SNRs below which there are provided MAE values that the DBSCAN estimator probably suffers a large increase in MAE. When DBSCAN was utilised for AMC, it was found that the technique's performance at SNRs below 15dB was poor as the constellation diagram became too noisy for the algorithm to extract any meaningful information [37, 38]. Furthermore, the SNR at which the algorithm lost perfect classification accuracy was proportional to the modulation order, with 64QAM falling to 20% accuracy at 13dB, 16QAM at 11dB, and 4QAM at 9dB. It is therefore reasonable to expect that the SNR estimation performance achieved by this method would also be poor below these SNRs.

Of all the NDA SNR estimation methods covered here, only the DL [23,24,29,30] methods and DBSCAN [38] have demonstrated consistent accuracy outside of the 0dB to 10dB SNR range. Only DBSCAN [38] and Xu's polynomial fitting method [27] have demonstrated strong accuracy across a wide range of signals.

Following this comparison, the performance of the strongest NDA SNR estimation systems has been evaluated and compared. The next section concludes this NDA SNR estimation section and identifies prime candidates for development.

3.6.4 NDA SNR Estimation Literature Review: Section Conclusion

The comparison between the performance obtained by the various NDA SNR estimation techniques found in the literature revealed several conclusions. Before the conclusions are expanded upon the requirements of an NDA SNR estimation system in a CR enabled system must be reiterated.

Firstly, the system must be capable of estimating the SNR of a wide range of modulation schemes. The goal of the proposed CR system is to enable dynamic modulation scheme adjustments; thus, it is a necessity to have the ability to operate on a large set of modulation schemes.

Secondly, the system must be capable of accurately estimating the SNR across a wide range of SNRs. Information about a received signal's SNR must be obtained with a high degree of confidence regardless of the SNR in order to enable intelligent and informed decision-making.

Finally, the system must be feasible to implement in portable and low-powered hardware. Deployment in portable devices is a key requirement to enable the envisioned CR-enabled beyond 5G systems. Furthermore, minimising utilisation costs and power consumption minimises the expense required to implement such functionality within the required infrastructure. None of the works discussed in this literature review provide statistics for a hardware implementation, therefore comparing the various systems in this regard is difficult, any comparisons discussed henceforth must be seen as an educated guess based upon findings within the AMC portion of this literature review.

With these requirements in mind the SNR estimation performance of the respective methods can be evaluated. The systems which demonstrated the largest SNR range of effectiveness were the CNN-based models [23, 24, 30] other than the constellation diagram accepting model [29]. Each of these systems were exhibited to have a somewhat weaker performance

below an SNR of -5dB but at all SNRs greater than this the MSE remained close to 0. The exception to this was the raw I/Q accepting CNN [30] which did maintain consistent accuracy but said accuracy was lower than that of the other models. Conversely, these systems were only demonstrated to achieve this level of accuracy on QPSK, however, the I/Q accepting CNN [30] was demonstrated to be capable of obtaining a similar level of accuracy with 64QAM as it did with QPSK. While the authors of the other CNN systems reported in the text that a similar level of performance was achieved when employing various other modulation schemes, they neglected to include the results in the form of an MSE against SNR curve and the modulation schemes which were claimed to provide strong performance were not as high-order or varied as was demonstrated for certain algorithmic methods. Further investigation is therefore required to ascertain the exact performance of the various CNN-based methods on a wider range of modulation schemes as the reported results in the literature are inconclusive in this regard. Finally, with respect to the hardware requirements, it is likely that this class of method would require the largest utilisation and power consumption of any SNR estimation system in the comparison. The hardware comparisons in Section 3.4 demonstrated that CNNs require significant resources, however by employing the various optimisation techniques such as pruning, CSE, and quantisation, the required resources could be minimised [18, 39].

Algorithmic NDA SNR estimators were demonstrated to have the ability to operate on a wide range of modulation schemes. In particular, the polynomial fitting method proposed by H. Xu et al. [27] demonstrated a low MSE across orders of QAM from 16 to 256 which would generally be similar to the expected range of modulation orders employed within a CR system [2]. Similarly, the envelope-based estimator by Zuo [26] obtained an MSE comparable to that of the DL-based estimators on both 32QAM and 64QAM. However, all the algorithmic estimators were found to have the fundamental flaw of only being highly accurate within the SNR range of approximately 0dB to 10dB. This flaw discounts them from deployment within a CR system as the high degree of inaccuracy at very high and low SNRs does not allow for the high degree of confidence in the estimated SNR necessary for effective modulation adaptation. This flaw is unfortunate as these systems only require the implementation of a closed form mathematical operation to obtain SNR estimation, such an implementation would likely lead to a smaller and less power consuming implementation than the CNN estimators.

The DBSCAN-based SNR estimator [38] may offer a compromise between the efficiency and applicability to a range of modulation schemes offered by the algorithmic methods and the accuracy across a wider range of SNRs offered by the CNNs. DBSCAN was exhibited to have approximately consistent estimation accuracy across QAM orders 4 to 64, matching the applicability to a variety of modulation scheme demonstrated by the algorithmic techniques. Furthermore, while this level of performance was demonstrated only within a particular SNR range, it was outside the 0dB to 10dB SNR range which no algorithmic method was shown to be capable of performing well outside of. Finally, while the implementation of this method may not be as efficient as many of the algorithmic methods, it is probable that the utilisation and power consumption would remain considerably lower than that of the CNNs. Therefore, it is possible that this technology could match the qualities achieved by other techniques in each of the three stated requirement categories. The primary limiting factor to the realisation of this goal is the achievement of strong estimation accuracy below 15dB SNR, modifications to the algorithm would likely be required to achieve this.

Building upon the insights gained from this exploration of NDA SNR estimation and considering the findings of the prior AMC section of this literature review, the following section concludes this literature review by summarising the findings obtained throughout this review and identifies potential candidates for development.

3.7 Literature Review Conclusion

In the introduction to this thesis, it was stated that the aim of this work was to identify and develop technologies with the potential to realise the creation of an efficient and accurate AMC and SNR estimation system which can be embedded in portable devices to enable CR functionality. This literature review has now fully analysed the state of the AMC and NDA SNR estimation fields, this section will now conclude this literature review by providing a summary of the findings and identifying strong candidates for development.

The portion of this literature review dedicated to AMC first explored the accuracy which may be achieved by software implementations of feature-based and DL modulation classifiers. It was found that the majority of feature-based techniques are unable to match the level of accuracy achieved by the DL methods, particularly at lower SNRs. This finding is exemplified by the performance differential between the feature-based classifier which was found to provide the strongest performance, proposed by Alarabi et. al [50] to all DL methods. While Alarabi's classifier reached 100% accuracy, this level of accuracy was lost at 13dB SNR, while the strongest DL classifier ModNet [71] and FiF-Net [57] maintained 100% accuracy to as low as 4dB and 5dB SNR respectively. Furthermore, M-CNN [58] was demonstrated to maintain 100% accuracy to 10dB on a dataset which included higher order signals than that of [50].

One aspect where the algorithmic methods had the advantage over a particular class of DL model was the inability of the I/Q accepting models to reach 100% accuracy, throughout all the works evaluated there was no example of an I/Q accepting model reaching 100% accuracy, a constellation diagram input was therefore found to be a requirement for a CNN to achieve this level of performance. Clustering-based feature extraction methods were found to approximately match the accuracy of image classifying CNNs, the Subtractive Clustering feature extraction mechanism proposed by Wang et al. [51] was found to be capable of reaching 100% accuracy at SNRs greater than 16dB on a dataset of 6 different modulation schemes of modulation orders up to 256. Furthermore, when employing a dataset which consisted of only modulation schemes currently utilised in 5G communications it was found that 100% accuracy was maintained to as low as 6dB SNR, which was 4dB lower than the best performing DL system on high-order data [58]. These findings demonstrate that it is indeed possible to create a feature-based modulation classifier which outperforms the DL models if the employed set of modulation schemes are limited to a set which are conducive to being distinguished by clustering algorithms.

Evaluation of the implementation costs of classifier technologies was limited to comparing CNNs and a few select feature-based implementations as there is not a wealth of research being conducted in this area. It was found that optimisation techniques such as quantisation, pruning, and CSE were effective at reducing the utilisation of CNN implementations. RUNet [39] was found to achieve a 60% reduction in FF and LUT utilisation compared to a reference CNN implementation. However, the largest gains in utilisation reduction were thought to be obtained via an architectural redesign in which the number of output channels per layer

were minimised and more varied kernel sizes were employed to compensate for lost feature learning abilities. With this methodology RUNet [39] and MobileNetV3 [18] were found to achieve over a 90% reduction in FF utilisation compared with the ternary weight CNNs [17] which were quantised to a greater degree but did not employ this strategy. RUNet and MobileNetV3 were larger than the only feature-based classifier in the comparison but not be a significant amount, in terms of FF and LUT utilisation RUNet was only 28% and 77% larger respectively. MobileNetV3 was the only system to provide values for power consumption and still required 4.2W of power despite the highly optimised implementation. However, the hardware implemented CNNs were I/Q accepting systems which had been previously found to be weaker classifiers than constellation diagram accepting models in terms of peak accuracy. A hardware implementation of a constellation diagram accepting CNN would likely require a larger implementation due to the dimensionality requirements of the layers.

When comparing NDA SNR estimation techniques, it was found that algorithmic methods had been demonstrated to be capable of estimating the SNR of a larger range of modulation schemes than the CNN implementations. However, the SNR range in which the algorithmic methods were accurate was limited to approximately 0dB to 10dB in all cases. DL-based models were found to not suffer such limitations. An estimator employing the DBSCAN algorithm was found to be capable of estimating the SNR of a range of QAM orders with a high degree of accuracy outside of the 0dB to 10dB range, however in this case the SNR bounds of accurate performance was limited to 12dB to 35dB.

3.7.1 Candidate Technologies for Development

Through the comparisons presented in this literature review, two technologies have been identified as being promising candidates for the realisation of a lightweight yet effective AMC and SNR estimation system, image-based CNNs and clustering methods. Both technologies were found to provide 100% modulation classification accuracy to a low SNR, although the clustering method required a more limited dataset to achieve this degree of performance. Both technologies were also found to offer promising results in the field of SNR estimation. The CNN approach was shown to be capable of achieving high accuracy across a large SNR range but lacked a demonstration of this ability on modulation schemes other than QPSK. The DBSCAN clustering method conversely achieved strong accuracy on a range of QAM orders but lacked a demonstration of this level of performance being replicated at SNRs below 12dB. Both methods therefore require development and testing to realise accurate performance across a broader range of SNRs and modulation schemes.

What separates these two technologies is the scope for optimisation. While CNNs have been efficiently implemented via the previously mentioned techniques, these techniques were demonstrated on I/Q accepting models, there is therefore no guarantee that the larger layer dimensionality required for an image processing CNN would enable a similarly sized implementation to be achieved. Furthermore, no clear path to optimise the CNN implementation further could be identified. Development of this methodology would therefore amount to reimplementing a CNN with similar techniques to RUNet and MobileNetV3 and evaluating the performance and utilisation of the system.

Conversely, while clustering methods have not been found to outperform CNNs in all regards, under some conditions promising results have been obtained, namely the DBSCAN SNR estimation and the Subtractive Clustering algorithm AMC performance. No hardware

implementations of clustering algorithm-based systems could be found in either field meaning that there is scope to contribute to knowledge in this respect. Furthermore, the DBSCAN algorithm has demonstrated the capability to perform both AMC and NDA SNR estimation. Although the algorithm was found to lack low SNR performance in both fields, changes to the mechanism of operation may enable performance to be improved in both aspects as well as enable significant reductions in utilisation. Finally, utilising this technology can enable each operation to be performed with a single implementation, further compounding potential utilisation reductions.

To summarise, it was decided that the most viable approach for realising a truly efficient and highly accurate joint AMC and NDA SNR estimation system was development of the DBSCAN algorithm for the following reasons:

- To obtain 100% classification accuracy with a DL model, an image-classifying CNN is required.
- While CNNs can be implemented with minimal utilisation, even the smallest utilisations require 4.2W of power [18].
- An image classifying CNN is likely to be larger than the 2 most efficient I/Q classifying CNNs due to the layer dimensionality.
- A CNN with dual SNR estimation and AMC capabilities would necessitate more utilisation than AMC alone.
- Clustering algorithms are the only technique other than image-classifying CNNs to maintain 100% accuracy to as low an SNR as the CNNs [51].
- While DBSCAN is not demonstrated to classify modulation schemes with as high an accuracy as the subtractive clustering algorithm [37, 51]. It has demonstrated SNR estimation capabilities [38], potentially enabling the creation of a dual-functional system, thereby increasing efficiency.
- Developing a clustering algorithm for hardware contributes to knowledge as such a task has not been completed in the AMC nor NDA SNR estimation fields.

The following chapter expands upon the limitations of the past attempts to utilise this technology and details the numerous improvements to the algorithm which have been developed with the aim of improving accuracy and reducing hardware utilisation.

Chapter 4

An Optimised DBSCAN-Based Classifier

The following chapter of this thesis concerns the principles behind the proposed method of using DBSCAN clustering to generate features for AMC and NDA SNR estimation. As discussed in the literature review, past attempts to apply this technique for feature extraction exhibited limitations in terms of computational complexity and the inability to distinguish modulation schemes of equivalent order. The proposed design addresses these 2 limitations with an architectural modification to the operation of the DBSCAN algorithm.

This chapter begins by outlining the methodology and reasoning behind the modifications which have been made to the DBSCAN algorithm. Section 4.1 details the operation of the original DBSCAN algorithm within the context of AMC, to ensure that the limitations are made clear. Section 4.2 explains how the algorithm is decomposed into 2 1D components and shows how this decomposition solves the problems of computational complexity and same-order classification. Section 4.3 details why absolute and argument clusters are ideal features and details how they are obtained efficiently. Section 4.4 provides an overview of how the classifier structure was selected and optimised for this specific task. Finally, Section 4.5 integrates each component of the AMC and SNR system by describing the algorithm from input to output. This chapter will therefore provide an understanding of the algorithmic operation of the proposed system.

4.1 An Introduction to DBSCAN

DBSCAN is a clustering algorithm which was first introduced in 1996 by M. Ester et al. [34]. As the name suggests, it is a density-based clustering algorithm, grouping points based on their proximity and density. Crucially for this work, it is non-parametric; that is, it requires no assumptions about the dataset, unlike most popular clustering algorithms such as K-Nearest Neighbours (KNN) and K-Means [53].

Although DBSCAN is non-parametric in the sense that the number of expected clusters is not required to be specified, 2 hyperparameters are required to be set before operation can begin. These hyperparameters are traditionally known as ε and *minPts*. The hyperparameter ε sets the maximum radius around a point for other points to be considered part of the same cluster, *minPts* sets the minimum number of local points which must be found within an ε

neighbourhood before cluster formation may begin.

The operation of the DBSCAN algorithm on BPSK data at an SNR of 15dB is shown in Figure 4.1. A point P is randomly selected and a circle of radius ε is drawn around this point (subplot 1). Points within this radius are labelled core points if there is at least *minPts* points If at least *minPts* core points are found, cluster formation begins. Circles of radius ε are iteratively drawn around discovered core points, again if there are at least *minPts* points within a neighbourhood the points are labelled as core points, points without *minPts* within their neighbourhood are labelled as border points. Subplot 2 shows the resulting neighbourhood circles from cluster formation. The resulting cluster from this process is labelled in blue in subplot 3. Once all points of a cluster have been found the algorithm randomly selects another point in the dataset, as shown by the pink circle, the same process iterates until all points within the second cluster have been found (subplot 5). The only points which remain are not dense enough to constitute the formation of another cluster as they do not have *minPts* points within their ε neighbourhood, nor are they close enough to a cluster to be a border point; thus, they are labelled as noise as shown in 6th subplot of Figure 4.1.

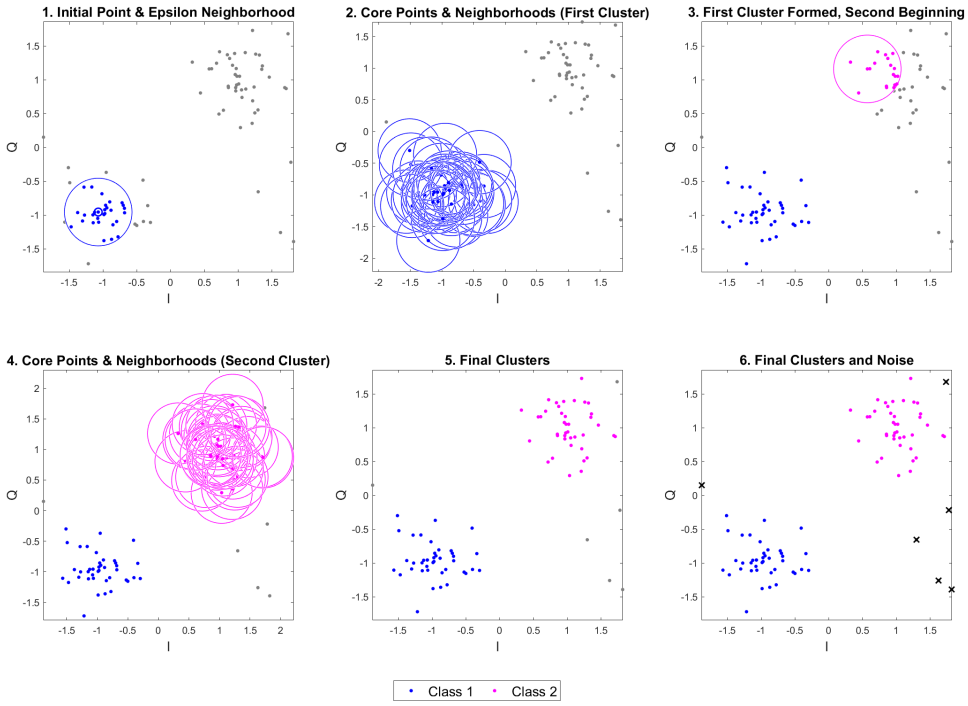


Figure 4.1: *The Operation of the Traditional DBSCAN Algorithm*

4.1.1 Generality Limitations of DBSCAN for Modulation Classification

As discussed in the literature review, previous attempts to apply DBSCAN for the purposes of AMC feature extraction have relied solely upon determining the number of constellations on the constellation diagram. This method was shown to be effective but suffered from the major limitation of being unable to classify differing modulation schemes of equivalent order.

Figure 4.2 (a) and (b) respectively show the resulting clusters when DBSCAN is applied to the 16QAM and 16PSK constellation diagrams at an SNR of 20dB.

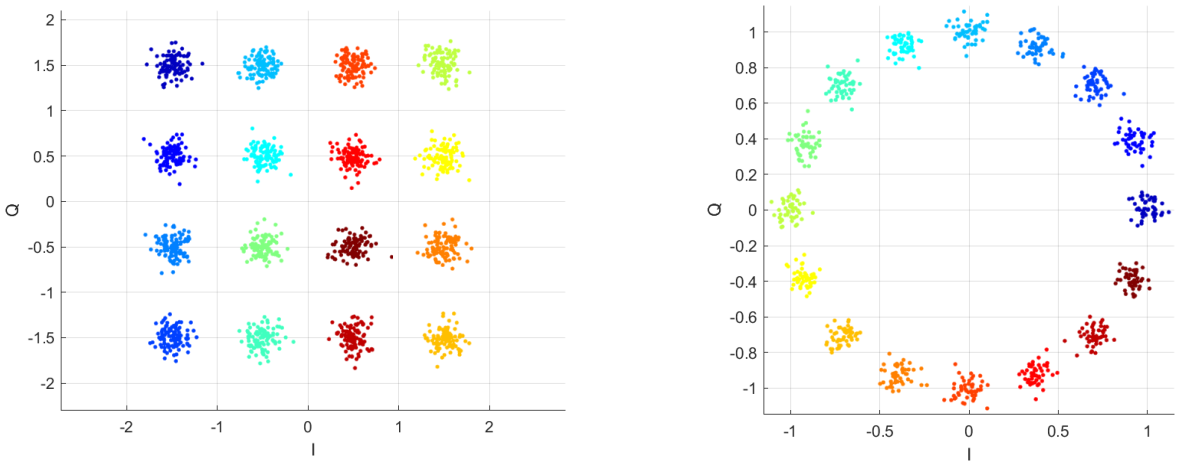


Figure 4.2: *Clusters Identified by the 2D DBSCAN Feature Extractor on (a) 16QAM and (b) 16PSK*

In both cases shown in Figure 4.2 16 constellation points are found, there is no information gained which can allow for differentiation between the 2 modulation schemes. It is therefore advantageous to develop a method of obtaining a measure of constellation point positioning in addition to the number constellations themselves. It can be said that this method of using DBSCAN for AMC lacks generality for this reason.

4.1.2 Algorithmic Limitations of DBSCAN Feature Extraction

When the DBSCAN algorithm executes it does so by iteratively calling a *rangeQuery* function. The *rangeQuery* function finds the Euclidean distance from a point P to every other point in the dataset, *rangeQuery* must be called for every point in the dataset. This means that this implementation of DBSCAN has a worst-case time complexity of $O(n^2)$ as each *rangeQuery* execution has a worst-case complexity of $O(n)$ and must run n times. By utilising an indexing structure, a database structure which uses a spatial index to store information about the spatial relationship of datapoints, it is possible to reduce the number of computations required for each *rangeQuery* by only computing distances to points which are known to be local to a point P. This reduces the worst-case computational complexity of *rangeQuery* to $O(\log(n))$ which leads to an overall complexity of $O(n \log(n))$. However, E. Shubert et al. [35] prove that there can be no indexing structure which provides a worst-case computational complexity of $O(\log(n))$ for every dataset. In addition, building an indexing structure for an FPGA implementation increases the implementation size and introduces an additional preprocessing step which will cause delay in a real-time system such as in the proposed work of this thesis. A final disadvantage of using an indexing structure is that $O(n^2)$ memory is required, instead of $O(n)$, which further increases the amount of FPGA resources needed.

In addition to the complexity introduced by the requirement for calling the *rangeQuery* function over many iterations, the calculations required to find the Euclidean distance in 2D

space must be performed with the equation shown in Equation 4.1.

Given 2 points $P(x_1, y_1)$ and $Q(x_2, y_2)$ The Euclidean distance is:

$$\text{Distance} = \sqrt{((x_2 - x_1)^2 + (y_2 - y_1)^2)} \tag{4.1}$$

This formula requires the execution of 2 square operations and a square root. The square root operation is particularly computationally expensive to perform in hardware, with the Coordinate Rotation Digital Computer (CORDIC) algorithm being the generally preferred method of implementation [88]. This method is iterative and thus would require multiple clock cycles to execute, if this was required to be performed to find the distance between every point in a given dataset, either the required latency or implementation size if parallelised would be large.

4.1.3 DBSCAN Extraneous Operations

There are several operations within the DBSCAN algorithm which are extraneous and not required to be performed for the purposes of AMC. The first operation is the labelling of points as part of a particular cluster. Traditional DBSCAN implementations track which cluster each point in a dataset belongs to once it has been found, in a hardware implementation this introduces the requirement for additional storage to hold this information. The purpose of using DBSCAN for AMC is to find the number of constellations, or numbers of different magnitudes and arguments as is the case in the proposed system in this thesis. Therefore, the cluster to which each point belongs is irrelevant, the only information which is required is the total number of clusters which have been found. The second feature is the distinction between border points and core points, as with tracking the cluster to which each point belongs, making this distinction imposes an additional requirement for memory to store this information as well as logic to determine when the distinction should be made. Again, as the only metric which is required to achieve AMC is the total number of clusters, this information is not required to be determined or stored.

4.2 The DBSCAN 1D Decomposition

The 3 major limitations of DBSCAN for AMC in a hardware context have now been outlined, the first is the lack of ability to distinguish between differing modulation schemes of the same order, the second is the computational complexity of the algorithm itself, and the final is the requirement to label points as belonging to a cluster and as a border or core point. This thesis proposes that all stated problems may be solved with by decomposing the 2D constellation diagram into 2 datasets consisting of the component arguments and magnitudes of each point, sorting each dataset, and designing an algorithm to accommodate these changes and optimise for implementation in hardware.

4.2.1 Solving the Generality Problem with Magnitudes and Arguments

It was shown how past attempts to apply DBSCAN to modulation scheme feature extraction could not distinguish different modulation schemes of equivalent order due to the algorithm only being able to find the number of constellation points. Therefore, a means of determining

the positioning of constellation points is also required to be found. This may be done by executing DBSCAN on the values of the magnitudes and arguments of each constellation point. Figure 4.3 shows the information which is obtained with this method.

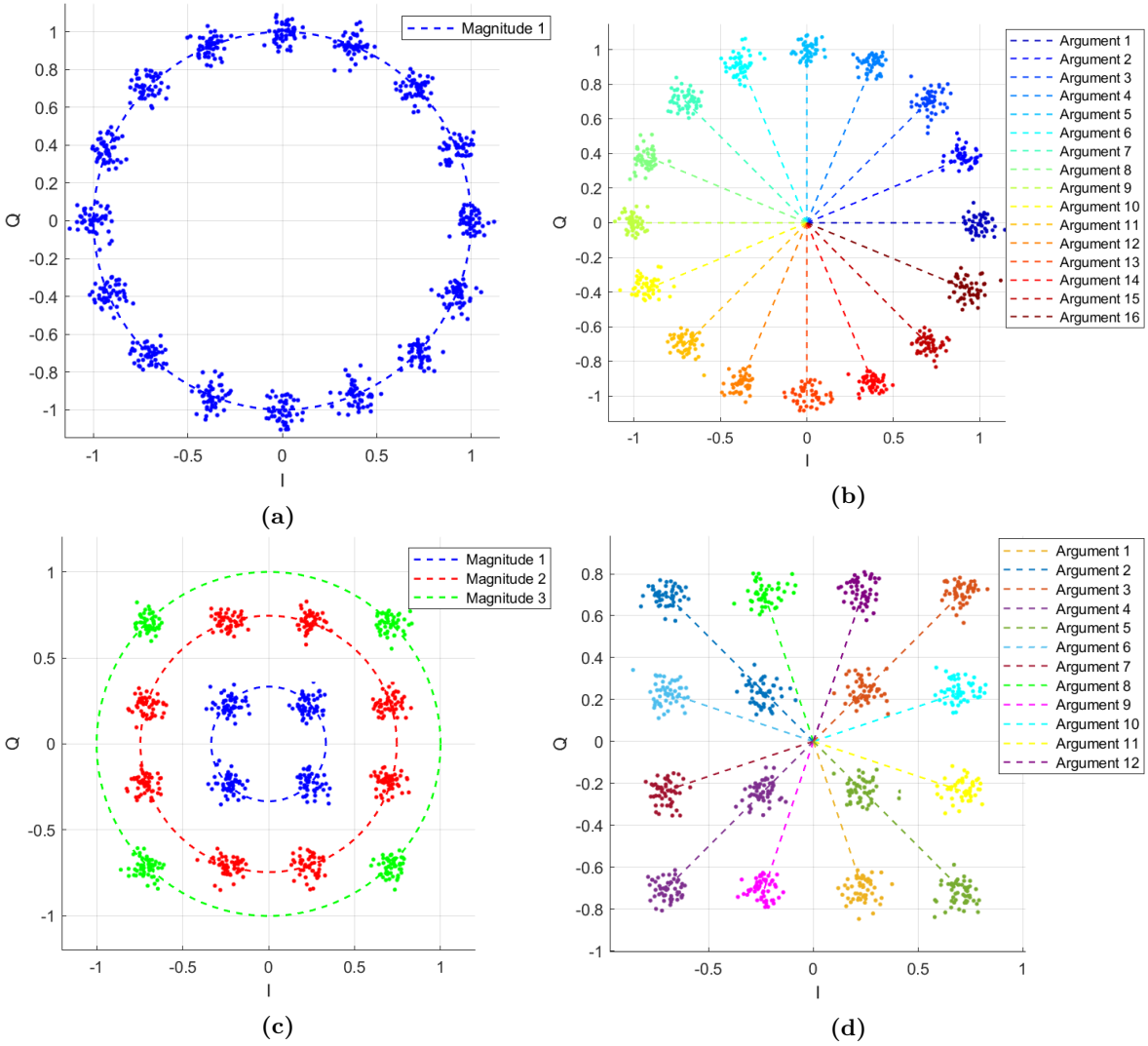


Figure 4.3: *Clusters Identified by the 1D DBSCAN Feature Extractor on (a) 16PSK Magnitudes, (b) 16PSK Arguments, (c) 16QAM Magnitudes, and (d) 16QAM Arguments*

Figure 4.3 shows the constellation diagram of 16PSK and 16QAM at 20dB SNR, for each constellation diagram the DBSCAN algorithm is applied to 2 individual datasets each containing the absolute and argument values for the respective constellation diagrams. 16PSK is shown in Figure 4.3 (a) and (b), the constellation points all lie equidistant from the origin, therefore the clustering algorithm determines that this constellation diagram has a single magnitude. 16PSK is also found to be formed using 16 different arguments. Similarly, 16QAM is found to consist of 3 different magnitudes, as shown by the 3 different rings, and 12 different arguments as in 4 cases there are pairs of constellations which have the same argument but

differing magnitudes. Applying DBSCAN to the dataset in this fashion therefore results in features which may be used to distinguish 2 modulation schemes with equivalent order.

4.2.2 Reducing Algorithmic Complexity with 1D Datasets

The decomposition of the constellation diagram data to 2 datasets consisting of magnitude and argument data necessitates DBSCAN to be executed on each dataset individually. Despite doubling the number of times DBSCAN is required to be executed, reducing the dimensionality of the 2 datasets enables the computational complexity of DBSCAN to be improved. Firstly, it was discussed that using an indexing structured database for 2D data can facilitate a computational complexity reduction of DBSCAN from $O(n^2)$ to $O(nlog(n))$ but requires an increase in memory and logic to implement this change. Data which is unidimensional may be sorted and stored in the sorted configuration. This eliminates the need for complex database indexing and the additional memory requirements, although the requirement for a sorting algorithm is introduced.

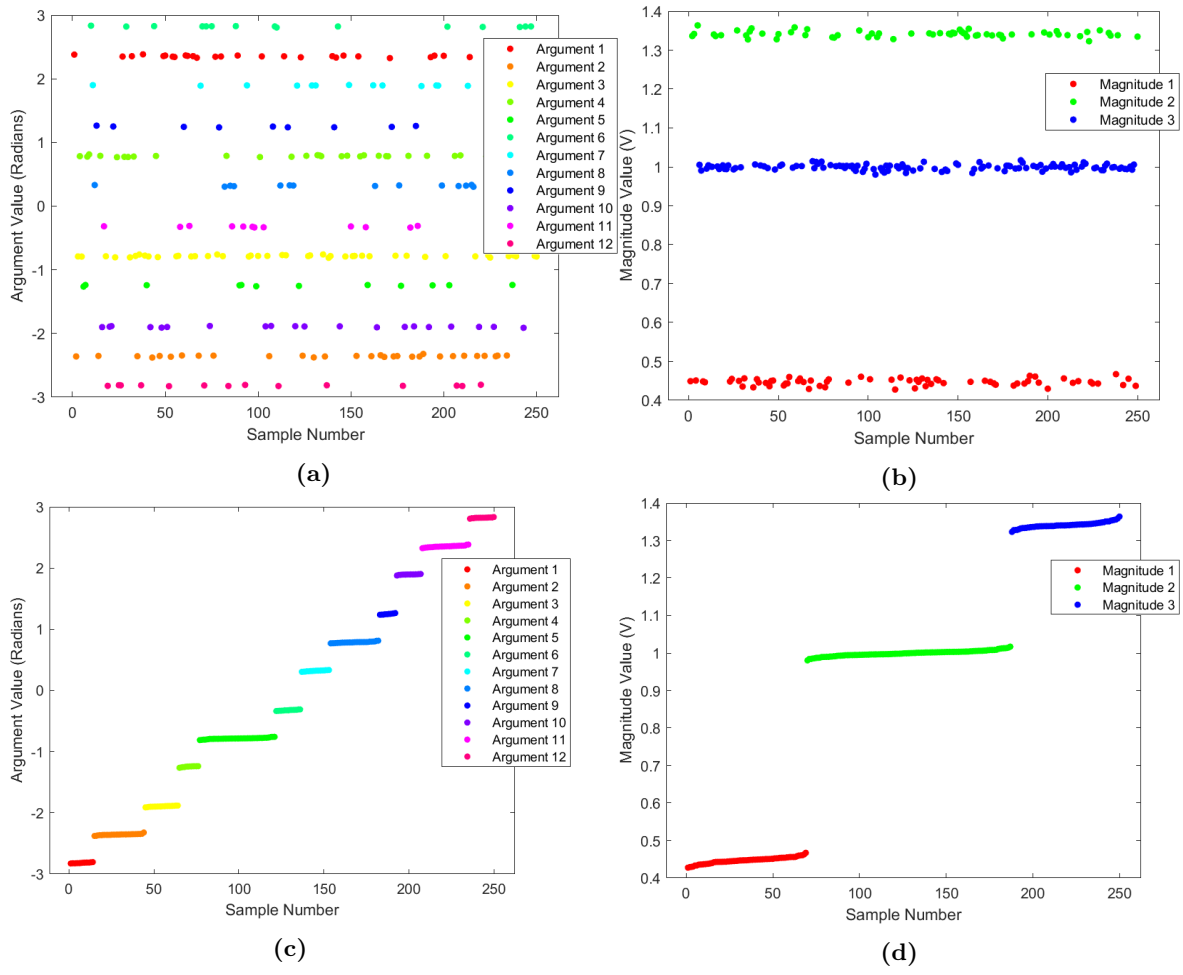


Figure 4.4: *Clusters Identified by the 1D DBSCAN Feature Extractor on (a) 16QAM Unsorted Argument Data, (b) 16QAM Unsorted Magnitude Data, (c) 16QAM Sorted Argument Data, and (d) 16QAM Sorted Magnitude Data*

Sorted data provides a twofold improvement in terms of computational complexity, firstly the requirement to execute a *rangeQuery* function for each datapoint to every datapoint within a dataset is eliminated; to find local points to a point P the algorithm need only check the distance to points which are local to the point P in the sorted dataset. The simplest implementation of this modified algorithm only checks the distance between a point P_n and the subsequent point $P_{(n+1)}$. Secondly, the function to determine the distance between points is reduced to a subtraction as the dataset now only exists in a single dimension. The distance between 2 points x_2 and x_1 may be obtained with Equation 4.2.

$$\text{Distance} = x_2 - x_1 \tag{4.2}$$

Comparing with Equation 4.1, 2 square operations, an addition, a subtraction, and crucially a square root are no longer required to be executed. Figure 4.4 illustrates the elimination of the *rangeQuery* function visually, in (a) and (b) the 40dB SNR 16QAM magnitudes and arguments are not sorted, in each case the clustering algorithm must check the difference between every point within the dataset. In Figure 4.4 (c) and (d) the data is sorted from smallest to largest, the distance between points is implicit in the dataset structure, therefore only the distance to the next point in the dataset must be found, eliminating the need to perform $n - 1$ difference operations per datapoint.

4.2.3 Eliminating Extraneous Functionality

It was previously stated that the functionality to label datapoints as core and border points, as well as tracking the cluster to which each point belongs is functionality which is not required for the purposes of AMC in this context. The only metric which is required to be output is the total number of clusters found by the algorithm. For this reason, the modified algorithm is not implemented with such functionality.

4.2.4 The Modified DBSCAN Algorithm

The optimisations to the dataset and requirements for a modified DBSCAN algorithm have been outlined in prior sections. The modified DBSCAN algorithm is now required to operate with two assumptions:

- The input dataset will be sorted and arrive serially
- The output should be the number of clusters found within n samples.

In addition to these assumptions the algorithm is required to incorporate a method of using the ε and *minPts* hyperparameters whilst iteratively finding the distance between a point P_n and the subsequent point P_{n+1}. Using these stated assumptions and requirements the following algorithm was designed:

The proposed optimised DBSCAN algorithm begins with the first point of a dataset, the difference between the first and second datapoint is taken and compared with the value of ε, if the difference is less than ε a variable called *PointCount* is incremented. The algorithm then iteratively moves through the sorted dataset, taking the difference between the 2 adjacent datapoints and comparing with ε. When the algorithm obtains a difference which is greater than ε all points within a cluster have thus been found, the value of *PointCount* is then

compared with the *minPts* hyperparameter to check if the number of found points are enough to enable a cluster to form. If the value is greater than *minPts* a second variable named *ClusterCount* increments, if it is not then *ClusterCount* remains unchanged, in both cases *PointCount* is reset to 0. This iterative process continues until the last point of the dataset is reached, at which point *PointCount* is checked to see if it is greater than *minPts*, *ClusterCount* is then incremented accordingly. Following these steps the value of *ClusterCount* holds the number of clusters which have been found and is output. The algorithm can also be viewed in terms of pseudocode in Algorithm 1, the pseudocode assumes a 50-point input of QPSK magnitude data.

Algorithm 1 Algorithm for optimized 1D DBSCAN

$\varepsilon \leftarrow 8$, $minPts \leftarrow 3$
$ClusterCount \leftarrow 0$, $PointCount \leftarrow 0$
$Data[50] \leftarrow Input[50]$
for $i = 1$ to 49 **do**
  $Data[i] - Data[i+1] = Diff$
  if $Diff \leq \varepsilon$ **then**
    $PointCount++$
  else
    if $PointCount \geq minPts$ **then**
      $ClusterCount++$
      $PointCount \leftarrow 0$
    else
      $PointCount \leftarrow 0$
    end if
  end if
end for
if $PointCount \geq minPts$ **then**
  $ClusterCount++$
end if
$Output \leftarrow ClusterCount$

4.2.5 Total Computational Complexity Improvements

The original DBSCAN algorithm is said to have a worst-case computational complexity of $O(n^2)$, although the average run time complexity is $O(nlog(n))$ [34], owing the requirement to perform n *rangeQuery* operations for a maximum of n datapoints. The proposed modifications necessitate only n operations be performed, resulting in a worst-case computational complexity of $O(n)$. In addition to reducing the total computations by a factor of n, the individual computations themselves are optimised as 2 square operations, 2 additions, and a square root operation are no longer required to be performed.

Although the computational complexity is significantly enhanced, there is now a requirement to sort each 1D dataset which should increase computational complexity once more as even the quickest sorting algorithms have a worst-case computational complexity of $O(nlog(n))$ [89]. However, a hardware specific sorting algorithm was created which took

advantage of the real-time nature of the data input. The implementation of this sorting algorithm resulted in data being sorted with a computational complexity of $O(n)$, allowing for the improvements to DBSCAN to be maintained. The discussion of this algorithm may be found in Chapter 5 which covers the hardware implementation.

4.3 The Requirement for Magnitude and Argument Data

At the receiver the received waveform data is digitised as complex I/Q pairs, converting this rectangular data to the polar representation introduces additional calculations which must be performed. It was found that this is a necessary step as the number of I and Q clusters is not always consistent. The constellation diagram can appear to be rotated in certain situations, examples of effects which may induce this rotation are the Doppler effect, Carrier-Frequency Offset (CFO), or both simultaneously [10].

At the receiver, the Local Oscillator (LO) is tuned to the expected frequency of an incoming carrier wave; by multiplying the signal with the LO frequency the modulation information is extracted from the carrier [10, 90]. CFO is caused by the frequency of the LO in the receiver not being synchronised with the carrier frequency of the received signal. The cause of this desynchronisation can either be caused by a difference in frequency of the transmitter and receiver oscillator or by movement of the transmitter or receiver introducing a Doppler effect [10, 90].

The reason for this rotation may be explained via the mathematical form of a modulated signal. As shown in Chapter 2, a modulated signal may be expressed as with the form:

$$y(t) = A(t)(e^{j\phi(t)}e^{(j(2\pi f_c t))}) \tag{4.3}$$

Where f_c is the carrier frequency, $A(t)$ is the amplitude with respect to time, and $\phi(t)$ is the signals phase with respect to time. A received signal with a frequency offset may be expressed as:

$$y(t) = A(t)(e^{j\phi(t)}e^{(j(2\pi(f_c+\Delta f)t))}) \tag{4.4}$$

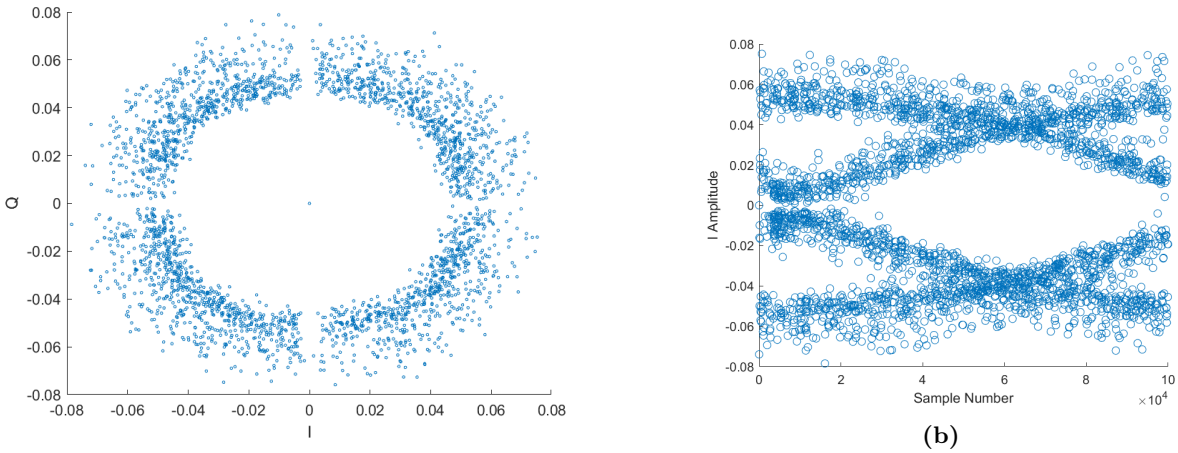
Where Δf is the difference between the LO frequency and the signal frequency. This equation may be rewritten as:

$$y(t) = A(t)(e^{j\phi(t)}e^{(j(2\pi f_c t))}e^{j2\pi\Delta f t}) \tag{4.5}$$

When the LO extracts the carrier frequency from the received waveform equation, the $e^{j2\pi\Delta f t}$ term remains:

$$y(t) = A(t)(e^{j\phi(t)}e^{j2\pi\Delta f t}) \tag{4.6}$$

What remains is the magnitude $A(t)$, the phase rotation $e^{j\phi(t)}$, and an addition rotation which is imposed by the remaining frequency component which has not been extracted by multiplication with the LO $e^{j2\pi\Delta f t}$ [10, 90, 91]. An example of a CFO induced rotated constellation diagram of 20dB SNR QPSK data, alongside the resulting changes in I values is shown in Figure 4.5.



(a) (b)

Figure 4.5: *The Effects of CFO on (a) the QPSK Constellation Diagram and (b) the I Values of QPSK*

Figure 4.5 (a) demonstrates a QPSK constellation which is subject to rotation induced by CFO. Rather than appearing as 4 well defined constellation points, the diagram is subject to a rotation about the origin. Figure 4.5 (b) shows that this rotation results in a continuous change in the number of different I values which are extracted from the signal, at any given time there may be 2, 3, or 4 different I values obtained from the I/Q data which forms the constellation diagram, the same is true for the Q values.
Figure 4.6 shows another QPSK constellation diagram at 30dB SNR rotated to various degrees, in this case the rotation is manually applied to illustrate why a CFO induced rotation results in changing I values. The red lines in Figure 4.6 intercept the midpoints of each constellation point. Depending upon the degree of rotation, various numbers of I values are obtained, as shown in figure 4.6 there may be 2, 3, or 4 different I values. Having a feature which is variable depending on CFO is not conducive to consistent classification performance as the classifier is trained by learning specific values of input features, features which vary over time may cause overlap between classes in the feature space and therefore reduce classification performance.

Instead, by utilising the magnitude and argument as the extracted features, robustness to these effects is introduced into the classification system. This is demonstrated in Figure 4.7 where the same QPSK rotations are shown, however in all cases a single magnitude and 4 arguments are consistently obtained. However, in scenarios where the dataset size is large enough to capture a consistent rotation effect, the argument data will be affected. In this case the argument data may appear as a single cluster, or no cluster may be formed. The magnitude data extraction remains unaffected and can still be used as a strong feature for AMC. Utilising the argument and magnitude therefore introduces partial robustness to a continuous constellation diagram rotation as the argument data is still affected, complete immunity to a static rotation is obtained as in this case the number of extracted arguments and magnitudes remains constant.

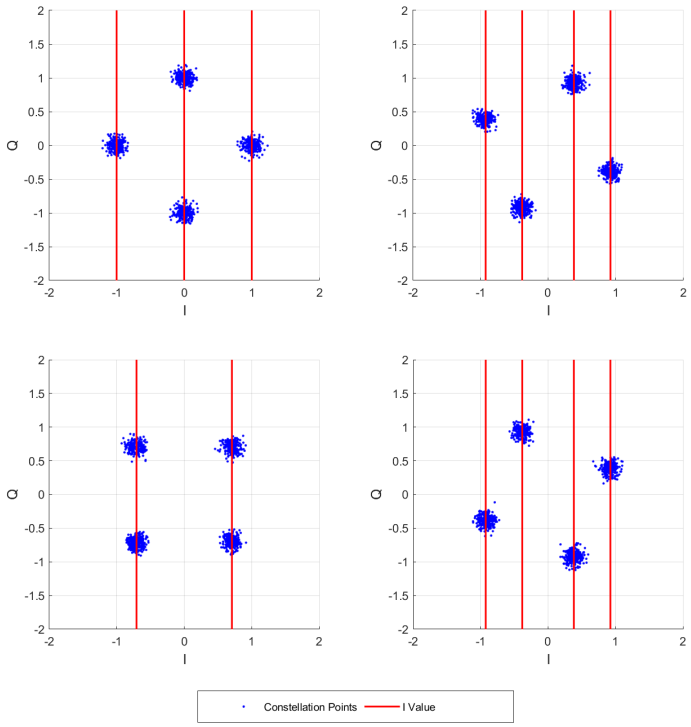


Figure 4.6: *How the Extracted I Values Differ Depending upon Rotation Imposed by CFO*

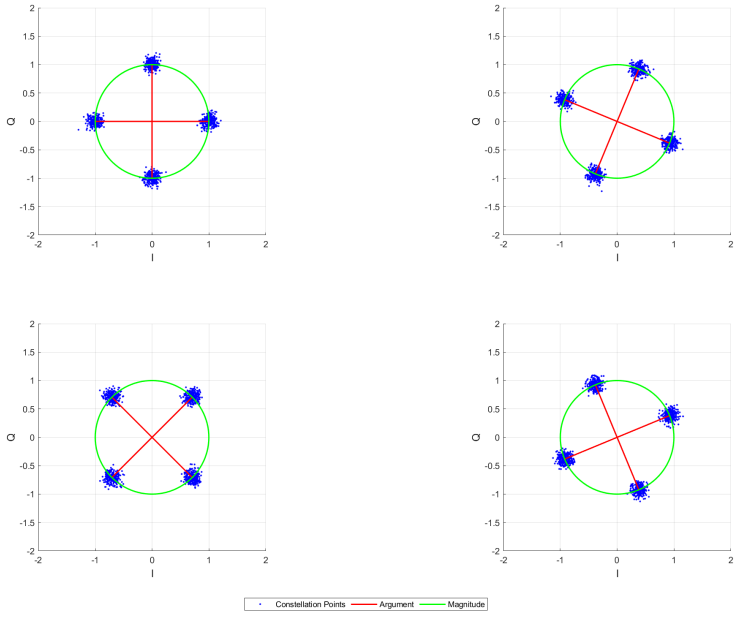


Figure 4.7: *How Utilising Argument and Magnitude Data Results in Partial CFO Robustness*

4.3.1 Calculating the Magnitude and Argument Values

By converting the I/Q data from rectangular to polar form, the constellations points can be represented in terms of an amplitude and an argument. This may be performed with the equations shown in Equations 4.7 and 4.8.

$$A = \sqrt{I^2 + Q^2} \tag{4.7}$$

$$\phi = \text{atan2}(Q, I) \tag{4.8}$$

Where A is the magnitude of the I/Q pair, ϕ is the argument. The square root term for calculating the absolute value and $atan2$ function for finding the argument are operations which are best avoided where possible due to the complexity of implementation and the length of execution in digital hardware. However, should an engineer wish to implement these functions in hardware, the most efficient method of implementing them is with the CORDIC algorithm [88]. This is an algorithm which decomposes a larger rotation into an iterative succession of smaller rotations performed by shift and add operations, which are far simpler for a digital computer to perform. Rather than implement the required operations to calculate the absolute value and argument individually, the CORDIC algorithm can perform the rectangular and polar conversion directly, which eliminates the need for a division operation and reduces the number of required CORDIC blocks in a hardware implementation from 2 to 1.

4.3.2 The CORDIC Algorithm

The mathematical operation of a CORDIC operation can be explained as follows: A 2-dimensional rotation is given by the equations:

$$\begin{aligned} x_{i+1} &= x_i \cos(\theta) - y_i \sin(\theta) \\ y_{i+1} &= y_i \cos(\theta) + x_i \sin(\theta) \end{aligned} \tag{4.9}$$

Using the identity:

$$\cos(\theta) = \frac{1}{\sqrt{1 + \tan^2(\theta)}} \tag{4.10}$$

The rotations can be substituted to become:

$$\begin{aligned} x_{i+1} &= \frac{(x_i - y_i \tan(\theta))}{\sqrt{1 + \tan^2(\theta)}} \\ y_{i+1} &= \frac{(y_i + x_i \tan(\theta))}{\sqrt{1 + \tan^2(\theta)}} \end{aligned} \tag{4.11}$$

The rotation is now in the form of an addition or subtraction of 2 coordinates and 2 multiplications with $\tan(\theta)$ as well as a scaling term $\frac{1}{\sqrt{1+\tan^2(\theta)}}$ [88, 92, 93]. The scaling term is cancelled by multiplying both sides by $\sqrt{1 + \tan^2(\theta)}$, this leads to the rotation producing a larger magnitude than a true rotation would produce, but this can be compensated for after the algorithm is complete [88,92]. Recall that CORDIC implements a complete rotation with

a sequence of iterative rotations. By using this iterative rotation process, this multiplication by tan can be performed with a shift operation by only using values of $\tan(\theta)$ equivalent to negative powers of 2 [88]. A table of these angles in degrees can be seen in Table 4.1.

i	Rotation (°)
0	45
1	26.57
2	14.04
3	7.13
4	3.56
5	1.79

Table 4.1: *The Rotation Angles Achieved by the Iterative CORDIC Shift Operations*

By iteratively performing positive or negative rotations by these set angles, any rotation between $\pm 99.7°$ can be approximated. Each individual step (excluding the scaling term) can now be expressed as:

$$\begin{aligned} x_{i+1} &= x_i - (y_i \gg i) \\ y_{i+1} &= y_i + (x_i \gg i) \end{aligned} \tag{4.12}$$

Figure 4.8 shows how an angle θ can be expressed with iterative rotations of a_i.

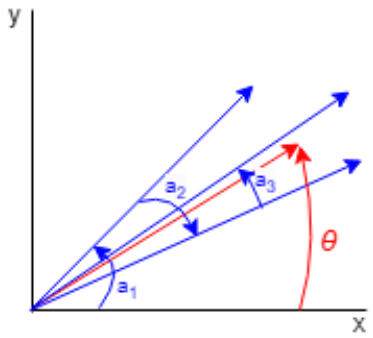


Figure 4.8: *How Iterative Rotations may Approximate an Angle with the CORDIC Algorithm*

With each successive rotation the resulting angle gets closer to the true angle of rotation, in order to achieve k bits of precision, k iterations are required because $\tan^{-1}(2^{-i}) \lesssim 2^{-i}$ converges as i increases [92]. The magnitude of the rotation increases with each rotation, this can be accounted for after the rotation process has completed. As there is a known set of angles comprising the total rotation, the scaling factor $\sqrt{1+\tan^2(\theta)}$ can be precomputed and applied to the result at the output to find the correct magnitude, the amount of scaling K can be found with Equation 4.13 [92]:

$$K = \prod_{i=0}^{n-1} \sqrt{1+\tan^2(a_i)} \tag{4.13}$$

The value asymptotes 1.646760258 as the number of iterations increases [88].

4.3.3 Using CORDIC to Obtain the Polar Form Directly

In the case of this work the CORDIC algorithm was utilised to convert complex numbers in the rectangular form to the polar representation, a detailed explanation of how this process can be implemented may be found at [93]. Before the CORDIC algorithm begins the I/Q pair must be rotated to be within 45° of the positive x-axis, this is because the CORDIC algorithm can only approximate angles between ±99.7° as previously stated. By only rotating to ±45° the first step of the CORDIC algorithm, which a rotation by 45°, may be skipped which reduces compute time. To determine the pre-rotation angle, the sign bits of the I and Q value is checked to find the quadrant which the I/Q pair lies in. A clockwise pre-rotation is then applied the I/Q pair to ensure that the I/Q vector lies within the required ±45° for the CORDIC algorithm to operate. The designated pre-rotation angles by quadrant can be seen in Figure 4.9. The pre-rotation is performed by an addition of the I and Q values, negated values are used where required. A table of operations for the pre-rotation can be seen in Table 4.2.

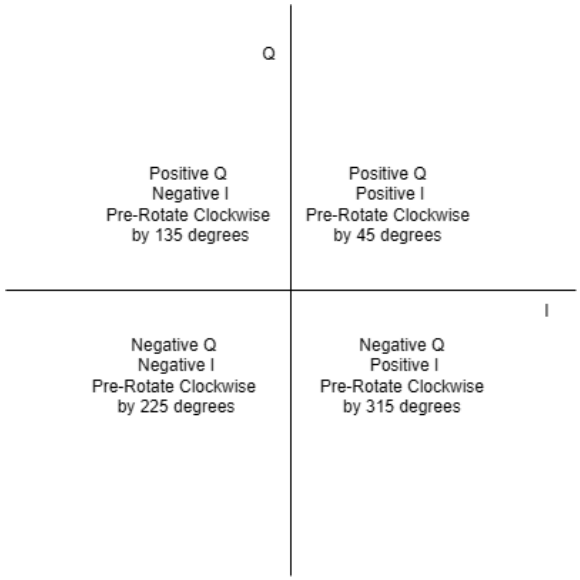


Figure 4.9: *The Pre-Rotation Angles Applied According to the Quadrant in which an Angle Belongs*

Rotation	New I Value Formula	New Q Value Formula
-315	I-Q	I+Q
-225	-I-Q	I-Q
-135	-I+Q	-I-Q
-45	I+Q	-I+Q

Table 4.2: *Rectangular to Polar CORDIC Pre-Rotations*

This pre-rotation angle value is stored and added to the argument found during the CORDIC operation. The angle is now within the acceptable bounds for the CORDIC algorithm to begin, the target angle is assumed to be 0° and operation begins. With each iteration the y value is checked for being positive or negative, if the y value is negative a positive rotation is applied and vice versa. An additional sum of each rotation angle θ is kept. Once the CORDIC iterations have finished the y value should be 0 meaning that the vector has rotated to 0°, the x value represents the magnitude of the input I/Q pair multiplied by scaling factor K, and the sum of each rotation performed is the argument of the I/Q pair minus the rotation applied before the CORDIC began. Thus, by scaling x by K with a multiplier or a LUT and re-adding the pre-rotation value to θ, the magnitude and argument of an I/Q pair can be determined.

4.4 The Choice of Classifier

Every component of the proposed system structure has now been outlined other than the classifier itself. As the algorithms which operate before the classifier in the overall system operate as a feature extraction mechanism, any model which performs its own feature generation is not a viable candidate classifier for this system, furthermore, these classifier structures have high complexity and therefore high implementation sizes as demonstrated in Chapter 3. These factors rule out the usage of deep learning networks such as LSTMs, CNNs, and Transformers.

There are several requirements which the selected classifier should fulfil. Firstly, the classifier structure should be the simplest implementation which can maintain the maximum level of performance, this enables a small implementation size without a sacrifice in overall classification accuracy. Secondly, the classifier should be highly reconfigurable as it is required to be modified to suit specific SNRs as well as be reconfigured for SNR estimation itself. Finally, it should have an algorithm structure which is conducive to hardware implementation.

4.4.1 Evaluating Classifier Performance

Various classifier structures were first evaluated to explore which models performed well on the dataset. Firstly, the performance of linear and non-linear models was compared. It was stated in Chapter 3 that performance differentials between linear and non-linear models are largely due to the arrangement of the classes in the feature space. For the data employed in this thesis it was found that for the most part the classes showed strong separation and were linearly separable, particularly when the features were extracted from high SNR data.

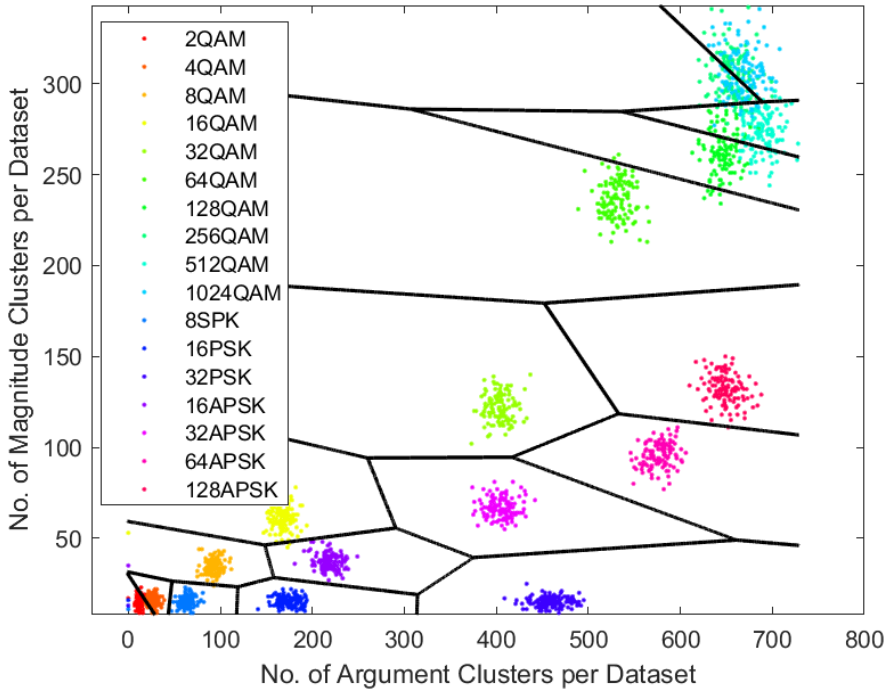


Figure 4.10: *The Decision Boundaries Learned by the Linear SVM on Magnitude and Argument Data Extracted from Signals with an SNR of 30dB*

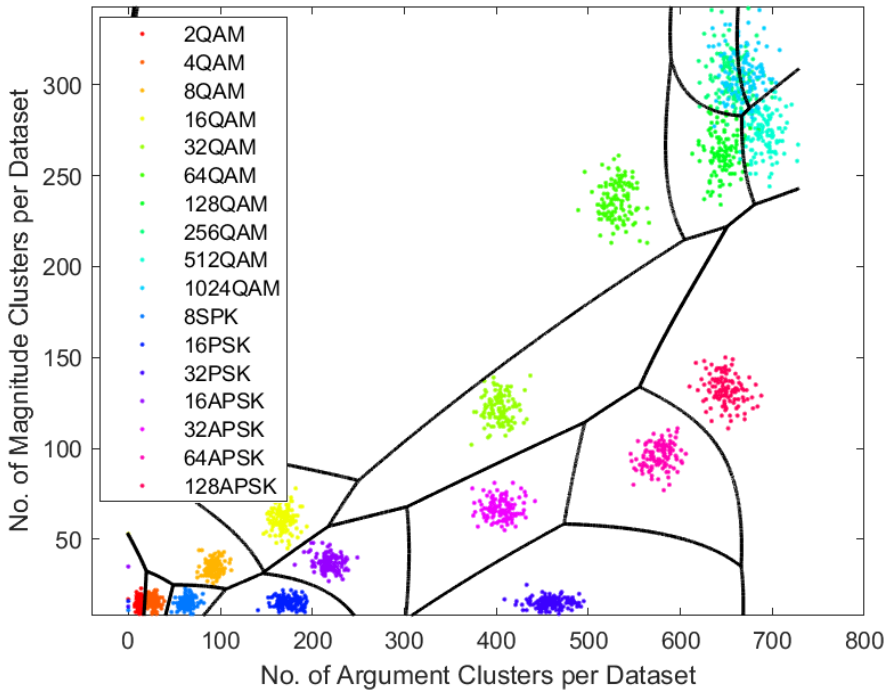


Figure 4.11: *The Decision Boundaries Learned by the Non-Linear SVM on Magnitude and Argument Data Extracted from Signals with an SNR of 30dB*

MATLAB R2021b [94] was used to train 2 SVMs , 1 linear and 1 non-linear, the dataset consisted of the number of argument and magnitude clusters found on signals modulated with QAM orders 2-1024, PSK orders 8 to 32, as well as APSK of orders 16 to 128, the SNR of the data was 30dB. The learned decision boundaries in the feature space can be seen in Figures 4.10 and 4.11.

In each of the provided figures the coloured clusters represent each class, the black lines denote decision boundaries learned by each classifier. It may be seen that the high-order QAM feature clusters exhibit a small degree of feature cluster overlap but in general are separated. The linear SVM fails to accurately capture the distribution of high-order QAM feature clusters, with decision boundaries intersecting the midpoints of clusters in many cases. Conversely the non-linear SVM is better able to capture the relationship, with the majority of feature cluster points being within their respective decision boundaries. Using 5-fold cross validated training it was found that the linear and non-linear models each achieved 88.82% and 93.68% accuracy respectively, a difference of nearly 5%.

From the provided examples it is clear that a model which has the ability to find non-linear decision boundaries is advantageous as a higher classification accuracy may be achieved. This limits the set of classifiers which are capable of this functionality to MLPs, SVMs, and RFs.

4.4.2 Evaluating Classifier Reconfigurability and Hardware Implementation Sizes

The selected classifier structure should have the ability to be implemented in hardware while using minimal resources. Furthermore, the classifier is required to be reconfigured to suit differing functionalities such as switching between SNR estimation and AMC. Therefore, there should be as few FPGA resources as possible required to suit this reconfiguration requirement.

RF models do not follow a regular structure, they consist of multiple DTs with each tree consisting of nodes and leaf nodes each connected in a structure which is unique to each trained model [53]. Switching between various RF models cannot be performed by changing values, the entire structure of each tree and forest must be stored [95]. Furthermore, the irregular structure of each DT makes the traversal of an RF classifier difficult to parallelise [95]. Van Essen et al. [95] attempted to solve this problem by creating compact RF models with a maximum depth, however their implementation required multiple FPGAs to fully implement. To accommodate the ability to switch functionality multiple models must be implemented on 1 FPGA, as a single model cannot be implemented on a single FPGA, RFs are unviable for this use case.

SVMs were utilised to evaluate the performance difference between linear and non-linear model structures, therefore it was known that they perform well on the feature set. However, when considering this structure for FPGA implementation they were found to be suboptimal. This is due to the requirement to store the support vectors, kernel parameters, and alpha values [96]. This requirement has been shown to lead to large FPGA implementations, particularly due to the support vectors being a subset of the training dataset [96]. While this constraint may be mitigated with quantisation for a single model implementation [97], there is a requirement to store upwards of 12 different model configurations which would quickly lead to an explosion in FPGA memory usage.

The final candidate model structure to evaluate is the MLP. This model is well suited

for FPGA implementation as only multiplication and add operations are required to achieve classification [53, 54]. MLP model structures can be predetermined and trained to suit the desired structure [53]. Furthermore, there is no requirement to store any training data alongside the implemented model, weight and bias values are all that is required to be stored alongside the logical structure of the model [98]. Both stated MLP features are highly advantageous in the case of the proposed system as only 1 logical structure is required to be implemented and to reconfigure the model for a different function all that is required is an update to the weight and bias values. Furthermore, said values can be quantised to reduce the total storage capacity of the implementation [98]. Finally, and perhaps most critically is the ability of the MLP to perform both classification and regression functions with a simple modification of weights. This enables the SNR estimation operation to take advantage of a regression model, enabling the ability to estimate the SNR of signals which may be of an SNR between the examples which were provided in the training data, rather than estimating classes which correspond to exact SNR values, thereby increasing flexibility by learning the relationship between feature space positioning and SNR.

The MLP is the ideal model structure for the proposed system due to the ability to learn non-linear decision boundaries [53, 54], a regular structure across various use cases, ease of reconfiguration, minimal implementation requirements to enable reconfiguration [99], and the ability to perform both classification and regression by modifying only weight values.

4.4.3 MLP Structure and Training Process

The structure of the implemented MLP can be found in Figure 5.8. The choice of the number of input and output nodes is tied to the number of utilised features and classes [53]. The proposed work only uses the number of magnitude and argument clusters as features, therefore only 2 input nodes are required. As will be discussed later in this thesis, varying numbers of classes may be used in testing, the lower limit of this number is 4 for the cases where only BPSK, QPSK, 8QAM, and 16QAM are to be classified, but the number of classes may grow as high as 17 when all signals are included within the applied dataset. The number of output nodes therefore varies between 4 and 17 but could be increased if necessary.

The only structural decision not tied to the number of used features and classes is the number of hidden nodes. As the proposed work attempts to minimise the size of the hardware implementation it is imperative that this value is set to the minimum possible value which achieves the highest obtainable level of performance. Testing to obtain this value was performed using MATLAB R2021b [94], the entire dataset of signals was utilised therefore 17 output nodes were required. This is the most complex configuration which the classifier would be expected to operate with, therefore finding a strong number of hidden nodes in this configuration would guarantee strong performance across all configurations. The dataset was split with a ratio of 80:20 and 5-fold cross validated training was performed for each test. The results from this testing process are shown in Figure 4.13. The only configuration where a noticeable reduction in classification accuracy was observed was when a single hidden node was utilised, otherwise the model performed similarly despite the number of nodes. On some occasions there were anomalous reductions in accuracy, but this phenomenon occurred throughout the range of utilised hidden nodes. Although 2 hidden nodes was found to be sufficient in tests, it was decided to use 3 hidden nodes in the final deployment as this provided a buffer of an extra hidden node to mitigate any anomalous behaviour while not incurring a

large utilisation increase.

Quantisation was also performed to minimise the implementation cost of the logic and datapath, as well as reduce the memory required to store weights and biases of the various MLP configurations. Training of the implemented models was performed using PyTorch 2.0 [100], rather than MATLAB due to the nn.linear [101] function providing simpler means of reporting the obtained weight and bias values. All training was performed using either the Levenberg-Marquardt (LM) algorithm [102] or Adam algorithm [103]. LM is known for its fast convergence and robustness but has a relatively high computational cost [104]. Given the small size of the model the computational cost was found to be manageable. Adam is an optimisation method similar to stochastic gradient descent but makes use of momentum, which has been found to assist with the the avoidance of local minima in the loss landscape [103].

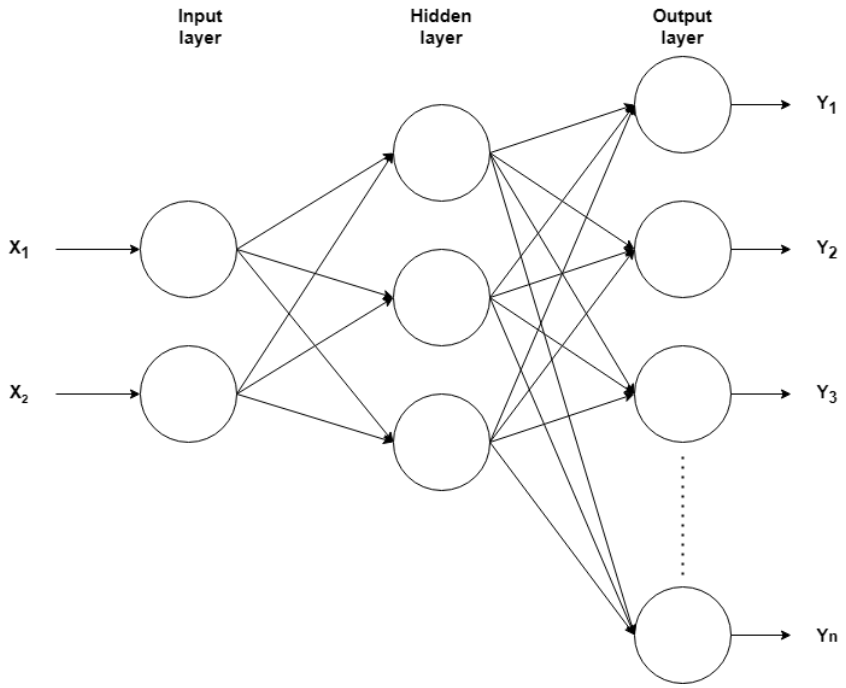


Figure 4.12: *The MLP Structure Employed within the Proposed System*

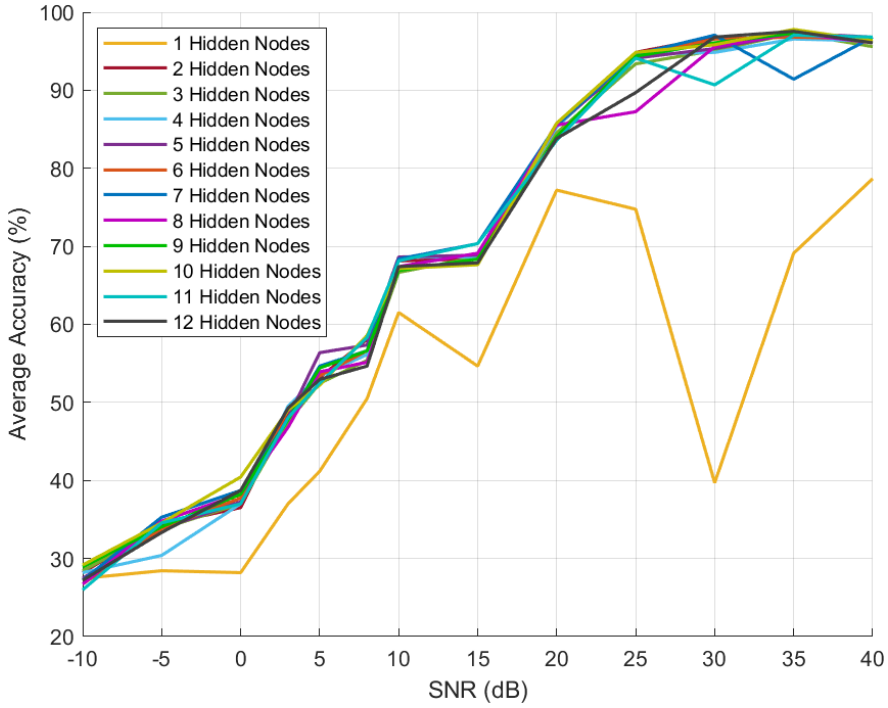


Figure 4.13: *How the Number of Hidden Nodes Affected the Average Accuracy (%) of the AMC System*

4.5 Overall System Structure

The methodology behind the creation of each constituent part of the DBSCAN AMC system has been thoroughly explored throughout this chapter. This final section will provide an overview of how each component operates as part of a complete algorithm.

The system may be split into 2 functional components, the DBSCAN based feature extraction mechanism and the MLP classifier which classifies the extracted features. The system structure is designed to be pipelined and operate on a real-time data stream, pipelining and real-time functionality will be described in greater depth in the following section which outlines the hardware implementation. For now, the algorithmic functionality is the principal focus. The step-by-step process is listed below:

1. Signal data in its I/Q representation is input to the system.
2. A CORDIC block converts the I/Q data to the polar form (magnitude and argument form).
3. The datapath is split in 2, with each data path operating on either the magnitude or argument data.
4. Each datapath sorts either the magnitude or argument data from smallest to largest.
5. Once n points have been sorted, they are input into the modified DBSCAN algorithm.

6. 2 structurally identical modified DBSCAN algorithms respectively find the number of argument and magnitude clusters.

7. The datapaths recombine, the number of argument and magnitude clusters are used as features for classification with the MLP.

8. The MLP outputs a classification result of which modulation scheme the input signal is most likely to be.

This process can be seen visually in figure 4.14, particular focus is given to how the dataset changes in each stage of the algorithm.

Following this explanation of the system structure, the mechanism of how this AMC system operates has been fully outlined in an algorithmic context. The following chapter details how the algorithm can be efficiently implemented on an FPGA, the implementation is fully pipelined to minimise latency and make the design suitable for operation with a real-time input. There are also 3 reconfigurable parameters which have not been discussed in this chapter, these are the size of the dataset which is used to generate features n, as well as the 2 DBSCAN hyperparameters ε and *minPts*. All 3 of these parameters are intrinsically linked to each other and the overall performance of the system. The optimisation of these parameters is complex and new methods have been developed to ensure that optimal parameter values are selected. Chapter 6 is entirely dedicated to the optimisation methods of these hyperparameters.

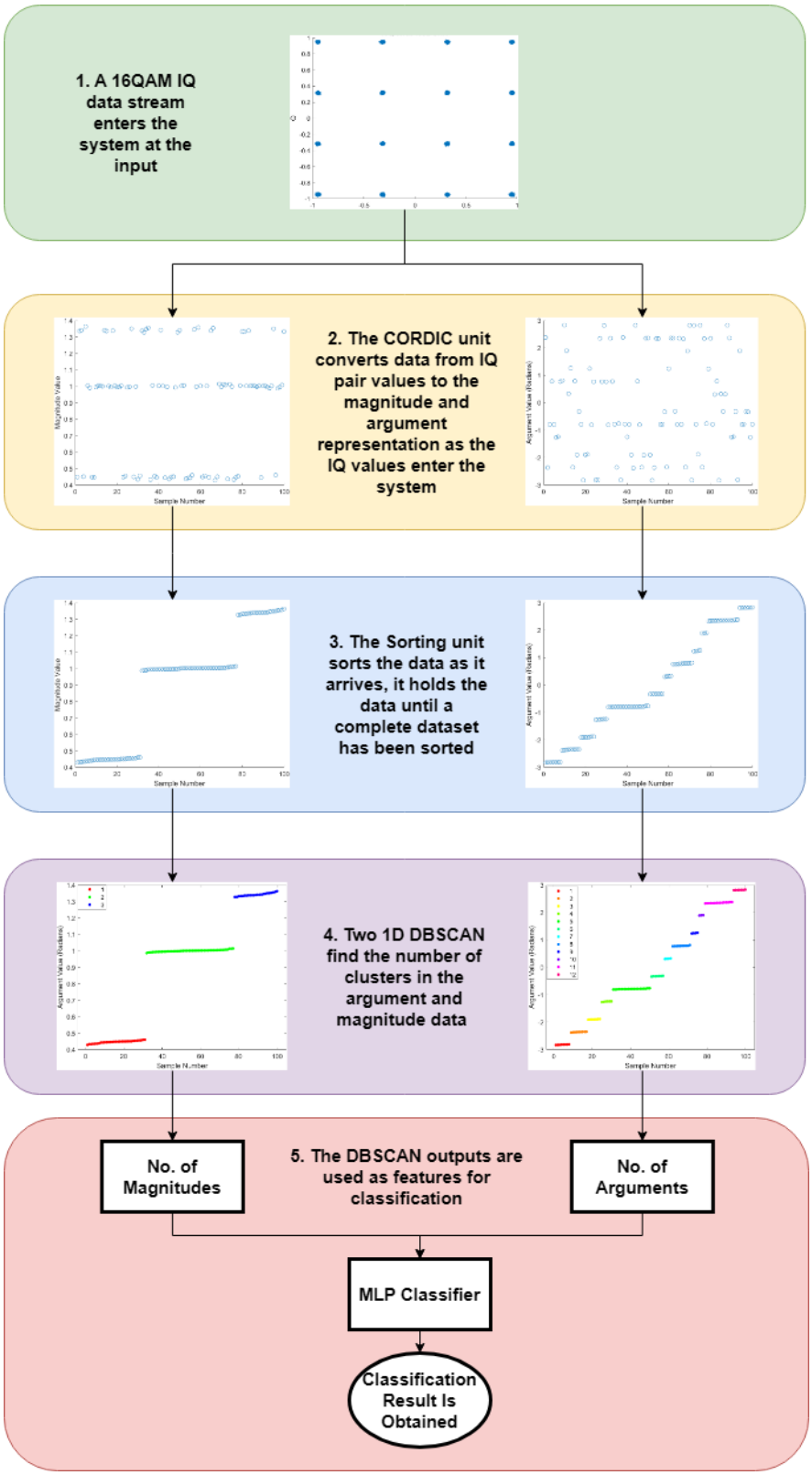


Figure 4.14: *How the Proposed System Manipulates a 16QAM Input Constellation Diagram to Achieve Modulation Classification*

Chapter 5

The DBMC/DBSNR Hardware Implementation

The prior chapter has discussed the theoretical underpinning of the proposed modulation classification and SNR estimation algorithm, this chapter details the methodology behind the creation of the hardware implementation. This includes a structural overview of the complete system as well as the code and settings required to realise the achieved implementation. A GitHub repository is provided which includes the Verilog sources so the reader may examine the implementation in more detail, this is available at https://github.com/billjgavin/DBMC-DBSNR.

The creation of this system was completed using Vivado 2021.2 [105], the code for the hardware was written in Verilog [106]. The target hardware platform was the Zedboard [107] which features the Xilinx Zynq XC7Z020-CLG484 FPGA [108]. This target platform was chosen due to providing a large enough array of programmable logic to enable the implementation of a system which matches the utilisation of the smallest state-of-the-art modulation classification systems as well as being available through departmental infrastructure.

This chapter is structured as follows: An introduction to the system datapath is provided alongside a discussion of the system's structure, latency, and control mechanism. Then an in depth exploration of each component module is presented and discussed, with design decisions explained and utilisation statistics provided. Following this, a detailed overview of the methods in which the system was created, tested, and verified is given including a description of the methods which were employed to use the system to obtain the results which are provided in Chapters 7 and 8. Finally a comparison between the proposed implementation and those found in the literature is made and discussed. Following this chapter the reader should have a thorough understanding of the structure and operation of the proposed system as well as the methods with which it was verified and tested.

5.1 System Overview and Structure

The aim of the work proposed in this thesis is to achieve the creation of a low-power, low-area, and low-latency modulation classifier and SNR estimator which also achieves competitive accuracy in comparison to works found in the literature. Another principal requirement of the proposed system was to facilitate operation on a stream of data with as small a latency

as possible, thereby enabling rapid feedback and continuous monitoring in a CR scenario.

5.1.1 Full System Datapath

The datapath of the system can be seen in Figure 5.1.

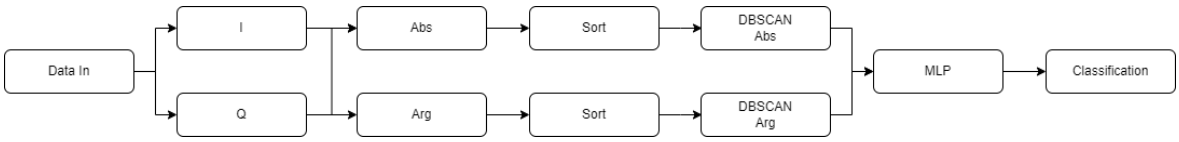


Figure 5.1: *The DBMC/DBSNR Full System Diagram*

The system datapath may be seen as having 3 functional blocks, the first is a CORDIC module which performs the required rectangular to polar conversion as discussed in Chapter 4. The second is the sorting algorithm and modified 1D DBSCAN, this block could be seen as two disparate elements but their implementation and operation is intricately designed to support each other. The final functional block is the MLP which performs either regression or classification on the outputs of the DBSCAN algorithm. Not pictured in the system diagram is the control logic, the control module lies within the sorting algorithm logic which was found to be a requirement to eliminate synchronisation errors, this design decision will be expanded upon in Section 5.3.

An overview of the path data takes through the system as well as the representation is as follows: Pairs of 14-bit I and Q values with a Q4.10 fixed-point representation are input to the system every clock cycle. They enter the CORDIC functional block which utilises a 15-bit datapath with a Q5.10 representation to account for overflow, the exception being the argument calculations which are performed with 19-bit Q10.9 fixed-point precision. The CORDIC module has a latency of 14 clock cycles and outputs a pair of magnitude and argument datums every cycle owing to the pipelined implementation. Both CORDIC outputs are truncated to unsigned 10-bit precision, the magnitude data has a Q2.8 fixed-point precision whereas the argument data has a Q9.1 precision. From here the datapath splits into two parallel paths which are structurally identical, each path processes either the magnitude or argument data. Each path consists of a sorting block followed by a DBSCAN module, there is therefore 2 of each of these modules within the complete system. As previously mentioned, the sorting and DBSCAN modules are designed to intricately support each other and each operate in cycles of n, where n equals the dataset size. Prior to the sorting unit there are $n/10$ buffer registers to reduce fanout which introduce a cycle of latency. Each sorting unit sorts batches of n datums as they exit from the CORDIC module which is performed in n clock cycles, therefore as the nth datum enters the sorting module a size n batch of data is sorted. Following this a batch is output serially to DBSCAN and sorting of a new batch begins. DBSCAN accepts a new datum every clock cycle as it is serially output from the sorting block, it iteratively takes the difference between two local values and compares with a user-defined ε value. Following $n-1$ difference operations being performed by each DBSCAN module the number of clusters within the argument and magnitude batches has been found, these 10-bit results are then loaded into the MLP as 24-bit signed Q17.7 fixed-point values. The MLP is the final stage of the datapath, as suggested by the input data precision it features a 24-bit datapath, weight and bias values are stored as 16-bit values with

Q9.7 fixed-point precision to reduce utilisation requirements. The MLP itself requires 5 clock cycles to execute with the first cycle required to load new values, and a single clock cycle to compute the multiplications or sums of both the hidden and output layers. Following these 5 clock cycles the regression result becomes available, the value of the 0th output node is taken as the regression result. The classification result requires an ARGMAX function to iteratively compare the output node values to find the largest, which takes a further C clock cycles, where C is equal to the number of output nodes.

The system operates with a maximum clock frequency of 142.86MHz, equivalent to a 7ns time period. This latency was found to be the maximum operating frequency due to limitations imposed by the structure of the sorting unit, the reasons for this limitation are outlined in Section 5.3.

5.1.2 Datapath Latency and Throughput

The latency of the system may be defined according to multiple definitions. These are:

- The time taken from when the first datum pair of a batch enters the system to when a classification result is obtained.
- The time taken from when the first datum pair of a batch enters the system to when a regression result is obtained.
- The time taken from when the final datum pair of a batch enters the system to when a classification result is obtained.
- The time taken from when the final datum pair of a batch enters the system to when a regression result is obtained.

To illustrate the reasons for each definition the initialisation routine may be analysed. Upon initialisation the CORDIC module begins performing rotations, taking 14 cycles from input to output, after 2 further cycles the rotated pair to enter their respective sorting unit. A further n cycles are required to fill the sorting unit, another n cycles are required to perform DBSCAN on the sorted dataset. Finally, the MLP requires 5 cycles to load data and execute, resulting in a total of $14 + 2 + n + n + 5$ cycles or $2n + 21$, this is the initialisation latency before a regression result is obtained, to obtain a classification result the ARGMAX function must be accounted for which necessitates a further C cycles of latency, providing $2n + 21 + C$ cycles.

However, the systems from the literature with which the proposed system will be compared against are not designed to operate on a datastream, as such their latency calculations assume that a batch of data is at the input, preprocessed, and ready for classification [18,39]. While the proposed system is technically operating from when the first input pair of a batch arrives, this is a required waiting period which is ignored in the latency calculations of other systems. To provide a more informed comparison later in this chapter, the latency of the system will be calculated according to when the final datum of a batch arrives at the input to the CORDIC unit, therefore a total of $n + 21$ for regression and $n + 21 + C$ for classification.

Due to the pipelined datapath, apart from the MLP each module is continuously operating. So while the latency of the initialisation routine is a maximum of $2n + 21 + C$, once a result is obtained, a new result is obtained every n cycles. Given a clock period of 7ns the

system provides a classification or regression result every 7μs with a dataset size of $n = 1000$, therefore resulting in 142,857 classification or regression operations per second, and providing the potential to process 143 million I/Q pairs per second. Given that a 5G NR system has a nominal sample rate of the baseband signal of 122.88 Mega samples per second (MSPS) [109], the throughput of the proposed system would be capable of handling such rates with a 16% overhead, the clock period could therefore be increased to 8ns to allow for the relaxation of design constraints in such a scenario.

5.1.3 System Control

Now that the datapath has been detailed, the control may be explained, Figure 5.2 shows the elaborated design, of particular interest to this section is the *SR_chain_control* module which acts as the control unit for the entire system, it began as the control unit for the sorting modules but was later expanded to control all modules. It is placed inside the larger sorting module for placement within the same pblock as the $n + 1$ bit enable signal is sensitive to signal delays.
Figure 5.2 shows that 3 control signals are output from the control unit, *MLP_en*, *enable*, and *final*, which control the MLP, the sorting chains, and the DBSCAN modules respectively. Recall that the sorting and DBSCAN modules operate in cycles of n clock cycles, as such within the control unit there is a 10-bit counter which counts to n, after n cycles it resets to a value of 1. The three output control signals have their value governed by the current value of this counter. The *final* control signal is set to a value of 1 when the counter value is equal to n, it signals that 999 difference operations have taken place and to output the final result. The *MLP_en* signal is active between a counter value of 1 and 23, it signals that the DBSCAN outputs are ready and that the MLP should commence operation. The $n + 1$-bit *enable* signal manages the ratio of registers within the sorting modules which are currently outputting data to the DBSCAN modules or sorting the next batch, how this exactly relates the sorting logic will be explained further in Section 5.3. The *enable* signal is structured in such a way that it is a string of consecutive 0s and 1s, for instance take a 9-bit enable signal for an $n = 8$ sorting module, on a counter value of 1 the signal will be 011111111, at 2 001111111, this continues until the counter value is equal to n where the control signal is 000000000, signifying that all the sorting registers are actively sorting. The ratio of 0 to 1 governs the ratio of sorting module registers actively sorting or outputting a previous batch. Care must be taken to ensure that only a single bit of this control signal changes per clock cycle, actively setting each bit every clock cycle led to timing errors.

A pseudo control signal is determined by a second counter within the control unit, upon initialisation this counter counts to 14 before enabling the main counter to begin incrementing, this is to allow for the CORDIC unit to flush with input data before sorting begins.

A minor inefficiency is that the DBSCAN enable signal is governed by the input *Start* signal, this signal is always set to 1, therefore DBSCAN is always operating. This means that for $n+15$ clock cycles following the first input pair it is needlessly switching, over a long execution this inefficiency is minimal but could be optimised if desired.

The broad structure, datapath flow, and control of the overall system has now been demonstrated. The following subsections will explore the design of each component module in greater detail, providing explanations of the decisions which govern the structure of each module in addition to the implementation statistics. Following this, the implementation and

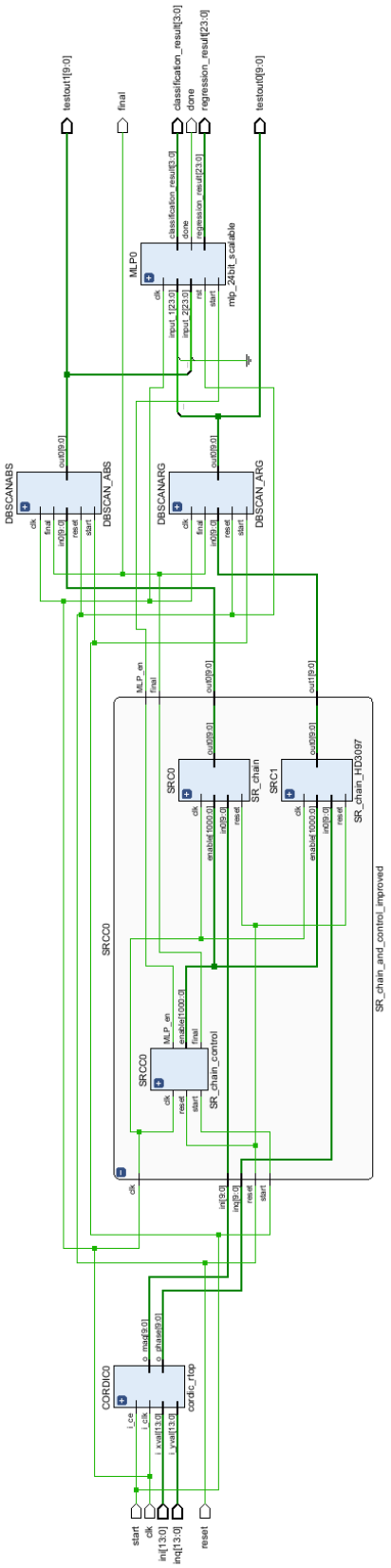


Figure 5.2: *The DBMC/DBSNR Elaborated Design*

verification strategy of the proposed system is given within the context of the requirements of each module and the system as a whole.

5.2 The CORDIC Module

The importance of converting I/Q values into their argument and absolute value form was made clear in Chapter 4. Converting I/Q data to the absolute and argument form can be done with the Equations 5.1 and 5.2:

$$A = \sqrt{I^2 + Q^2} \tag{5.1}$$

$$\phi = \text{atan2}(Q, I) \tag{5.2}$$

It was explained in Section 4.3 that the issue with performing both computations above is that they involve operations which are costly in terms of the number of clock cycles required. The square root and $atan2$ operations are generally both individually executed using the CORDIC algorithm, an iterative algorithm which results in an approximation of the desired result. Rather than performing both operations with separate CORDIC units it is possible to convert 2 rectangular inputs to their polar representation with a single CORDIC implementation, a detailed explanation of this process from an algorithmic perspective may be found in Section 4.3.

5.2.1 Rectangular to Polar Conversion with CORDIC

It was crucial that the CORDIC algorithm was implemented in such a way that the data path pipeline was maintained, therefore an unrolled structure was required. It is designed to accept 14-bit I and Q values as this is the typical precision at which an ADC digitises 5G signals in modern communication systems [110]. The CORDIC datapath operates with 15-bit precision to account for overflow however the separate phase calculations use 19-bit precision. It outputs truncated unsigned 10-bit values as the subsequent system datapath demands. Before the data enters the CORDIC logic, the first bit of the I and Q values is checked for a sign. The clockwise pre-rotation is applied the I/Q pair to ensure that the I/Q vector lies within the required $\pm 45°$ for the CORDIC algorithm to operate. The rotated I and Q values along with the pre-rotation angle then enter the CORDIC system proper, the first CORDIC stage is incorporated into this pre-rotation step to save a clock cycle.

A diagram of the CORDIC implementation can be seen in Figure 5.3. The CORDIC structure can feature up to N pipelined stages, each stage consists of 3 adders and 2 shift registers. The I value is assigned to X, the Q value to Y. The sign of the Y value determines the direction of rotation by putting the adders in add or subtract mode. In each stage the Z value has a pre-determined rotation argument added to or subtracted from the total rotation. Following the completion of N stages the X value will hold the pre-adjusted magnitude, the Y value should be 0, and the Z value will be the argument. Finally, the magnitude held in X will have its value adjusted using a LUT to compensate for the growth which occurred during the CORDIC operation. k cycles are generally required to achieve k bits of precision [92], however as the output values are truncated it was found that performing the 12 CORDIC iterations and then truncating allows for minimal error in the output values.

5.2.2 Rectangular to Polar Conversion with CORDIC Implementation Statistics

The CORDIC implementation of the rectangular to polar converter is fully unrolled and can process a continuous stream of data as required by the rest of the system. The Worst Negative Slack (WNS) was found to be 1.388ns and the Worst Hold Slack (WHS) was 0.091. The latency of the design is 14 clock cycles, the first cycle loads the input I/Q pair, the second performs the sign checking, pre-rotation, and initial CORDIC stage, 11 cycles then perform each further CORDIC iteration, the final cycle performs the magnitude scaling with DSP and truncates the ouputs to 10-bit precision.

The dynamic power consumption of the CORDIC implementation uses a total of 52mW, a full breakdown is shown in Table 5.1.

Table 5.1: *Dynamic Power Consumption of the CORDIC Functional Block*

	Clocks	**Signals**	**Logic**	**DSP**	**Total**
Power (mW)	4	36	11	1	52
Percentage %	7.7	69.2	21.2	1.9	100

The utilisation of the CORDIC implementation is shown in Table 5.2.

Table 5.2: *FPGA Resource Utilization of the CORDIC Functional Block*

Element	**LUTs**	**FFs**	**DSPs**	**Slices**
Number	586	575	1	194
% of total	1.1	0.54	0.45	1.46

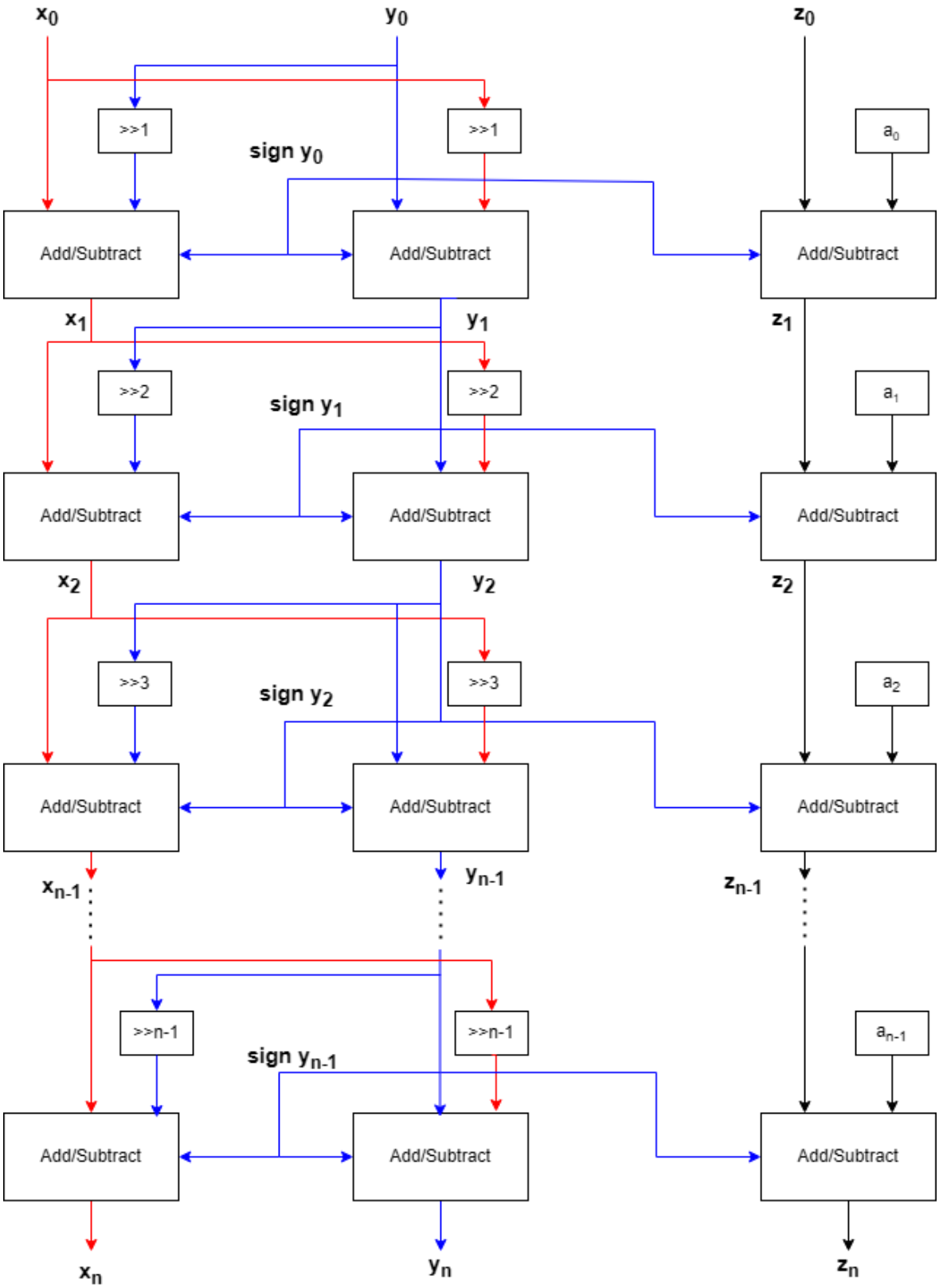


Figure 5.3: *The Operation of the Rectangular to Polar CORDIC Conversion Block*

5.3 Real-Time Sorting Unit Implementation

The optimisations provided to DBSCAN by utilising 1D data were discussed in Section 4.2. It was explained that 1D data can be sorted by magnitude to allow for DBSCAN to execute with a computational complexity of $O(n)$. Although a reduction in DBSCAN execution time may be achieved, an additional step of having to sort the dataset is introduced, for a total reduction in computational complexity to be realised across the entire system the data must be sorted in an efficient manner. While most common software sorting algorithms are designed to operate on an entire dataset at once and have a lower-bound computational complexity of $O(nlogn)$ [89], there exists hardware sorting algorithms capable of sorting a streaming input with $O(n)$ complexity. Bitonic sort [111] and merge sort [112] both see frequent use due to their parallel structure enabling efficient sorting in hardware. However, for the purposes of this system it was found that Bitonic sort was not suitable owing to the full dataset being required to be available before sorting begins, merge sort also has an initialisation latency making it unsuitable for this task where sorting must begin as the first datapoint arrives. Thus it was decided to develop a bespoke algorithm which offers guaranteed $O(n)$ complexity as well as requiring $O(n)$ resources. To the best of the author's knowledge this is a novel sorting implementation.

5.3.1 Sorting Algorithm Implementation Structure and Operation

In Figure 5.4 a block diagram of the sorting system is shown. The system consists of a chain of n 10-bit registers each connected to the input via a comparator, n being the number of datapoints which DBSCAN will execute upon. The system can be as large as required, although a maximum of 1000 was used in this work. The system accepts a single datum on the left-hand side as the input. If it is the first datum to enter the system, it is stored in $SR - 1$. For all data points other than the first, the stored value in each shift register is compared to the input via its respective comparator. Should the input datum be larger than the value stored in a shift register, the value stored in a register will move down 1 place in the chain. If the value in the shift register is larger than that of the input datum, the shift register's value remains unchanged. This system leads to all values which are smaller than the input datum shifting down 1 position, leaving a gap for the input value to be stored. Repeating this process n times leads to an array of sorted values. The process of placing an incoming input value into the correct shift register can be seen visually in Figure 5.5. In Figure 5.5 an input value of "1111111001" is required to be sorted, this value is represented by blue. Each comparator outputs a control signal of 0 or 1 to its connected shift register, if this signal is 0 (represented by green in the diagram) no action is taken, if the signal is 1 (represented by red in the diagram) the currently stored value shifts down the chain into the next shift register. The control logic for each shift register has access to the comparator result above it in the chain as well as its own, if the output value of the above comparator is 0 and its own value is 1 then it stores the input datum.

As DBSCAN requires all n datapoints to be available before execution can begin, this sorting algorithm essentially requires 0 clock cycles due to it operating during a required waiting period. When the sorting algorithm module is reconfigured to accommodate larger datasets the number of comparators and registers as well as the number of clock cycles for operation is increased to the required dataset size.

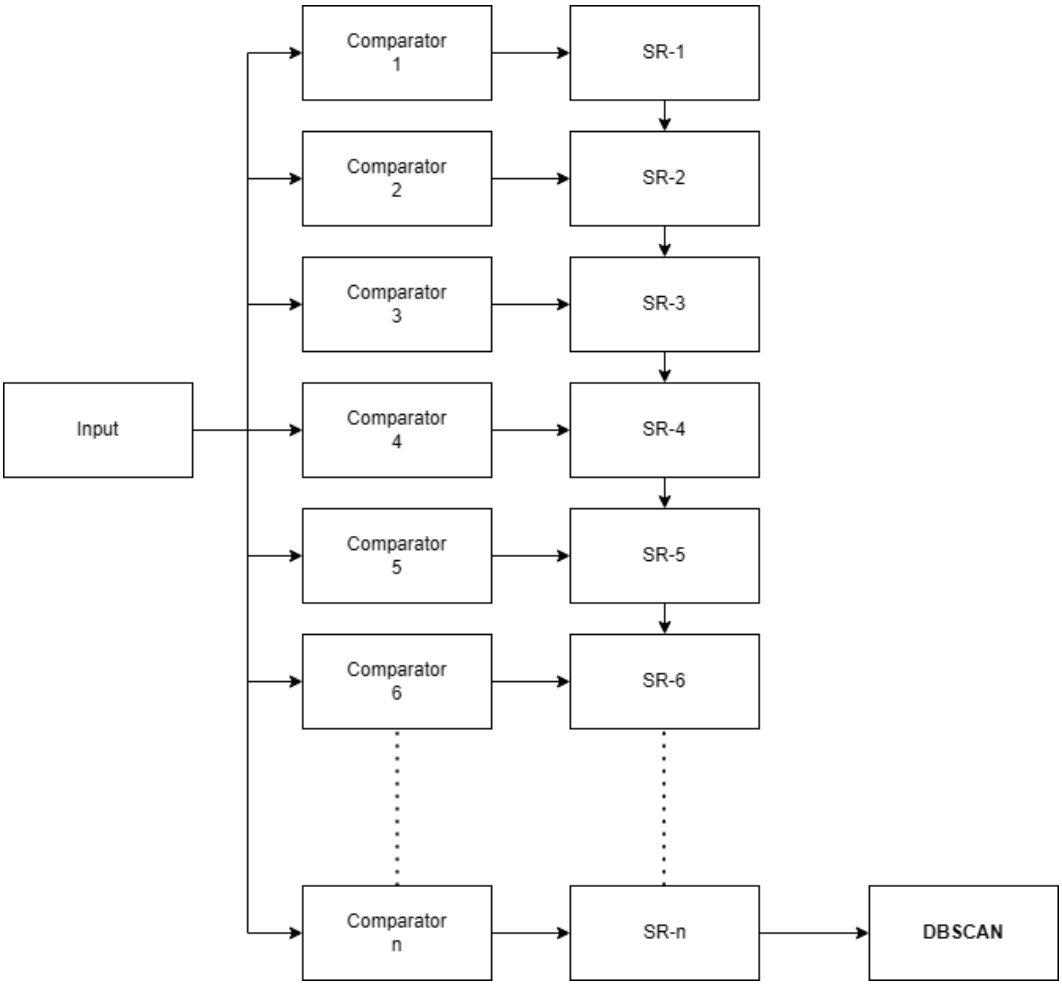


Figure 5.4: *The System Diagram of the Proposed Sorting Algorithm*

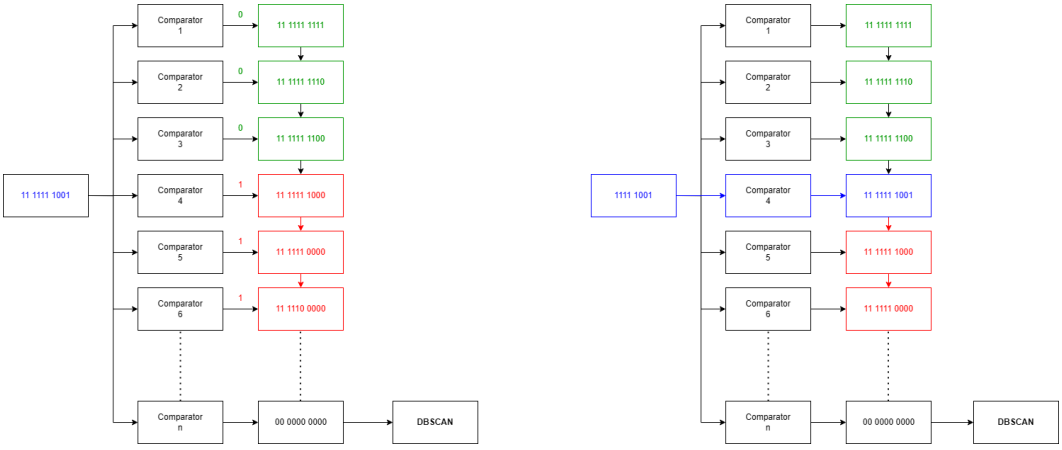


Figure 5.5: *The Operation of the Proposed Insertion Sort Algorithm Implementation*

5.3.2 Sorting Algorithm Serial Output

Following n datapoints being sorted, they are serially loaded into the DBSCAN module. Registers at the top of the chain then begin to sort another dataset, as more datapoints of a prior dataset leave the system more registers are freed-up to enable the storage of the next dataset. This functionality is controlled by the control unit which ensures that only x registers are available for new data to be loaded x cycles into a new sorting operation. The $n + 1$-bit *enable* signal which is output from the control unit informs each shift register in the shift register chain of the expected behaviour for that clock cycle. Each shift register has access to 2-bits of this enable signal, a received value of 00 indicates normal behaviour as previously described, a value of 11 indicates that the shift register should only load the value stored in the above registers, this behaviour is required to serially output a sorted sequence to the DBSCAN module. A value of 10 is an edge case which signals that the register should load the input value if the value held in the above register is larger than the input value or load the above value if it is not, the currently held value's relationship to the input value is ignored as the register currently holds the largest value from a previous sorting operation. Figure 5.6 visually demonstrates this process. This mechanism allows for the sorting modules to operate as both a sorting module and a storage array, without this method another $2n$ 10-bit registers would be required, resulting in nearly a doubling of the FF utilisation.

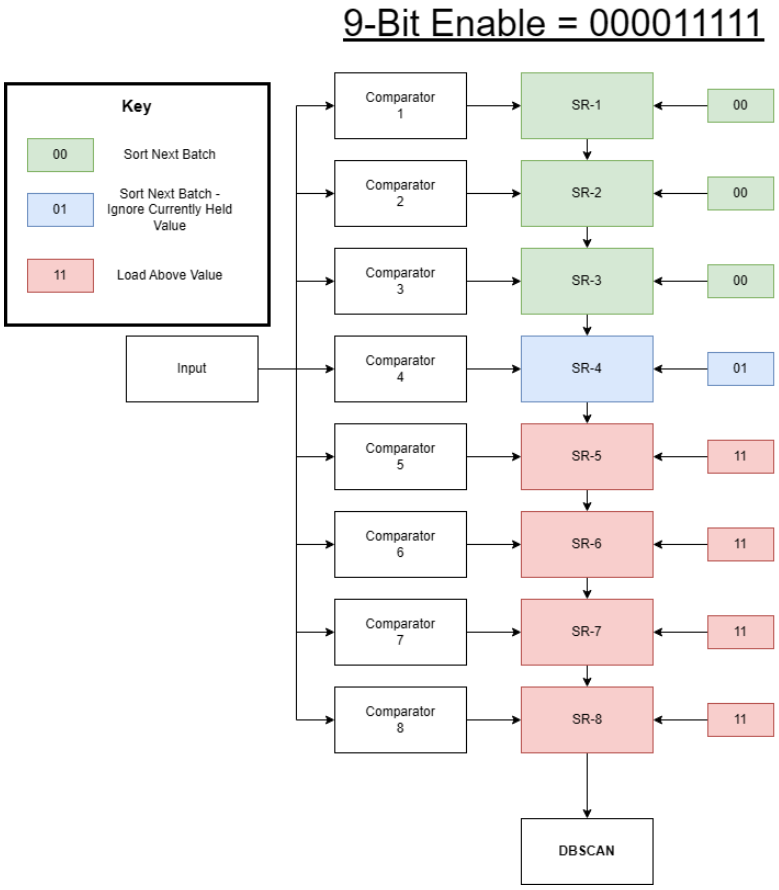


Figure 5.6: *How the Enable Control Signal Determines Register Functionality*

5.3.3 Sorting Algorithm Implementation Statistics

The sorting algorithm passes timing checks at the target 142.86MHz clock frequency with a setup WNS of 0.334ns and a hold WNS of 0.060ns in the 1000 dataset size configuration. The paths with the smallest WNS are the paths through the combinatorial comparator logic. A pipeline step of input buffer registers was created to limit fanout, it was found that 1 level of $n/10$ registers was required for an n point configuration. Ultimately meeting timing requirements with this module necessitated a clock frequency reduction to 142.86MHz from the target 200MHz. The latency savings by the utilisation of this design were deemed to be advantageous over implementing a more traditional sorting algorithm at a higher clock frequency due to maintaining an $O(n)$ computational complexity and the pipelined datapath.

The dynamic power consumption of the sorting algorithm in a 50, 250, 500, and 1000 datapoint configuration is shown in Table 5.3. The overview of the FPGA elements required for these datapoint configuration is also shown in Table 5.4.

Table 5.3: *Dynamic Power Consumption (mW) of the Sorting Unit for Various Configurations*

	Clocks	**Signals**	**Logic**	**DSP**	**Total**
Power 50 (mW)	4	9	9	0	13
Power 250 (mW)	18	42	62	0	123
Power 500 (mW)	35	85	125	0	245
Power 1000 (mW)	70	171	249	0	490

Table 5.4: *Sorting Unit FPGA Resource Utilization for Various Configurations*

Resource	**LUTs**	**FFs**	**DSPs**	**Slices**
Number - 50	804	550	0	592
Number - 250	4020	2750	0	1361
Number - 500	8068	5500	0	2712
Number - 1000	16128	11000	0	5409

The sorting unit is the only datapath module in the system which increases in size when the system is modified to accept larger datasets, while 4 different configurations are discussed in this section, any number of dataset sizes can be accommodated as long as there are sufficient available slices on the FPGA. In both tables it can be seen that the power consumption scales with the increase in the required elements to realise the design. When considering the 50-point configuration, the implementation size and power consumption values are roughly equivalent to the values seen in other modules in the system. However, for larger configurations this module becomes the most power consuming and largest in terms of chip utilisation of any module in the system, this is further compounded by the fact that 2 sorting units are required to realise the complete DBSCAN modulation classification system. Therefore, it is advisable to engineers who wish to use the DBSCAN classifier to tailor the size of the dataset to the modulation schemes which they are expecting to classify as doing so will provide a classification system which is as quick and efficient as possible, details about how the dataset

size impacts classification accuracy are provided in Chapter 6. The next Section will discuss the DBSCAN clustering module which lies immediately after the sorting units in the complete classification system.

5.4 Modified DBSCAN Algorithm Implementation

The modified DBSCAN algorithm is the key component of this classification system as it is the mechanism by which the features for classification are generated. The algorithmic development of this modified clustering algorithm is detailed in Section 4.2. The following section will discuss the implementation and operation of the algorithm in detail as well as the specific changes to made to the algorithm to optimise for throughput, implementation size, memory requirements, and power consumption.

5.4.1 Operation of the Modified DBSCAN Algorithm

The block diagram for the hardware implementation of the 1D DBSCAN algorithm can be seen in Figure 5.7.

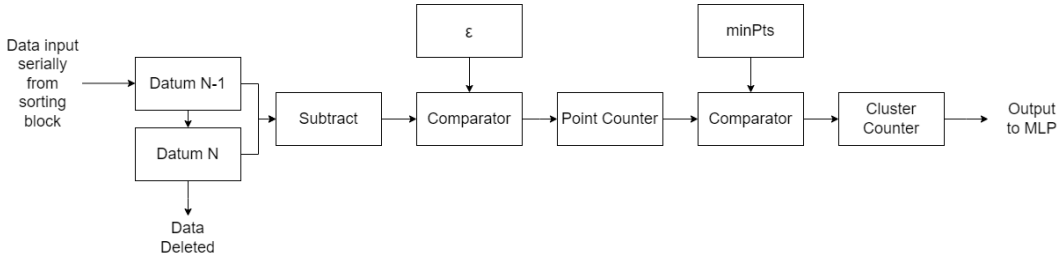


Figure 5.7: *The Block Diagram of the Proposed Modified DBSCAN Algorithm's FPGA Implementation*

It can be seen from Figure 5.7 that the modified 1D DBSCAN algorithm has been efficiently implemented with 2 10-bit registers to hold each data point, a 10-bit full adder for subtraction, 2 10-bit registers to hold *PointCount* as well as *ClusterCount*, 2 10-bit comparators, and 2 10-bit LUTs to hold the ε and *minPts* values.

The dataset which is input to the DBSCAN block has been pre-sorted, therefore there is no hardware required to execute a *rangeQuery* function other than taking the difference between 2 points which lie adjacent in the dataset. Unlike traditional DBSCAN implementations, the constellation points are not required to be labelled as part of a cluster, only the total number of clusters is required to achieve modulation classification. This allows for saving in the total memory space required as only the maximum label value is required to be stored, rather than labels for every datum. Furthermore, once the difference between a datapoint's neighbours has been found it is no longer required to be stored, it is therefore discarded to reduce utilisation requirements.

This DBSCAN implementation always requires n clock cycles to produce a result, where n is the number of points in the dataset. At a clock frequency of 142.86MHz with a 50-point dataset this results in a latency of 364ns. No hardware modifications are required to realise

an implementation capable of accommodating datasets larger or smaller than 50, other than operating the algorithm for a number of clock cycles equal to the dataset size n.

5.4.2 Modified DBSCAN Implementation Statistics

The DBSCAN module passed timing with a setup WNS of 1.259ns and hold WNS of 0.184ns at a clock frequency of 142.86MHz. The total dynamic consumption of the DBSCAN module is only 3mW, the lowest of any module in the system. A complete breakdown is shown in Table 5.5.

Table 5.5: *Dynamic Power Consumption of the Modified DBSCAN Algorithm*

	Clocks	**Signals**	**Logic**	**DSP**	**Total**
Power (mW)	1	1	1	0	3
% of Total	33.33	33.33	33.33	0	100

The low power consumption is testament to the degree of optimisation of the implementation. An FPGA implementation of the traditional DBSCAN [113] exhibited a power consumption of 570mW in its lowest power configuration.

A breakdown of the FPGA elements utilised by this design is shown in Table 5.6.

Table 5.6: *FPGA Resource Utilization of the Modified DBSCAN Algorithm*

Resource	**LUTs**	**FFs**	**DSPs**	**Slices**
Number	54	60	0	28
Percentage	0.10	0.06	0	0.21

Similarly to the power consumption, the optimisations resulted in a very small implementation, the smallest of any module in this system. In comparison to an implementation by S. Shi et al. [114], their implementation of a parallelised DBSCAN utilised 30300 slices of a Virtex-7 FPGA, the 2 DBSCAN and sorting blocks in this thesis utilise 10874 in the $n = 1000$ configuration. It is hard to draw an exact comparison as both implementations were tested on different hardware but the 64.1% reduction in slices required should illustrate the effectiveness of the optimisations made to DBSCAN in this thesis.

5.5 MLP Implementation

The final stage of the classification and regression system is an MLP. This ML subsystem accepts 2 inputs which are the number of different argument and magnitude clusters and outputs a classification of the most likely modulation scheme or a regression result which is the signal SNR. Other than the sorting unit, the MLP is the largest and most power consuming part of the complete system but following a review of all available technologies it was found to provide the best accuracy to implementation cost ratio. The process of model selection may be found in Section 4.4.

5.5.1 MLP Structure

The implemented MLP consists of 3 layers, the input layer features 2 nodes for both inputs, the hidden layer has 3, the output layer requires C nodes where C is the number of different modulation schemes included in the training set. Its structure can be seen in Figure 5.8.

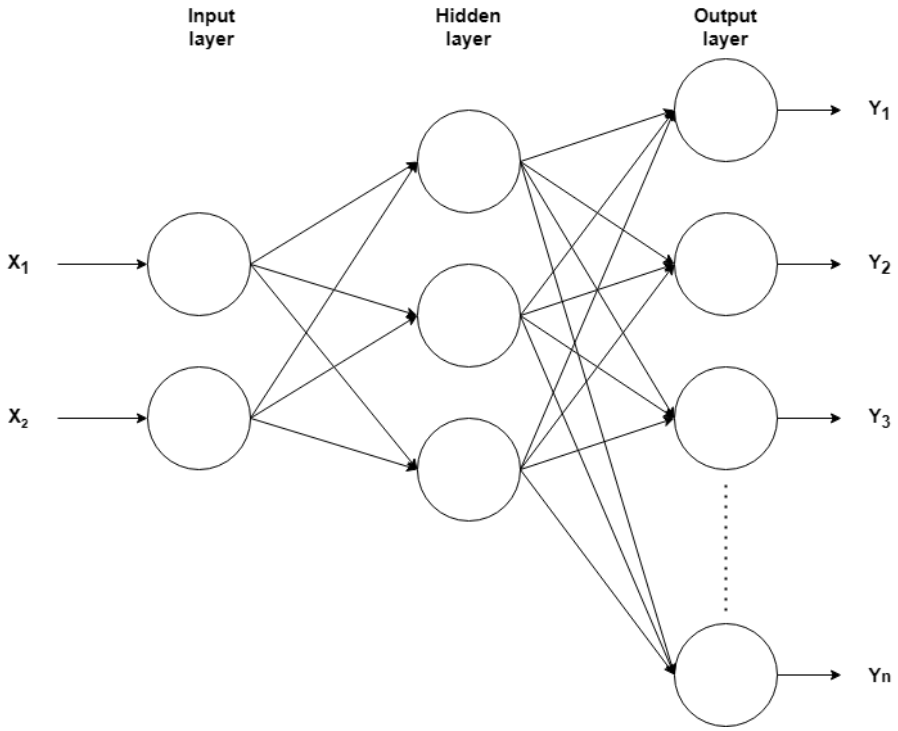


Figure 5.8: *The Structure of the MLP Classifier Implemented in the Proposed System*

The value of the number of different arguments and magnitudes enter the MLP on the left, denoted by X_1 and X_2. These values flow through the model, undergoing successive operations at each node until a result is reached, the classification result is determined by the largest value of the output nodes Y_1 to Y_n. At each node the operation to achieve an output is as follows. Each input to a node is multiplied by a weight, the products of each node input and their respective weights are summed with an additional bias weight to produce a nodes output, as shown in Equation 5.3.

$$h_i = \text{ReLU}(w_{0,i} \times X_0 + w_{1,i} \times X_1 + b_i) \tag{5.3}$$

The outputs of the hidden nodes are applied to a ReLU function. To reduce system implementation size, the training of the weights and biases was performed with PyTorch 2.0 [100] using nn.linear [101]. The learned weight and bias values were exported to a .CSV file and saved to the FPGA in a LUT. Weights and biases were trained assuming quantised values of 16-bit fixed point precision and then implemented as such. While the 2 input features were 10-bit values, the data path of this module was raised to 16-bits to account for the higher precision weights and biases. The choice of 16-bit precision lead to a significant reduction in

the implementation size of the MLP in comparison to using larger unquantised values, there was no loss of accuracy observed.

5.5.2 MLP Operation

The MLP is controlled by a state machine which consists of 7 states, Figure 5.9 displays a state machine diagram showing the transitions between states. The default state is *IDLE*. Upon receiving an *MLP_en* control signal the state switches to *COMPUTE_HIDDEN_MUL* which signals the inputs to be multiplied by the hidden layer weights and triggers the next state. The second state is *COMPUTE_HIDDEN_SUM* which sums the resulting products of the inputs and hidden layer weights with a bias and applies the ReLU function. The third state *COMPUTE_OUTPUT_MUL* finds the products of the hidden node outputs and the output layer weights. The fourth and final MLP state is *COMPUTE_OUTPUT_SUM* which sums the obtained products with the appropriate biases, no activation function is applied. Each of these states requires only a clock cycle to execute. The *ARGMAX* state is the only state which requires more than a single clock cycle to operate, its operation is explained in detail in the following subsection. Following the completion of *ARGMAX*, the state transitions to *DONE* which asserts a 1-bit *DONE* signal, the classification result is output, and the state transitions back to *IDLE*.

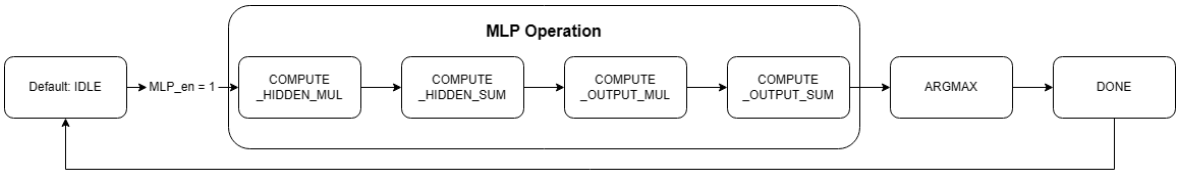


Figure 5.9: *MLP State Machine Diagram*

5.5.3 MLP ARGMAX Output

The index of the largest value of the outputs from the output layer is taken as the classification result. To determine the index of the largest value of the MLP outputs, all C outputs are iteratively compared with a single 24-bit comparator. The hardware structure to perform this iterative comparison may be seen in Figure 5.10. On the first of C clock cycles the value of Y_1 is compared with 0 to load its value into the *Max Value Register*, the *Max Value Index Register* then loads the value held in the counter which is 1. With each clock cycle the value held in *Max Value Register* is compared with value stored in Y_x, where x is equal to the clock cycle number. The largest value found by each comparison is stored in *Max Value Register*. If a change in the value held in *Max Value Register* occurs, then *Max Value Index Register* is updated with the current counter value. After C clock cycles the value held in *Max Value Index Register* is equal to the index of the maximum value, it is then output as the classification result.

This ARGMAX implementation imposes a latency of C clock cycles on the overall system latency, therefore if C is set to 17 then 17 additional clock cycles would be required. This latency could be reduced to a worst case of 5 cycles for a $C = 17$ scenario by performing the comparisons with an array of comparators in a tree configuration, but this would necessitate a larger implementation size. As the rest of the system is structured in such a way that

minimises implementation size, this linear search algorithm was implemented in the manner which results in the smallest possible utilisation.

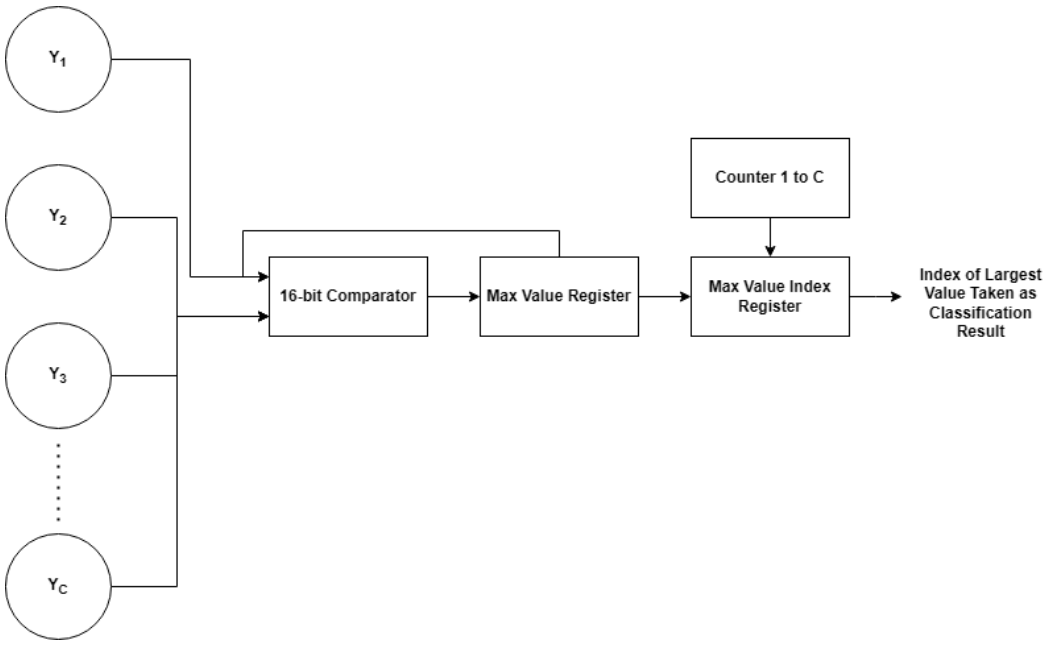


Figure 5.10: *The Structure of the MLP Linear Search Output*

The output of the 0th node is taken as the regression result, therefore the operation of ARGMAX is not required for SNR estimation.

5.5.4 MLP Classifier Implementation Statistics

The MLP and linear search algorithm was found to have a setup WNS of 0.451ns and a hold WNS of 0.187ns at a clock frequency of 142.86MHz. The latency of the design scales with the number of classes C due to the linear search ARGMAX algorithm requiring C iterations. The MLP itself required 5 clock cycles in total. Therefore the total latency of the entire functional block was found to be $C + 5$ cycles for classification or simply 5 cycles for regression.

As the number of output nodes must be scaled to accommodate the number of classes C, the implementation statistics are provided for 3 configurations, the value of C for each configuration is 4, 10, and 17. These configurations respectively represent the smallest configuration, an intermediate configuration, and the largest configuration which was utilised for testing, there were various other configurations utilised for testing but rather than listing each possible variation, it was decided to provide a maximum, minimum, and intermediate configuration to provide a measure of each extreme and how the utilisation grows with increasing numbers of classes.

The number of each FPGA resources used by each configuration can be seen in Table 5.7. In comparison to the CORDIC module the MLP is similarly sized, particularly in the $C = 17$ configuration. Despite the increases in utilisation which are required for increasing values of C, the utilisation is still dwarfed by the implementation size of the sorting unit, meaning that increases in C should not provide a significant difference in total utilisation. The relationship

Table 5.7: *FPGA Resource Utilization for Various MLP Configurations*

Configuration	LUTs	FFs	DSPs	Slices
C=17 Raw	1034	571	47	335
C=17 %	1.94	0.54	21.36	2.52
C=10 Raw	732	429	33	227
C=10 %	1.38	0.40	15.00	1.71
C=4 Raw	264	166	11	121
C=4 %	0.49	0.15	5.0	0.91

between C and the utilisation required for LUTs, registers, DSP, and slices scales as C increases. Each incremental value of C approximately increases required number of LUTs by 20, registers by 17, DSPs by 2.15, and slices by 11. There is optimisation room available without sacrificing any latency, as the DSP slices for the hidden layer multiplications are not operating during the output layer multiplications and vice versa, the same DSP blocks could be reused for both layers without a latency penalty but would incur a slight penalty to the number of slices due to additional logic requirements.

The dynamic power consumption breakdown for each MLP configuration is shown in Table 5.8.

Table 5.8: *Dyanmic Power Consumption Analysis for Various MLP Configurations*

Configuration	Clocks	Signals	Logic	DSP	Total
C=17 Power (mW)	4	29	21	64	118
C=17 % of Total	3.4	24.6	17.8	54.2	100
C=10 Power (mW)	3	17	16	43	77
C=10 % of Total	3.9	22.1	20.8	55.8	100
C=4 Power (mW)	2	8	7	21	38
C=4 % of Total	5.3	21.1	18.4	55.3	100

Similarly to the number of resources used, the DSP power consumption is responsible for most of the total power consumption. In general, the power consumption for the signals and logic also increases linearly with C.

5.6 System Control Utilisation

Tables 5.9 and 5.10 respectively show the power consumption and resource utilisation of the proposed control unit in the $n = 1000$ configuration.

Figures 5.9 and 5.10 show that the control unit is the second largest module in terms of FF and LUT utilisation. This is to be expected as the major role of this module is controlling the highly complex sorting unit. Despite the large size it consumes less power than the MLP as well as a similar amount to the CORDIC module.

Table 5.9: *Dynamic Power Consumption of the Control Unit*

	Clocks	Signals	Logic	DSP	Total
Power (mW)	7	16	8	0	31
% of Total	22.6	51.6	25.8	0	100

Table 5.10: *FPGA Resource Utilization of the Control Unit*

Resource	LUTs	FFs	DSPs	Slices
Number	1053	1041	0	413
Percentage	1.98	0.98	0	3.11

5.7 System Implementation, Vivado Settings, and Constraints

The proposed system passed timing checks with a 7ns clock period. To achieve this goal considerable optimisation has been performed. This section details the implementation strategy which is required to achieve timing closure and therefore predictable operation. To reiterate, the design was created with Vivado 2021.2 [105] and the target platform was the XC7Z020-CLG484-1 [108].

To begin, the module which ultimately led to the clock period being set to 7ns was the sorting units. There are numerous paths in this unit which require careful optimisation. Firstly, the path from the sorting unit input through the comparators to the registers themselves naturally has a large delay. Numerous constraints were utilised to ensure that this path remained within the required 7ns. Firstly the *set_max_delay* command was utilised to ensure that the delay paths from the input to the comparators and from the comparators to the registers was within 2ns, furthermore, the delay through the paths from the registers through the comparators and back to the registers was set to a maximum of 2.4ns. Secondly, the *set_property LOC_FIXED TRUE* command was employed to ensure that the comparators were always locally close to their attached registers. Thirdly, a high critical path and performance priority was given to these paths to ensure that the synthesis and implementation tools prioritised these paths. Finally, both sorting units and the associated control unit were placed in one pblock to further ensure that the implemented elements were locally close. This was particularly important for the $n + 1$-bit *enable* signal which also imposed timing issues due to it changing every clock cycle, should a value not update in time then an entire sorting operation became invalid.

The remaining modules were far less difficult to optimise and therefore required significantly less demanding constraints. The DBSCAN path from the two data registers through the full-adder and comparison with ε as well as the initial CORDIC sign check and pre-rotation presented some difficulties, but placing each of these modules within their own pblock solved any timing closure issues. All modules bar the MLP therefore are constrained within pblocks, it was decided to allow the Vivado tools to place the MLP logic anywhere to relax the constraints somewhat. Despite these optimisation the *reset* signal could not meet the timing requirements, therefore the *set_false_path* command was used to ignore these paths from timing checks. The *reset* signal failing timing checks is not an issue as it is not a part

of normal datapath operation.

The Vivado synthesis strategy was set to default whereas the implementation strategy was set to *Performance_ExploreWithRemap* to allow the tools to explore various implementations. Figure 5.11 shows the resulting implementation for an $n = 1000$ configuration, notice how the majority of slices are utilised as well as the large pblock dedicated to the *SR_chain* modules.

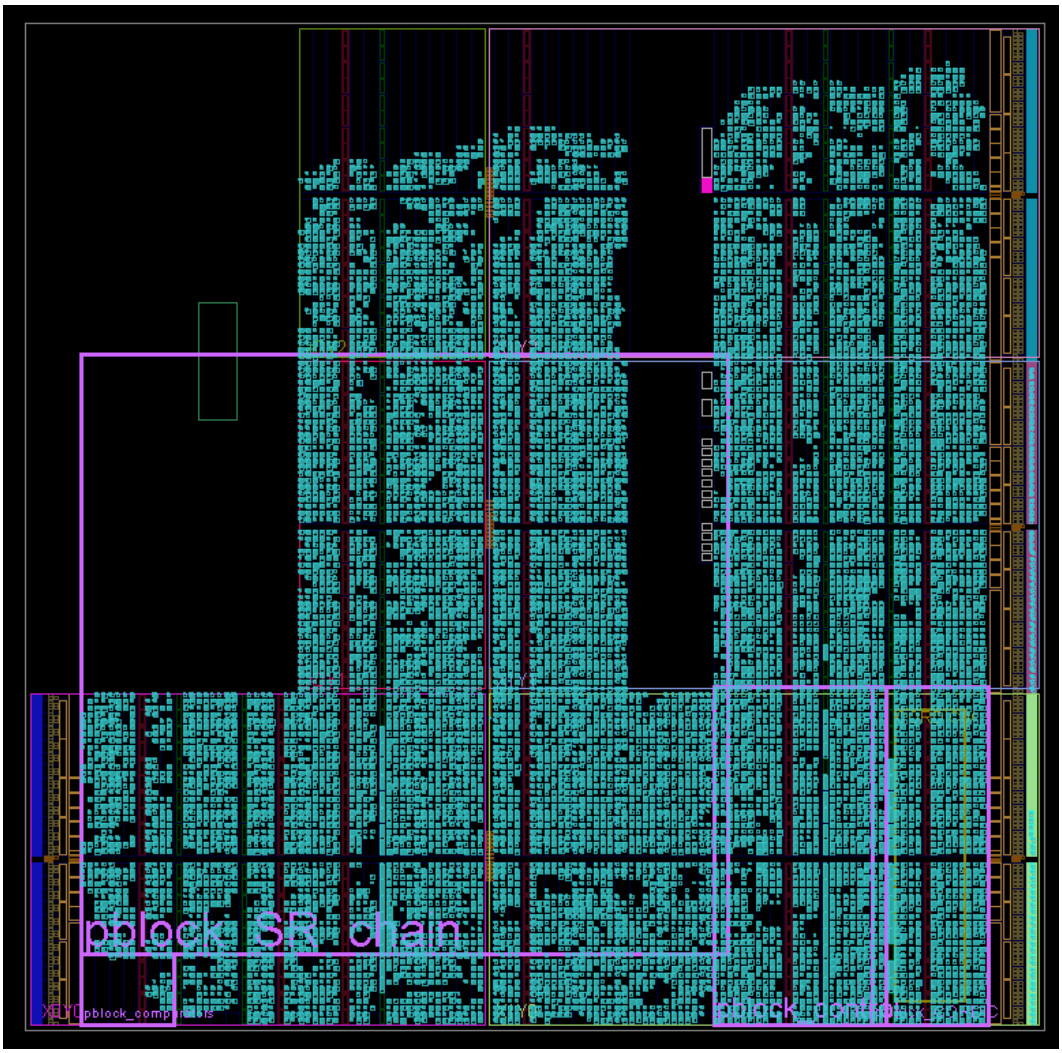


Figure 5.11: *Device Implementation Diagram*

It was ultimately found that the $n = 1000$ configuration passed timing checks with the narrowest margin of any of the proposed systems. With a setup and hold clock uncertainty of 80ps and 60ps respectively, the design passed timing with a WNS of 0.368ns and WHS of 0.030ns.

5.8 Implementation Verification and Testing

This section details the steps taken to create, verify, and test the system. Each step is explained and justified.

5.8.1 System Development

The creation of the system was performed in stages, each individual module was developed separately. The sorting module and accompanying DBSCAN module were co-designed. Next was the MLP, which initially featured a 64-bit datapath, weights, and bias values. Over many iterations the MLP was optimised to the current implementation, functionality to easily modify the number of hidden and output nodes was also added. The final module which was developed was the CORDIC functional block. Each module was independently verified, first with behavioural simulations and then post-implementation timing simulations. To perform these verification checks .csv files were created which provided input test vectors taken from the signals utilised to test and train the system and expected output values taken from a MATLAB prototype system.

The hardware system was then combined step by step, first by combining the sorting, DBSCAN, and MLP, then finally the CORDIC module. After each step the functionality was confirmed with behavioural and timing simulations. The final system verification steps were again performed using first behavioural and then post-implementation timing simulations. Again, .csv files were utilised to apply input test vectors and check the resulting outputs against the expected values. I/Q input values were drawn from the range of SNR and modulation scheme combinations utilised to obtain the final results provided in Chapters 7 and 8, both classification and regression functionality was tested. It was found that the obtained outputs matched what was expected in the vast majority of cases, the exceptions were when the input I/Q pairs were very close to 0. This only occurred when the constellation diagrams were so noisy that the algorithm would be incapable extracting useful information, in addition these cases were rare and did not contribute significantly to the classification result due to being one example out of a large batch.

5.8.2 System Verification

It was initially attempted to verify the system in hardware by connecting the FPGA to a PC. Directly interfacing with the FPGA from a PC was found to be unviable due to the input of the system requiring 28-bits of data every 7ns, resulting in a 4Gb/s bandwidth requirement, exceeding the rate of any link which was available. It was then attempted to batch load a dataset into the DDR3 memory, this method required loading PetaLinux [115] on the board processor and interfacing with it via custom scripts which required detailed knowledge of Ethernet protocols. Extensive work was performed to realise this verification setup but after countless hours of errors it was decided that dedicating more effort to this task was not a valuable use of time as this was not within the scope of the research. Two choices remained, either operate the system at a lower clock frequency and use a simpler streaming input via the USB OTG connection or directly load data via Verilog BRAM initialisation. Ultimately it was decided that the second option was preferable as operating at a lower clock frequency did not verify that the system could operate at the stated 142.86MHz frequency.

The implemented design was therefore verified on the target FPGA by initialising the FPGA BRAM with datasets of input I/Q values from a range of modulation scheme and SNR combinations, the expected outputs were stored in a LUT. The achieved outputs were compared with the expected classification results in the LUTs, an onboard LED was used to see if the expected outputs matched what was obtained, an example is shown in Figure 5.12. It was found that the results achieved by the programmed device exactly matched the achieved results in behavioural and timing simulations. Mixed Clock Mode Manager (MMCM) was utilised to generate the required clock frequency. It is unfortunate that a PC to FPGA interface was not achieved but the BRAM and LED method ultimately does verify that the proposed design operates as expected at the target clock frequency when implemented in hardware.

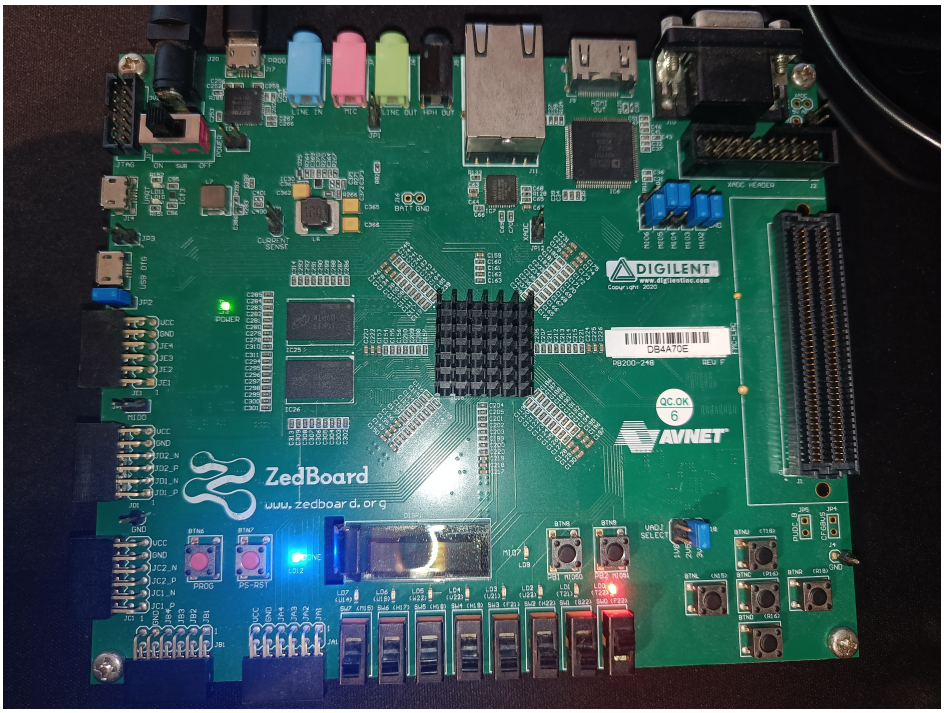


Figure 5.12: *The FPGA Verification Setup With Start Enabled by Switch F22 and LED T22 Indicating that no Unexpected Results Were Obtained*

5.8.3 System Testing Process

Once it had been verified that the system achieved identical results across behavioural and timing simulations as well as in hardware, testing was performed to obtain the accuracy results. As 149 separate tests were required to be performed, as well as each test requiring an additional run to obtain training data from the DBSCAN module outputs, results were obtained via behavioural simulations in Vivado to increase the rate at which tests could be performed. Despite a PC to FPGA link not being established, testing the proposed system via hardware implementation would have been impractical as new MLP weights needed to be found for each test, therefore necessitating running 2 implementation operations per change as well as programming the device for each change. The state-of-the-art system RUNet [39]

performed their testing in software simulations, demonstrating that while not ideal, this strategy is commonplace.

5.9 Complete System Implementation Statistics

The implementation size of each component of the DBMC modulation classification system has now been presented and discussed, the only remaining aspect of the hardware implementation to discuss is the utilisation of the system as a whole. As outlined throughout this work, the purpose of the DBMC system is to classify modulation schemes with a smaller FPGA utilisation, power consumption, and latency than the traditional deep learning-based methods whilst maintaining a comparable or stronger accuracy.

Before the comparisons are presented, it must be mentioned that many of the works found in the literature give no information about certain aspects of their system's performance, for instance some papers provide full statistics of the FPGA utilisation but neglect to mention power consumption or latency, although it would be ideal for all reported work to provide full statistics for their implementation, there are few examples of hardware implemented modulation classifiers in the literature so the strongest of the examples which do exist are included for a more complete comparison.

Another issue when comparing to other works is the absence of a standardised way of reporting FPGA implementation statistics, power consumption of course is always given in terms of Watts, but latency may be given in terms of clock cycles, delay in seconds, or throughput. The latency statistics of this work are given in terms of latency in seconds and conversions are made where required and possible. Additionally, conversions of the units are also performed for works found in the literature when enough information is given to give confidence in the accuracy of the conversion process, for instance, the information about the clock period which the work operates at is required to convert to latency in seconds from latency in clock cycles. The final issue with comparisons in terms of latency is that other works may operate on pre-recorded batches of data rather than a real-time data stream such as DBMC, half of the DBMC latency is waiting for the required amount of data to reach the input, context will therefore be given in the cases where comparisons are made to systems which operate on batches.

Implementation and synthesis settings in the Integrated Development Environment (IDE) as well as constraints may also cause the toolchain to prioritise various factors such as latency, timing, utilisation, and power consumption. Due to these differences across FPGA models and IDEs introducing uncertainty, where implementation statistics are similar, the benefit of the doubt will be given that the 2 works being compared are equal.

Finally, it was shown to be the case that classifying various numbers of different modulation schemes necessitates changes to the MLP structure to accommodate this functionality, it was also shown that for the most part the DSP utilisation is the statistic which sees the greatest change when these modifications are made. As different dataset size configurations are provided, and varying the dataset size has a significantly larger impact on the total utilisation, in the interests of keeping the comparison easier to read and digest, the utilisation statistics for each DBMC configuration will be provided using an MLP configuration which is representative of the maximum number of modulation schemes which each system can accommodate, for DBMC-50 this is 4, DBMC-250 and DBMC-500 have an MLP which can

classify 10 modulation schemes, and the results for DBMC-1000 are obtained with the largest MLP size of 17 classes. The reasoning behind the number of classes C for each configuration is made clear in Chapter 6. When the system is configured to operate as an SNR estimator the MLP is configured as a regressor rather than a classifier, in this case the MLP structure remains the same but the weight values to all but one of the output nodes are set to 0, the maximum value finding ARGMAX is also not required, the output of the 1st MLP node is taken as the estimated SNR.

5.9.1 Hardware Implementation Comparison

Table 5.11 presents the implementations statistics for all DBMC models alongside the best performing works from the literature. Works which achieve the lowest value in each utilisation category have their values bolded, in cases where a DBMC system achieves the lowest value all DBMC systems which achieve values lower than the literature are bolded.

Table 5.11: *Comparison of 4 DBMC Configurations with Hardware Implementations for AMC Algorithms from the Literature*

Implementation	FFs	LUTs	DSP	RAM	F(MHz)	Power(W)	Time(μs)	Hardware
Feature-Based DT[21]	16746	7933	180	14	100	–	15.79	XC7Z020-CLG484
RUNet[39]	21357	34563	**0**	40	–	–	7.5	ZCU111
QMCNet[39]	40476	61364	**0**	57	–	–	–	ZCU111
Baseline CNN[39]	54483	85151	**0**	70	–	–	–	ZCU111
TW-96 CNN[17]	369000	232000	1207	524	250	–	8	ZCU111
TW-BA-128[17]	333000	234000	1408	523	250	–	8	ZCU111
TW-INCRA-128[17]	324000	211000	1407	512.2	250	–	8	ZCU111
MobileNetV3[18]	25800	31200	162	22	250	4.2	–	ASIC
HISTO-SVM[20]	**462**	–	–	6144	–	–	20	Altera Cyclone II
DBMC-50	2135	**2741**	16	**0**	142.86	**0.125**	**0.525**	XC7Z020-CLG484-1
DBMC-250	6883	9697	34	**0**	142.86	**0.368**	**1.967**	XC7Z020-CLG484-1
DBMC-500	12637	17904	34	**0**	142.86	**0.503**	**3.717**	XC7Z020-CLG484-1
DBMC-1000	24297	34772	48	**0**	142.86	**1.188**	**7.252**	XC7Z020-CLG484-1

Beginning with the number of FFs, there is no DBMC configuration which achieves a smaller number of FFs than HISTO-SVM, but HISTO-SVM is an outlier in this regard as it is more reliant on RAM than FFs. Other than this outlier, the 3 smallest DBMC classifiers require fewer FFs than any other work which could be found in the literature, DBMC-1000 requires 31.1% more FFs than the feature-based DT classifier, it is therefore a significantly larger implementation in this regard. DBMC-1000 requires 1503 fewer FFs than the MobileNetV3 CNN [18] which is a small enough difference to be as a result of implementation settings. DBMC-1000 also requires 12.1% more FFs than the RUNet CNN [39], indicating that the efficient DBSCAN implementation has not enabled smaller FF utilisation than highly optimised CNNs.

The DBMC-50 configuration requires the least number of LUTs of any hardware-based classifier, the 3 largest implementations require more LUTs than the feature-based DT classifier but discounting this system they require fewer than any other system bar MobileNetV3 which requires 10.3% fewer LUT elements than DBMC-1000, again showing that the highly optimised CNNs match the utilisation achieved by the proposed work.

RUNet, QMCNet, and the accompanying baseline CNN require no DSP slices for their implementation, other than these systems all DBMC sizes require the least of any other hardware implementation by a significant margin. The system which requires the next least is once again MobileNetV3 which uses 162 which is a 237.5% increase compared to what is required for DBMC-1000. The DT classifier requires a number of DSP slices which is

similar to what is required by MobileNetV3, the ternary weight CNNs all require over 1000 DSP slices, which is 2 orders of magnitude greater than what is required by all DBMC configurations.

The final utilisation statistic to compare is the RAM usage. All DBMC systems require no RAM making them the only hardware implemented AMC system to do so. The feature-based DT and MobileNetV3 each require a small amount of RAM but neither eliminate its usage entirely, otherwise they are the systems which require the next least RAM. RUNet also requires minimal amounts of RAM as it uses only 40 blocks. HISTO-SVM system requires large amounts of RAM, this high RAM utilisation negates its otherwise compact implementation size.

Only 1 of the works found in the literature provide the power consumption of their proposed system. MobileNetV3 [18] was one of the CNN systems which achieved the strongest accuracy and is also one of the smallest implementations, but in terms of power consumption all DBMC systems are far more efficient, with even the largest DBMC configuration requiring 71.7% less power, showing that DBMC may be a better candidate for deployment in a mobile system despite the similar utilisation. The discrepancy in power consumption despite the similar implementation size is thought to be a combination of three factors. Firstly, the proposed design requires no RAM and far less DSP slices. Secondly the clock frequency of the proposed design is nearly half that of MobileNetV3. Finally, and perhaps most interestingly, the majority of the datapath featured sorted values, as adjacent values are similar by definition it was found that each clock cycle only a few FFs of each 10-bit register were required to update their value, resulting in minimal switching and therefore reduced power consumption.

The final statistic to compare is the latency, every DBMC configuration is quicker than any other classifier from the literature. DBMC-1000 approaches the latency of RUNet but is still 0.248μs quicker. This latency is testament to the highly optimised pipelined structure of the proposed system. The latency figures for RUNet and the TW CNNs are given assuming 1024 samples are at the input ready to be accepted into the classifier, they therefore do not take into account the time taken between when the first and last datapoint of the sample arrives. DBMC was designed with operation on a datastream in mind and therefore utilises this time to pre-sort the data before it enters the classifier, the sorting mechanism is technically part of the system so the added time could be included in the latency statistics but are not. If the sorting stage was included in the DBMC latency figures then they would be almost doubled.

5.9.2 Hardware Implementation Comparison Conclusion

DBMC-50 is overall the smallest, quickest, and most efficient classifier in comparison to every work which can be found in the literature, this is shown by requiring the least LUTs, RAM, power, and latency, as well as requiring the second least number of FFs and DSP slices. This conclusion discounts HISTO-SVM [20] as said design is largely RAM reliant as opposed to utilising logical hardware, making comparisons in this regard difficult. Although when included HISTO-SVM is the smallest in terms of FFs but the largest in terms of RAM and the slowest in terms of latency, making DBMC-50 on aggregate smaller and significantly quicker.

DBMC-250 also has a vastly smaller FF utilisation than any system in the literature other than HISTO-SVM. The LUT utilisation is however larger than the Feature-Based DT [21]. Despite this it is significantly quicker, and more power efficient than any other system which

provides values for these statistics.

DBMC-500 ranks the same DBMC-250 in all utilisation categories except is twice the size. It is beaten by the Feature-Based DT system in terms of LUT utilisation and the RUNet family of systems in terms of DSP utilisation. It is otherwise smaller, faster, and more efficient than any other logic-based systems.

DBMC-1000 achieves a similar level of utilisation to the smallest CNN classifiers, is larger than the Feature-Based DT system in terms of FF and LUT utilisation. As stated in the results analysis it requires more LUTs in comparison to MobileNetV3 and only slightly fewer than RUNet. Similarly, it requires considerably more FFs than RUNet and only marginally fewer than MobileNetV3. Like the 3 smallest DBMC systems it requires the second least number of DSP slices, the least RAM and the least amount of power. It has a lower latency than any system which could be found in the literature.

In the introduction it was stated that the DBMC system was designed to be smaller, more power efficient, and quicker than the currently available AMC systems, it can be concluded following this comparison that these goals have somewhat been achieved. The 3 smallest proposed systems do achieve smaller utilisations than the state-of-the-art CNNs but the largest configuration does not. Evaluation of each system's accuracy will therefore be required to determine whether the proposed strategy results in an implementation which achieves a smaller utilisation with competitive accuracy. Where the DBMC systems outperform the systems from the literature is in terms of RAM utilisation, power consumption, and latency.

It was also stated that the modulation classification system should be capable of classification of a real-time stream of data, throughout this chapter the steps taken to realise this goal have been made clear via the intricate pipelining of each functional block and the novel sorting algorithm to enable a computational complexity of $O(n)$ to be maintained, thus it can be concluded that this second goal has been achieved. It is however unfortunate that the sorting unit which was devised to maintain the pipelined architecture ultimately led to the large utilisation of DBMC-1000, it was found that this module scales poorly with large dataset sizes.

The hardware implementation of the DBMC system has now been detailed and compared with the literature, all that remains to fully characterise this implementation is to quantify the accuracy and draw comparisons to what is achieved by the works which could be found in the literature in this regard.

Chapter 6

Parameter Optimisation

Chapter 4 discussed the algorithmic implementation of the proposed DBMC feature extractor and accompanying classifier, Chapter 5 covered the hardware implementation of the proposed system. Throughout each section 3 parameters were mentioned but not expanded upon, these are the DBSCAN dataset size n, as well as the two DBSCAN hyperparameters ε and $minPts$.

The performance of the proposed system is highly sensitive to the choice of these hyperparameters, however the selection process is challenging due to their inherent interdependency; the optimal value for each is contingent upon the values of the others. The selection process is further complicated by the dependence of optimal hyperparameter values on both the SNR range and the modulation schemes targeted for classification.

In the following chapter the interplay between each of these variables will be investigated, the effects of choosing different values will be discussed, and recommendations for optimal performance will be made, including the proposal of a novel ε selection heuristic which provides increased performance over traditional selection methods while being nearly completely automated.

6.1 An Ideal Hyperparameter Selection Example

In order to contextualise the effects of suboptimal hyperparameter selection on the dataset and extracted features, a benchmark must first be established by presenting an idealised scenario before proceeding with a detailed analysis.

The analysis of the DBMC modulation classifier operation and optimisation is performed by comparison of the relative spacing of class clusters in feature space.

The DBMC modulation classification system can be thought of as consisting of two components, a feature extractor and a classifier which utilises the obtained features to predict the modulation scheme of the input dataset. The feature extraction and resultant features will be explained first.

The algorithmic operation and hardware implementation of the DBSCAN feature extractor was outlined in Chapter 4 and 5. In brief, I/Q samples of modulated signals are converted to their magnitude and argument representation, within the obtained sets of arguments and magnitudes there exists clusters of similarly valued data, the number of clusters which are obtained is then utilised as the feature for classification.

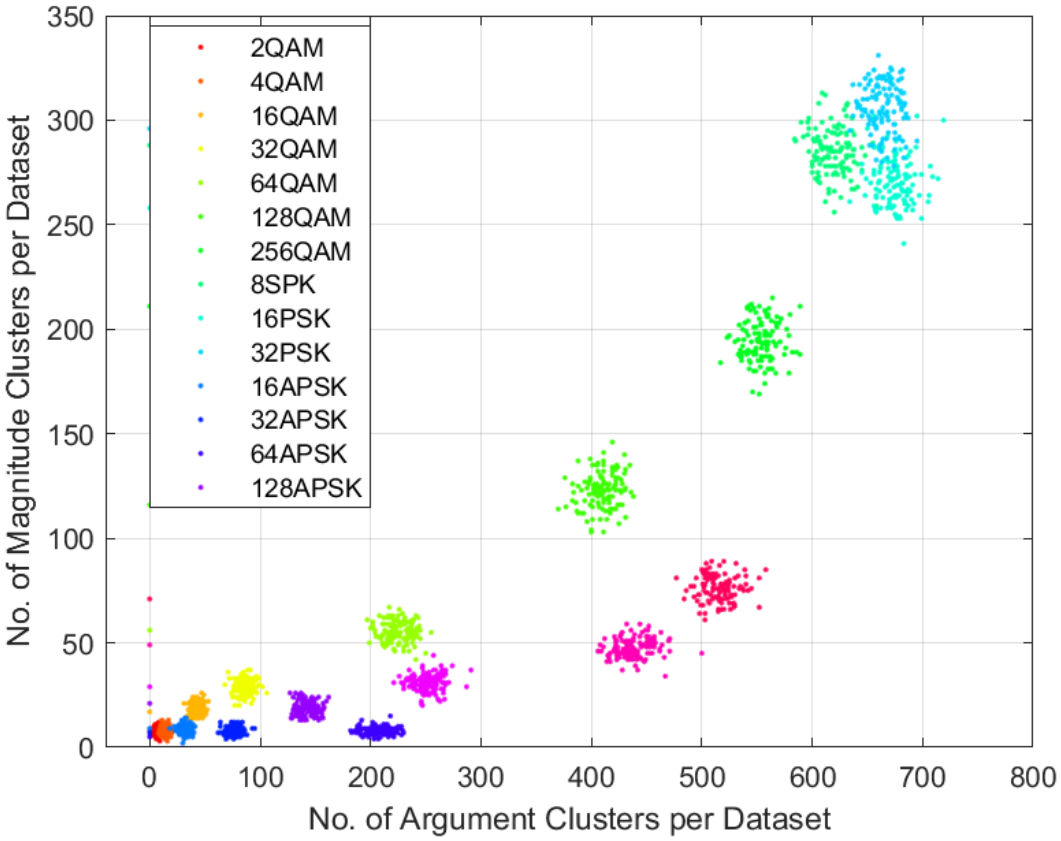


Figure 6.1: *An Example of Feature Space at 30dB SNR with Ideal Hyperparameter Selection*

The number of argument clusters and the number of magnitude clusters are each themselves a feature thus there are two features extracted with the DBSCAN feature extractor. The two features can be plotted on the Cartesian plane, an example of this is shown in Figure 6.1.

Figure 6.1 shows the resulting feature space of a well-tuned feature extraction system, the data used to obtain the displayed plot was at an SNR of 30dB. The obtained number of argument clusters is represented by the X-axis value and the obtained number of magnitude clusters is represented by the Y-axis value, the values combine to form clusters in feature space which represent various modulation schemes, each denoted by a different colour.

What this collection of clusters represents is a mechanism of determining how the extracted features relate to the class, or modulation scheme, to which they belong, the relationship between the cluster locations and their class can be utilised to train a classifier and therefore achieve classification of future inputs. It is therefore important that there is strong separation between the various clusters, clusters which overlap introduce risk that the classifier will be unable to differentiate between them. In Figure 6.1 the three highest order QAM clusters exhibit a slight overlap, similarly, 2QAM and 4QAM also exhibit some degree of overlap. In these cases a small amount of accuracy will be lost.

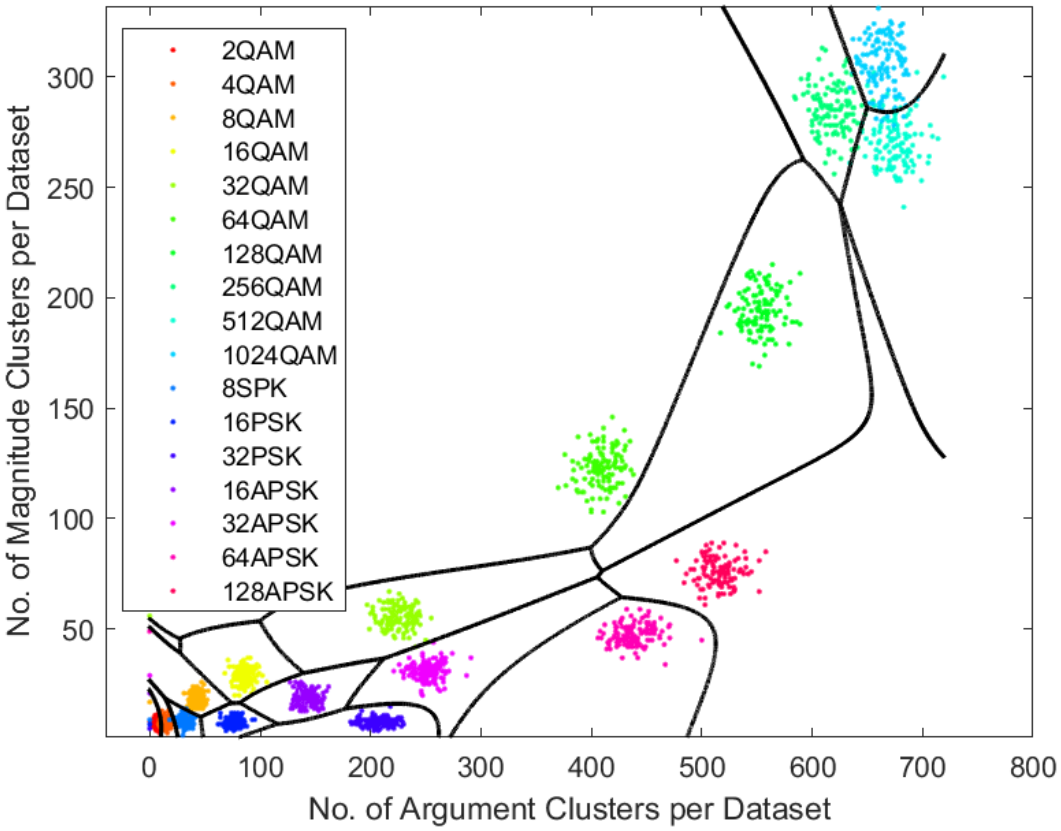


Figure 6.2: *An Example of the Learned Decision Boundaries at 30dB SNR with Ideal Hyperparameter Selection*

Figure 6.2 shows the decision boundaries learned by an MLP classifier on the feature space shown in Figure 6.1. Classes which have strong separation all have datapoints which lie within their own decision region, those classes will be classified with 100% accuracy. 256QAM, 512QAM, and 1024QAM on the right of the plot each have some points which cross into another classes decision region, these points will be misclassified, reducing the overall accuracy of the system.

This makes clear that the goal of the feature extraction tuning is to maximise the separation between class clusters to ensure that the maximum number of datapoint lie within their own decision region.

The hyperparameters utilised to obtain the feature set shown in Figure 6.1 and 6.2, were a dataset size of 5000, a *minPts* value of 2, an argument ε value of 0.1, and a magnitude ε value of 0.0007. This set of hyperparameters is optimal for this dataset size and modulation scheme set only, varying the dataset size and set of modulation schemes necessitates new ε value be selected, the following section explains how dataset size and choice of modulation scheme set relates to the systems accuracy.

6.2 DBSCAN Dataset Size

A crucial choice when classifying modulation schemes is the choice of how many datapoints are required to achieve a strong feature extraction and thus an accurate classification result. The dataset size is the only hyperparameter which is not contingent upon the values of the other hyperparameters, therefore it should be the first to be set and its value should inform the value of the remaining hyperparameters.

6.2.1 The Lower Bound of the Recommended Dataset Size

In general, the dataset size should be kept to a minimum because large datasets will introduce more latency as more data is required to be processed. Furthermore, it was shown in Chapter 5 that larger datasets require larger hardware implementation sizes and thus increased power consumption due to the requirement to increase the size of the sorting unit. Conversely, the dataset size needs to be large enough to allow for accurate feature generation. The minimum number of datapoints required to achieve accurate feature extraction may be given by the formula in Equation 6.1.

$$\text{Datapoints} \geq \mathit{minPts} \times \max(\text{No. of Argument clusters}, \text{No. of Magnitude clusters}) \quad (6.1)$$

Equation 6.1 states that the number of datapoints used for feature generation should be at least as large as *minPts* multiplied by the larger of either the expected number of argument or magnitude clusters of the highest order modulated signal in the employed dataset. The high modulation order of 1024QAM, with its corresponding expected number of argument and magnitude clusters, provides a representative worst-case scenario. The expected number of argument and magnitude clusters to be extracted from 1024QAM data is 847 and 109 respectively. The higher value of these two features is 847 which corresponds to the expected number of argument clusters found by DBSCAN. Recall that the *minPts* hyperparameter sets the minimum number of local points to constitute a cluster forming, should *minPts* be set to a value of 2 then at least double the expected number of argument clusters would be required as a dataset size to enable the DBSCAN algorithm to find the desired number of clusters.

Not allowing ample dataset sizes to facilitate accurate cluster generate leads to higher order modulation schemes overlapping with lower order modulation schemes in the feature space. Figures 6.3 and 6.4 illustrate this with examples of the obtained clusters in feature space when QAM orders 2 to 1024 at an SNR of 40dB are used for feature extraction with a dataset size of 5000 and 1000 respectively.

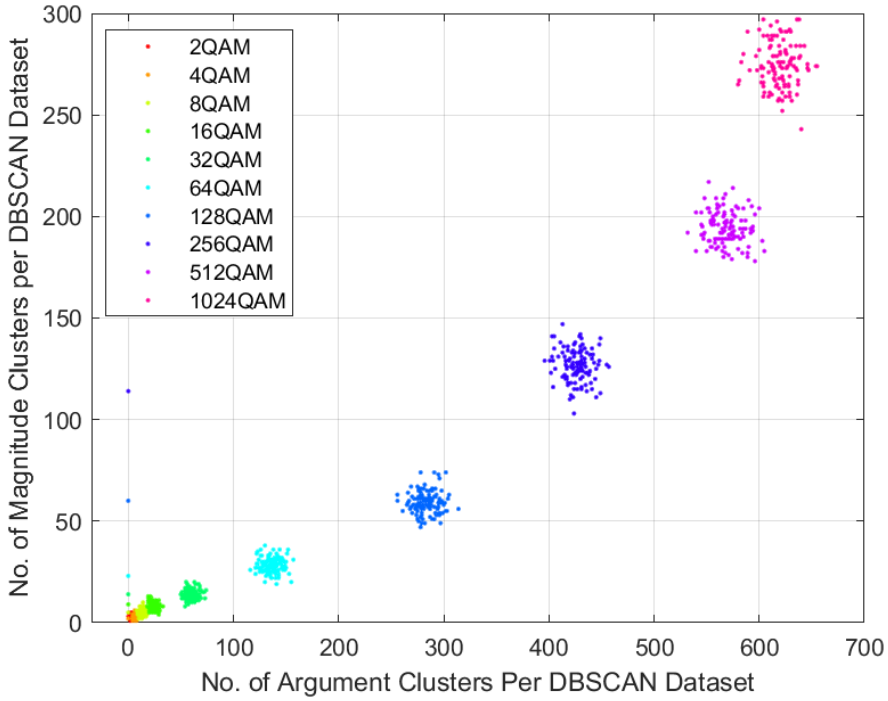


Figure 6.3: *An Example of Feature Space when the Dataset Size and Hyperparameters are Tuned Correctly*

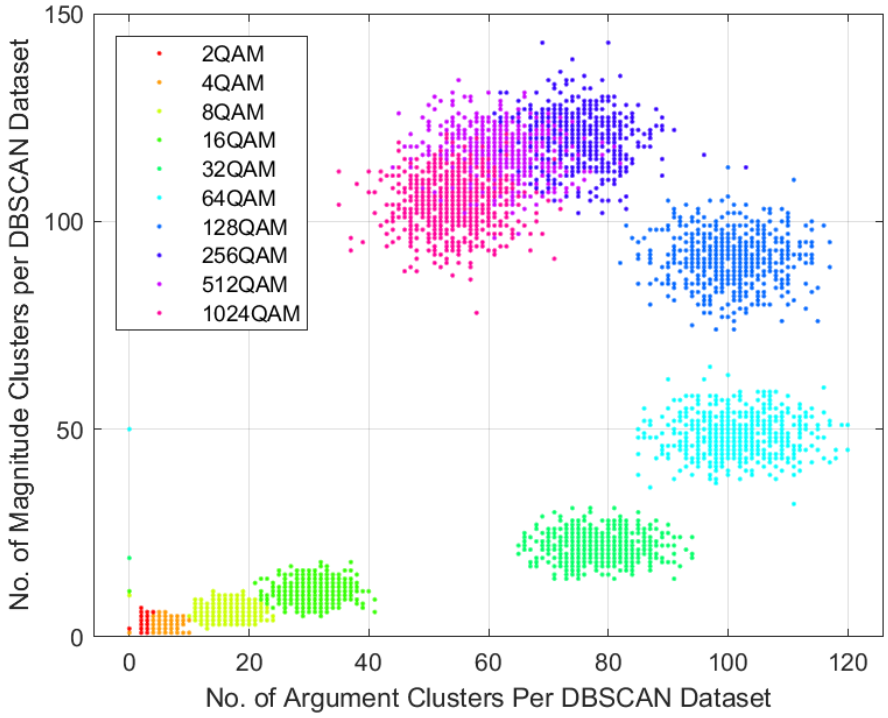


Figure 6.4: *An Example of Feature Space with a Dataset Size of 1000 and Hyperparameters Tuned for a Size of 5000*

Figure 6.3 shows the class clusters in feature space when a DBSCAN dataset size of 5000 is used for feature generation. Inspection of the 3 highest order classes shows strong separation and a growth in feature value which is characteristic of how the number of arguments and magnitudes increases as the modulation order increases. Conversely, Figure 6.4 was generated with a dataset size of 1000 and the correct *minPts* and ε for a dataset size of 5000. The 3 highest order classes show significant overlap in the feature space, in addition, the value of the obtained number of argument classes has decreased and is lower than that of 128QAM. This is because the dataset size is not large enough to allow for enough data to enable the DBSCAN clustering algorithm to find a characteristic number of argument clusters for these high-order modulation schemes.

6.2.2 Dataset Size and Feature Space Scale

When comparing Figures 6.3 and 6.4 it can also be seen that the value of the generated features decreases as the dataset size is reduced. For example, the feature cluster representing 128QAM has a midpoint of (275,60) in 6.3 but a midpoint of (100,80) in (b). Equation 6.1 showed that the minimum dataset size to obtain generated features which accurately find the correct number argument and magnitude clusters is equal to *minPts* multiplied by the larger of the expected number of argument or magnitude clusters. However, this relationship assumes that modulated symbols are all transmitted with an equal occurrence rate, in practice this is unlikely. For example, take the previous example of 1024QAM with an expected number of arguments of 847, it was said that the minimum dataset size of 1694 would allow for *minPts*=2 datapoints per argument to be obtained, therefore leading to the expected number of argument features being found. However, it is unlikely that every symbol will be transmitted exactly twice in a size 1694 sample, symbols which are not transmitted at least *minPts* times within a dataset do not result in a cluster being found at that particular argument, reducing the number of argument clusters found. Therefore, further increases to the dataset size increase the likelihood that symbols corresponding to each argument value are transmitted at least *minPts* times within said dataset. Therefore, increasing the DBSCAN feature generation dataset size beyond the minimum required value allows for feature generation closer to the expected values.

How this phenomenon influences the resulting classification accuracy may be explained in terms of the size of the feature space and the resultant separation between classes. Inspecting the X and Y-axis of Figures 6.3 and 6.4 shows that the scale of the axis in 6.4 is smaller in comparison to 6.3, this reduces the scale of the feature space which leads to feature clusters which are more densely spaced. This problem is less pronounced at high SNRs, as was the case when generating these figures, but when the SNR is decreased the reduced feature space can result in decreased accuracy due to overlapping feature clusters. For example, Figures 6.5 and 6.6 show two feature spaces of features extracted from QAM signals with an SNR of 15dB, 6.5 used a dataset size of 5000, 6.6 used a dataset size of 1000.

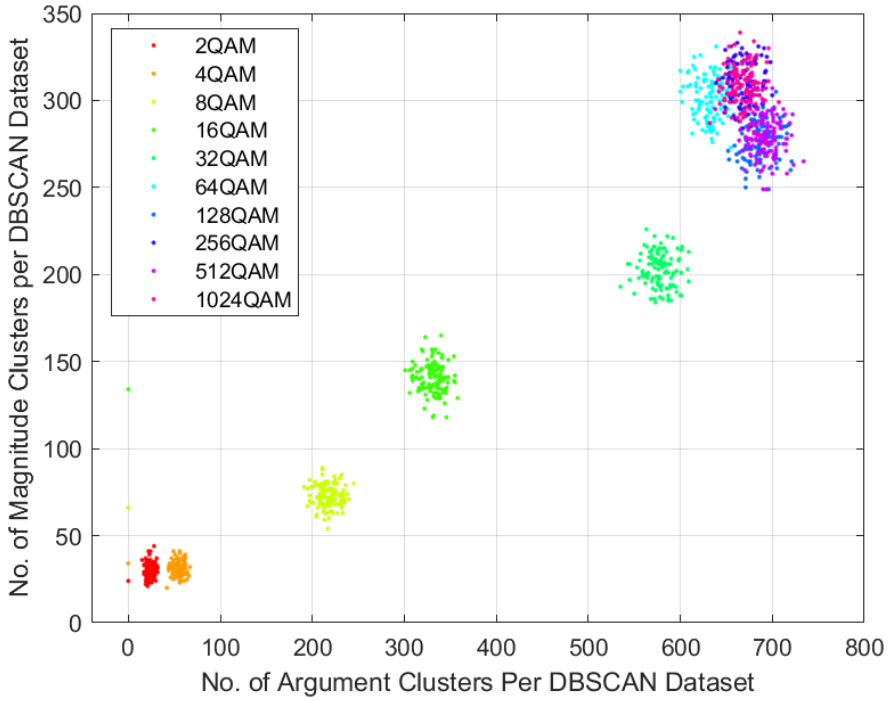


Figure 6.5: *An Example of Feature Space at 15dB SNR when the Dataset Size is set to 5000*

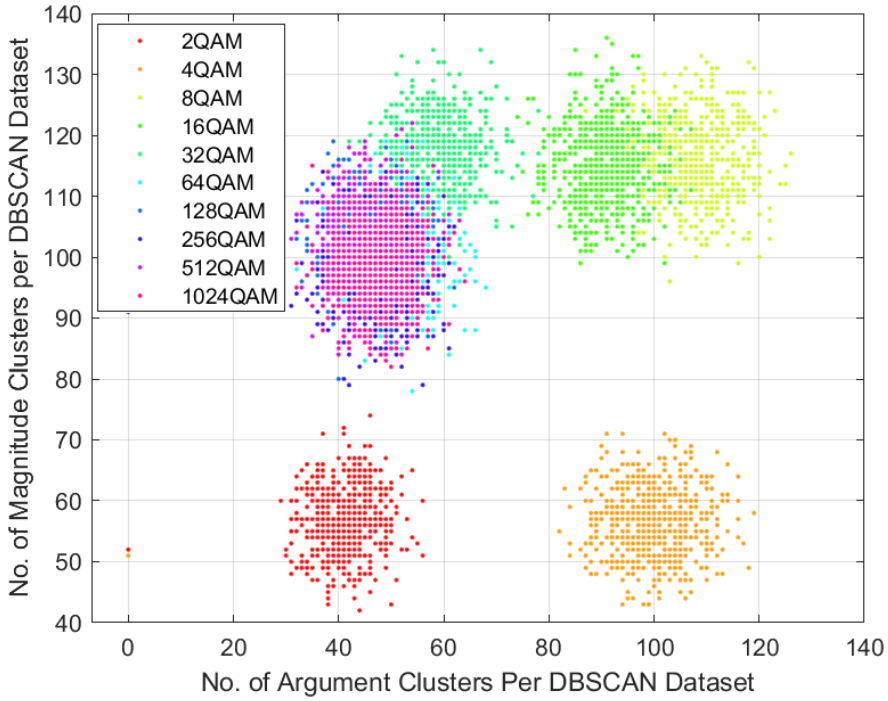


Figure 6.6: *An Example of Feature Space at 15dB when the Dataset Size is set to 1000*

In the examples shown in Figures 6.5 and 6.6 the feature spaces are once again vastly different in scale. Figure 6.5 shows feature clusters which have a large separation and appear more compact, although this compactness is primarily due to their relative scale in comparison to the scale of the feature space. Alone, this relative compactness and large separation will result in a greater average classification accuracy in comparison to 6.6. There is also a higher degree of overlap between feature clusters in Figure 6.6 due to the feature space not being sufficiently large enough to allow for accurate characterisation of the data.

This may be seen most clearly by comparing the feature clusters of 8QAM, 16QAM, and 32QAM. In Figure 6.5 there is a high degree of separation between these clusters, yet in Figure 6.6 the limited feature space causes feature values to be generated inaccurately, once again causing a curl towards the y-axis. The result is that the clusters representing 8QAM and 16QAM begin to overlap, and the 32QAM cluster intersects the high order supercluster. Each case will result in a reduction in classification accuracy due to the inability to separate each feature cluster entirely with decision boundaries.

In the context of the proposed system, the DBSCAN feature generation dataset size can therefore be thought of as setting the size of the feature space, larger values increase the size of the feature space which facilitates greater separation between feature clusters as well as providing the required feature space for feature values to be generated closer to the expected value.

6.2.3 The Relationship Between DBSCAN Dataset Size and Classification Accuracy

To demonstrate the effects of various dataset sizes have upon the classification accuracy, Figure 6.7 was created which displays the classification accuracy achieved on orders of QAM from 2 to 1024 at 40dB SNR across dataset sizes from 50 to 3000.

With a dataset size of 3000 all modulation schemes are classified with 100% accuracy. At a DBSCAN dataset size of 2000 the accuracy of these two modulation falls to 97.8%, below this dataset size the accuracy of these classes continues to decrease in general. At 1500 256QAM sees its first decrease in accuracy from 100% and continues to decrease as the dataset size decreases. Notably in Figure 6.7 it can be seen that with extremely low dataset sizes even the lowest order modulation schemes begin to decrease in accuracy. At a size of 250 there are no modulation schemes with 100% accuracy, at 100 only 4QAM is classified with above 80% accuracy. At a DBSCAN dataset size of 50 there are no classes classified with an accuracy greater than 90% accuracy, the majority of classes show under 70% accuracy.

Figure 6.7 clearly exhibits the relationship between the DBSCAN dataset size and the obtained classification accuracy, for a particular *minPts* and ε value, increasing the dataset size in general increases the classification accuracy achieved by the proposed system. Figure 6.7 shows that in this case 100% accuracy is achieved with a dataset size of 3000, this demonstration is purely to illustrate to the reader how increasing dataset sizes provide additional classification accuracy. The signals used to obtain the results shown in this figure were of an SNR of 40dB. When using signals of a lower SNR it was found that the system benefitted from further increases in dataset size, up to a value in the region of 5000, beyond which negligible increases were obtained. As such, Chapter 7 presents results when a dataset size of 5000 is employed to demonstrate DBMC top end performance.

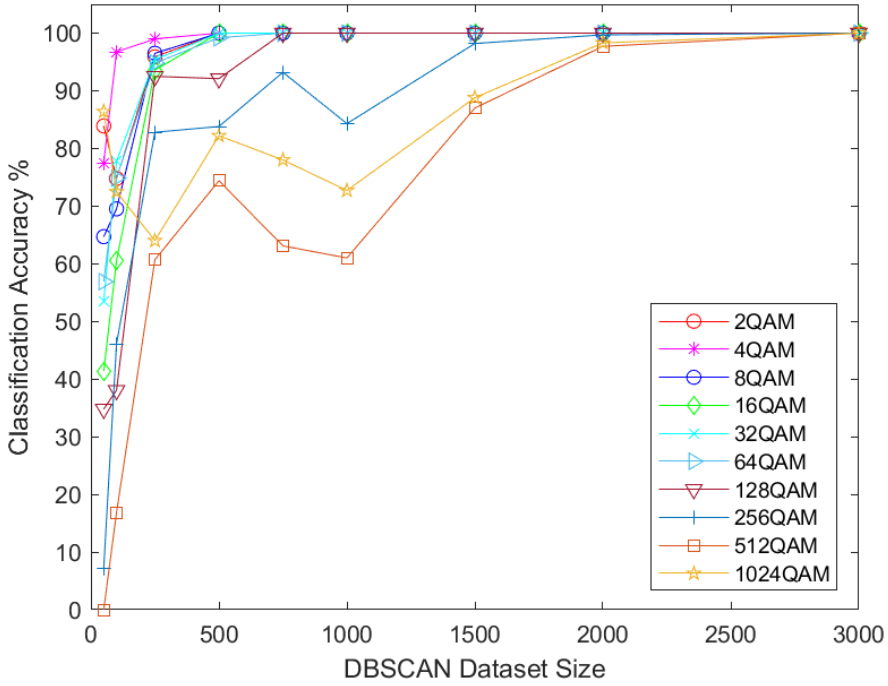


Figure 6.7: *How the Classification Accuracy (%) changes as the DBSCAN Dataset Varies*

6.2.4 Recommendations for the DBSCAN Dataset Size

This investigation into the effects of varying the DBSCAN dataset size has shown that its primary effect is setting the size of the feature space. Larger feature spaces enable greater separation of feature clusters, enable feature generation with values closer to the expected values, and reduce the curling towards Y-axis distortion. However, larger DBSCAN dataset sizes also increase the required FPGA utilisation and latency of the proposed hardware structure. A balance must therefore be found between large implementation sizes and the achieved classification accuracy. What has not been shown thus far in this discussion is that when the dataset size is small, ε may be tuned to maximise the spacing of feature clusters. Figure 6.8 demonstrates this by showing a similar example to Figures 6.5 and 6.6, in this case the DBSCAN dataset size was 1000 but the ε values were tuned specifically for a DBSCAN dataset of 1000. It can be seen that despite the smaller feature space there is still a degree of separation between feature clusters which is stronger than seen in Figure 6.6. Although it is the case that the separation is weaker than in Figure 6.5.

For the purposes of the work proposed in this thesis it was decided to use the minimum dataset size provided by equation 6.1 and tune the ε values to maximise performance for that specific size. If an engineer was looking to maximise the performance of the proposed DBSCAN feature extractor and they were not limited by hardware implementation costs, then it would be advantageous to maximise the DBSCAN dataset size to achieve stronger accuracy. This thesis will provide software results with a dataset size of 5000 to demonstrate the upper limits of performance.

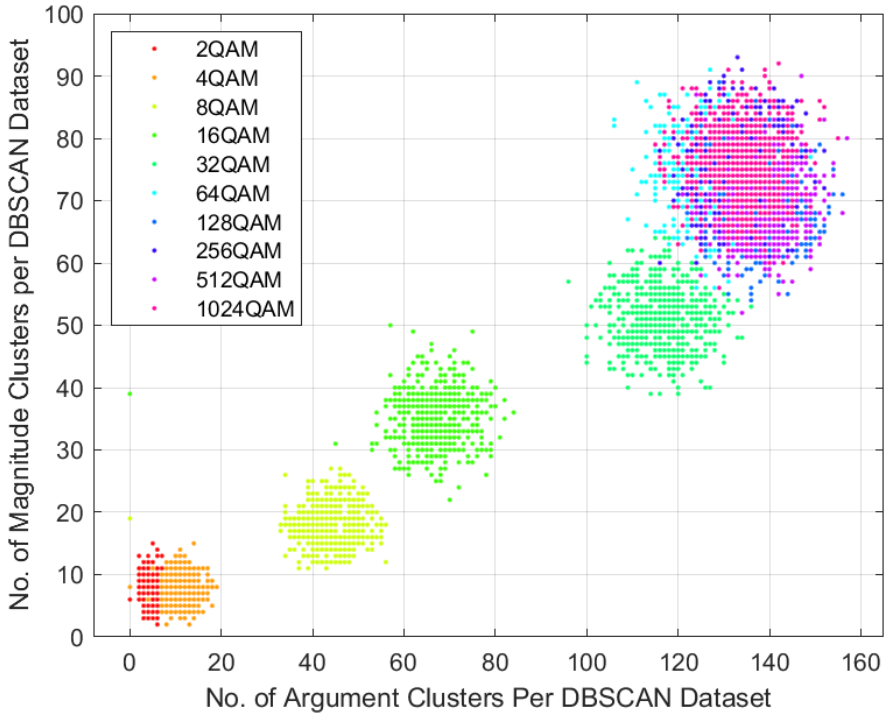


Figure 6.8: *The Resulting Feature Space at an SNR of 15dB and a Dataset Size of 1000 with ε Values Correctly Tuned*

6.3 The *minPts* Hyperparameter

The creators of the DBSCAN algorithm proposed two rules of thumb for finding strong values of *minPts* [34]. The first is that the lower bound of *minPts* should be at least the dimensionality of the data plus 1, shown by Equation 6.2.

$$\text{minPts} \geq D + 1 \tag{6.2}$$

Where D is the dataset dimensionality. Sander et al. also suggest that a strong *minPts* value can be obtained by multiplying the dimensionality of the data by 2 [116], shown by Equation 6.3.

$$\text{minPts} = D \times 2 \tag{6.3}$$

As the datasets used for feature generation in the proposed system are both 1D, a *minPts* value of 2 satisfies both equations as it is equal to the suggested lower bound and is the result of multiplying the dimensionality of the data by 2. Ultimately, a *minPts* value of 2 was employed in all configurations of the proposed system as it not only satisfied these rules of thumb but also enabled a minimisation of the DBSCAN dataset size which in turn reduced the total implementation size. Before deciding upon the optimum *minPts* value suggested by the above equations, additional investigations upon the effects of varying *minPts* were conducted to confirm that additional accuracy could not be gained from increasing the value of this hyperparameter, these investigations are shown in the remainder of this section.

6.3.1 Investigations into the Effects of *minPts*

Equation 6.1 showed that the optimal value for the DBSCAN dataset size is proportional to the *minPts* value. It has also been shown that that increasing the DBSCAN dataset size increases the required FPGA utilisation and power consumption, therefore selecting a value of *minPts* which is as small as possible enables smaller implementation sizes. According to this relationship, it follows that increasing the value of *minPts* will have the same effect on classification accuracy as reducing the DBSCAN dataset size, this relationship is confirmed by the graph shown in Figure 6.9 which was created from modulated signals at 40dB SNR and uses a DBSCAN dataset size of 3000.

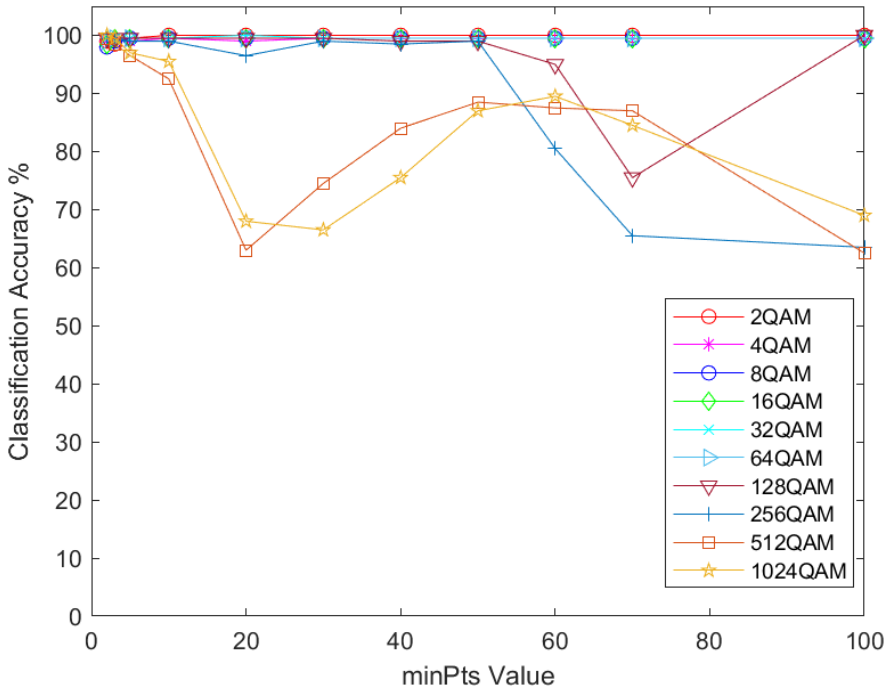


Figure 6.9: *How the Classification Accuracy (%) changes as the minPts Value Varies*

In Figure 6.7 it was shown how increasing the DBSCAN dataset size increases the accuracy achieved by the AMC system, higher order modulation schemes in particular required a larger dataset size to reach 100% classification accuracy, whereas all of the orders of QAM below 128 were classified with 100% accuracy with a dataset size as low as 500. The results shown in Figure 6.9 demonstrate a similar relationship, the higher order modulation schemes exhibit a reduction in classification accuracy as the *minPts* value increases. The modulation schemes 512QAM and 1024QAM are the first to see a reduction in accuracy from 100% at a *minPts* value of 5, 128QAM and 256QAM see a significant reduction at a value of 60. The results shown in Figure 6.9 confirm that the optimum value for *minPts* in this case is 2 as this is the point where the average classification accuracy is highest.

The relationship between *minPts* and the DBSCAN dataset size can also be seen by inspecting the clusters in feature space. Figure 6.10 shows the resulting feature clusters when a DBSCAN dataset of size 3000 in conjunction with a *minPts* value of 10 is used for feature

generation.

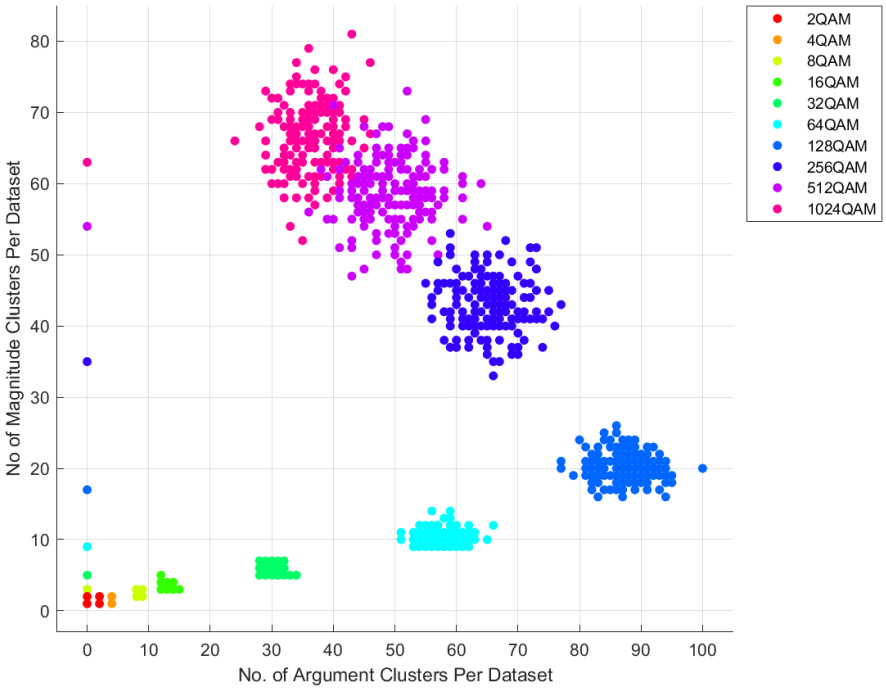


Figure 6.10: *The Appearance of Feature Space when a minPts Value of 10 is used in Conjunction with a Dataset Size of 3000*

Figures 6.5 and 6.6 in the previous section showed that reducing the dataset size causes DBSCAN output clusters to curl inwards towards the Y-axis. The same trend can be observed in Figure 6.10 when the DBSCAN dataset size was set to 3000 but *minPts* was set to 10, the 3 highest order classes are not able to be characterised accurately by the feature extraction algorithm and thus their feature clusters move towards the Y-axis. In both cases this phenomenon is caused by there being insufficient data to allow for accurate characterisation of the input modulation data, either due to an insufficiently large dataset size or a *minPts* value which is too large for an employed dataset.

The final point to highlight with regards to *minPts* is that the results shown in Figure 6.9 do not show nearly as much accuracy degradation of the lower order classes as was seen in Figure 6.7. The largest *minPts* value which was used to create Figure 6.9 was 100, according to Equation 6.1 this should provide equivalent results to using a *minPts* value of 2 and a dataset size of 60. In Figure 6.7 nearly all modulation schemes exhibited a classification accuracy of less than 100% with this dataset size and yet in Figure 6.9 the lower order signals are classified with an accuracy of 100% across all *minPts* values. This is thought to be due to the robustness introduced to the clustering process by using high values of *minPts*. Higher values of *minPts* provide increased stringency for the clustering process as a greater number of local points are required before a cluster forms, this increases the noise tolerance of the algorithm and in turn leads more compact feature clusters in feature space. This can again be observed by comparing Figure 6.4 and Figure 6.10 where the low order class clusters exhibit a distinct difference in dispersion, ultimately providing greater feature separation and therefore

increased accuracy.

Because *minPts* has the property of enabling denser feature clusters to be generated, there is potential for higher values of *minPts* to be utilised with extremely large DBSCAN dataset sizes in order to perhaps obtain greater classification performance. However, as the proposed work is constrained by hardware limitations, this avenue of research is beyond the scope of this thesis. Lower values of *minPts* allow for minimising the DBSCAN dataset size, and as the goal of this proposed work is to create an AMC system which requires as few resources as possible, it was ultimately decided to use a value of 2. Therefore, henceforth in this thesis all provided results were obtained using this value.

6.4 The ε Hyperparameter

The final parameter which must be chosen to operate the DBSCAN modulation classifier is ε, the maximum distance between datapoints for them to be classified as in the same cluster. It was briefly shown in Section 6.2 that ε may be used a means to tuning the spacing of the feature clusters in feature space to maximise separation. Values of ε which maximise feature cluster separation therefore are optimal as this in turn maximises the obtained classification system classification accuracy. Within the proposed system there are two DBSCAN feature extraction cores, each operating on either the magnitude or argument data, therefore 2 strong epsilon values must be found per DBSCAN dataset size.

Determining optimal values for ε presents a significant challenge. The optimal ε is highly sensitive to dataset size, the SNR of the signal data, and the employed set of modulation schemes. The dataset size is a parameter which is set depending upon the DBMC configuration, however each configuration must be capable of classifying a range of modulation schemes at various SNRs. Consequently, a compromise must be identified that maximizes performance across a diverse range of SNRs and modulation schemes for a given dataset size.

This section investigates the variability of optimal ε values across different SNRs and modulation formats. Initially, the widely used 'Elbow point method' is employed to demonstrate the process of determining optimal ε values. Through its usage the subjectivity of this method and its reliance on human intervention are made clear. To address these limitations, an automated heuristic approach is developed and proposed. Finally, this section demonstrates the superior performance of the proposed heuristic in identifying robust ε values compared to the elbow point method, while simultaneously minimizing the human input required.

6.4.1 The k-Distance Graph and the Elbow Point Technique

The analysis in the following section requires that the k-distance graph is first understood. The k-distance graph is the recommended technique for finding strong values of the ε hyperparameter [34]. It may be constructed by calculating the distance from each datapoint in a dataset to its k-nearest neighbours. These distances are then sorted in descending order and plotted to form a curve. The value of k should be set to the employed value of *minPts* minus 1. An example 4-distance graph for QPSK can be seen in Figure 6.11.

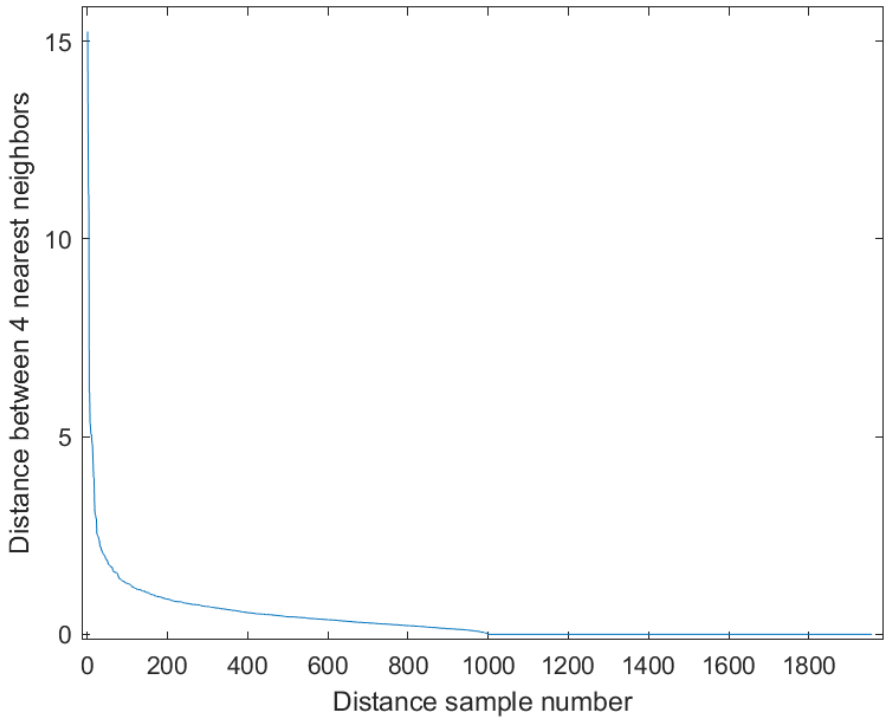
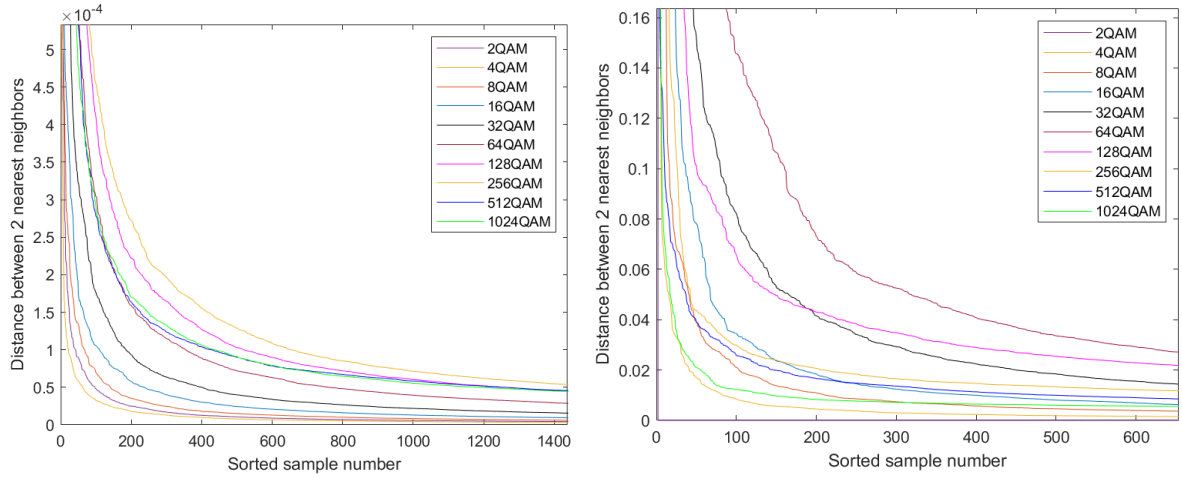


Figure 6.11: *An Example 4-Distance Graph of QPSK at an SNR of 30dB and Dataset Size of 5000*

The graph provides a visual representation of the distances between datapoints, as the sample number increases, the distance between points decreases. At an X-axis value in the region of 75 there is a sharp bend in the shape of the 4-distance graph, this bend is referred to as the 'Elbow Point'. It could also be defined as the point on the curve where the rate of change of the gradient is at a maximum. Datapoints to the left of the elbow point generally represent the distances between clusters, points to the right represent distances between datapoints within a cluster. Theoretically, the Y-axis value at the elbow point provides a strong value for ε as it is a distance which is greater than the distances between cluster points, but less than the distances between clusters.

6.4.2 How ε Varies with SNR and Modulation Format

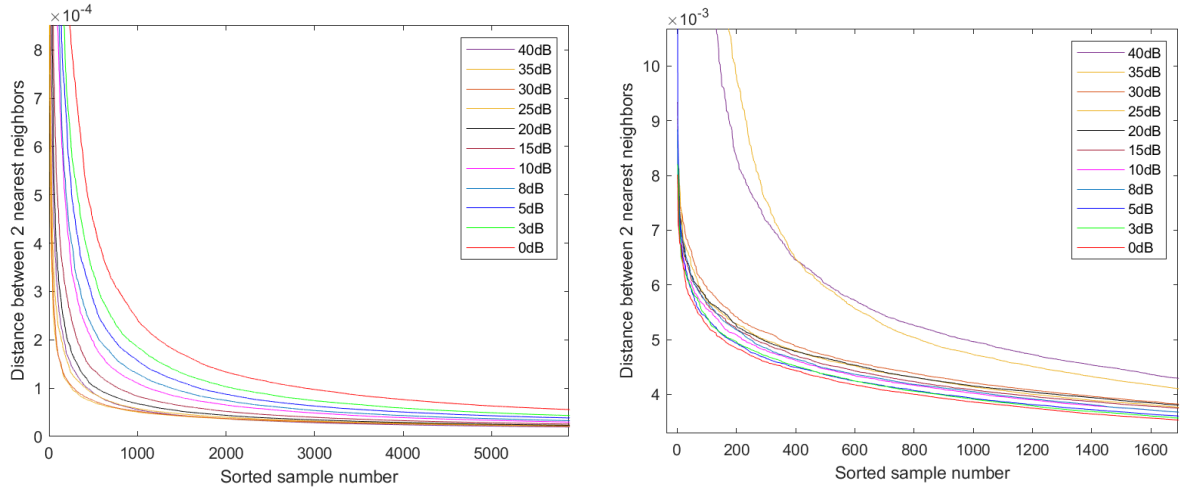
This classification system is designed to differentiate between a set of modulation schemes across a range of SNRs, this introduces a problem which is that each modulation scheme has an optimal ε value to achieve accurate clustering. The 1-distance graph of magnitudes and arguments of various QAM constellation diagrams at 30dB SNR with a dataset size of 5000, can be seen in Figure 6.12 (a) and (b) respectively.



(a) *Magnitude Data 1-Distance Graphs at 30dB SNR* **(b)** *Argument Data 1-Distance Graphs at 30dB SNR*

Figure 6.12: *How the 1-Distance Graphs Vary Between QAM Modulation Orders at an SNR of 30dB*

In both figures, each line represents a different modulation scheme's 1-distance graph using datasets with a SNR of 30dB, it can be seen that the elbow points for each modulation scheme differs. The argument elbow points differ by a larger degree than the magnitude, with a range of 0.015 to 0.06 as opposed to the range of 0.4×10^{-4} to 2×10^{-4}. This problem of not having a single optimum value for ε also is true for differing SNRs, Figure 6.13 (a) and (b) show the 1-distance graphs for the absolute value and arguments of a 128QAM constellation diagram at SNRs from 40dB to 0dB.



(a) *Magnitude Data 1-Distance Graphs of 128QAM at SNRs in the Range 0dB to 40dB*

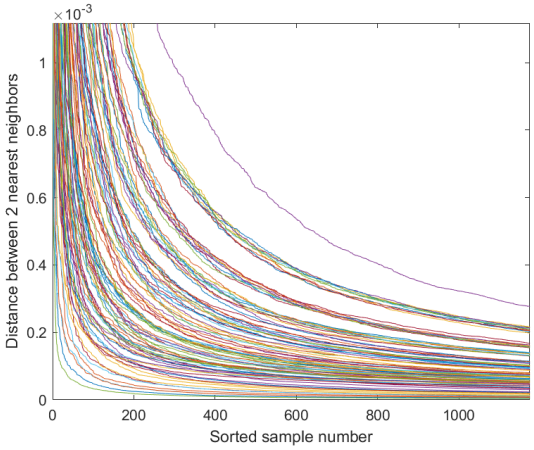
(b) *Argument Data 1-Distance Graphs of 128QAM at SNRs in the Range 0dB to 40dB*

Figure 6.13: *How the 128QAM 1-Distance Graphs Vary Between SNR*

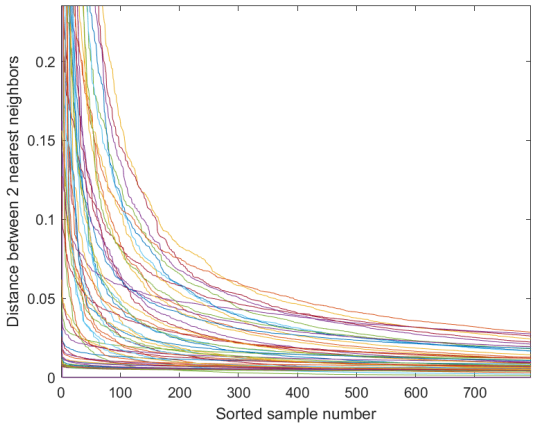
Just as how elbow points varied across each modulation scheme, the SNR introduces further variation in the optimum ε, with the absolute values having an optimum ε in the range of

2×10^{-4} to 1×10^{-4} and the arguments having an optimum value ranging from 5×10^{-3} to 6.5×10^{-4}.

By combining the curves of all modulation schemes at all SNRs the true range of ideal ε can be seen, this is shown in Figure 6.14 (a) and (b).



(a) *Magnitude Data 1-Distance Graphs across Various QAM Modulation Orders at SNRs 0dB to 40dB*



(b) *Argument Data 1-Distance Graphs across Various QAM Modulation Orders at SNRs 0dB to 40dB*

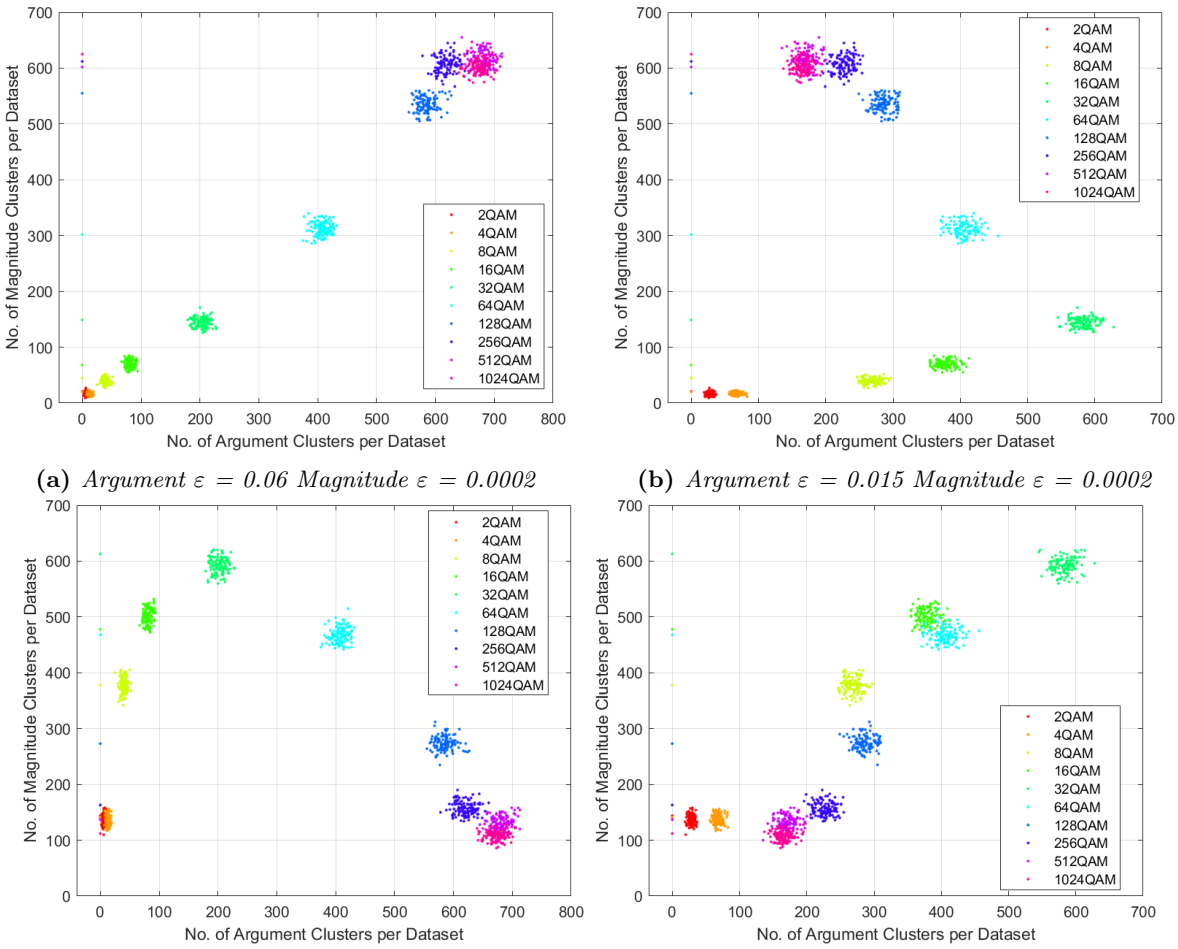
Figure 6.14: *How the 1-Distance Graphs Vary Across All SNRs and Modulation Schemes*

These figures illustrate the true range of optimal ε across the full QAM dataset. No legend is provided for these figures due to the number of curves, the exact modulation scheme and SNR which each curve represents is not important to know, the main takeaway should be that there is a large range of elbow points to choose from. The following section will discuss how designers should select a value of ε to achieve performance across this entire range.

6.4.3 Finding Strong ε Values for a Particular SNR

Figure 6.14 (a) and (b) in the previous section showed the 1-distance graphs for each modulation scheme at each SNR. It was found that the optimum ε value varies with both the SNR and modulation scheme. Because of the significant variability it is impossible to find a single ε value which enables the clustering system to find the exact expected number of clusters in every case. The challenge therefore becomes finding a pair of ε values which maximise feature separation is the largest number of cases, therefore achieving the highest classification accuracy on aggregate.

Figure 6.14 (a) showed that the low-order modulation schemes required a smaller magnitude ε value than the high-order modulation schemes to achieve accurate characterisation. Figure 6.14 (b) found that both the high-order and low-order modulation schemes benefit from a lower argument ε value, whereas the middle-order modulation schemes were found to require a higher argument ε value. To demonstrate how selecting different combinations of ε values result in the feature extraction system's ability to characterise the dataset, an example of how values taken from either extreme of the optimum ε range shown in Figure 6.14 (a) and (b) effects the resulting feature space is provided in Figure 6.15.



(a) *Argument ε = 0.06 Magnitude ε = 0.0002*

(b) *Argument ε = 0.015 Magnitude ε = 0.0002*

(c) *Argument ε = 0.06 Magnitude ε = 0.00004*

(d) *Argument ε = 0.015 Magnitude C = 0.00004*

Figure 6.15: *How the Arrangement of Feature Clusters Varies with C Values Chosen from Either Extreme of the 1-Distance Graph Optimum Range*

Figure 6.15 shows that the selection of the ε values results in significant changes to the distribution of feature clusters in feature space. Figure 6.15 (a) was created using ε values at the highest elbow point value for both the arguments and magnitudes. The arrangement of feature clusters follows a linear growth as the modulation order increases. Figure 6.15 (d) conversely was created using ε values found by taking the smallest elbow point values in the distribution, in this case there is a linear growth up until 64QAM, at modulation orders greater than this the feature clusters double back and begin to approach the origin. In this case there is a severe risk of feature cluster overlap being introduced, it can be seen that the clusters of 32QAM and 128QAM do indeed exhibit slight overlap which is not the case in any other example.

In the case of Figure 6.15 (b) and (c) there is 1 ε value taken from the maximum elbow point and 1 from the minimum. In a similar manner to what was observed in 6.15 (d), the number of clusters found by DBSCAN until 32QAM linearly increases, beyond this modulation scheme the trend of feature cluster positioning approaches either the X or Y-axis. Despite this, there is still minimal feature cluster overlap so strong performance would still

be expected, however this may not be the case at all SNRs.

It has generally found to be the case that utilising ε values which produce a predictable growth in magnitude and argument values between orders of the same modulation format has enabled higher degrees of optimisation across a range of SNRs. This is particularly important when PSK and APSK signals are included in the dataset as unpredictable relationships between modulation orders and the number of argument and magnitude found by DBSCAN can lead to feature cluster overlap where there otherwise would not be. Therefore, selecting ε values which are optimal for the highest order modulation scheme in the dataset allows for the characteristic linear increase between modulation orders shown in Figure 6.15 (a), therefore enabling stronger control of feature cluster arrangement.

6.4.4 Issues with Optimising ε Values Across SNRs

It has been demonstrated throughout this section how using the largest ε values in the optimum range enabled accurate characterisation of all modulation schemes at an SNR of 30dB. Figure 6.14 (a) and (b) showed that the optimum ε values change not only with the set of employed modulation schemes, but also with varying SNRs. To investigate the effects of various ε values at different SNRs, the optimum ε value pair at SNRs from 0dB to 40dB was found using the elbow point method, the two largest values in the optimum range were selected. The obtained values can be seen in Figure 6.16.

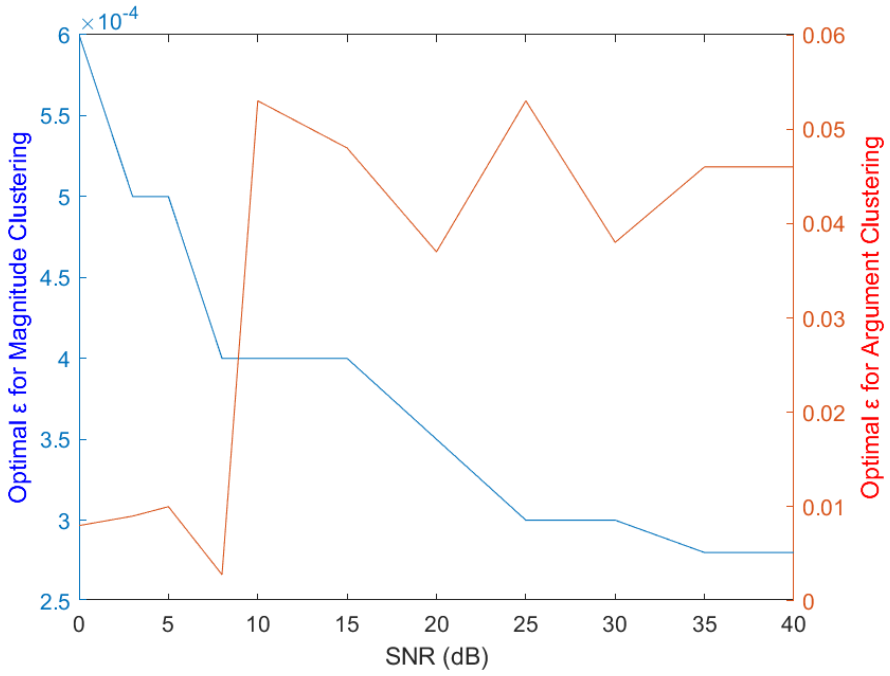


Figure 6.16: *How the Optimum ε Obtained via the Elbow Point Method Varies with SNR (dB)*

It is important to note that the values obtained to produce Figure 6.16 were obtained manually from inspecting 1-distance graphs, therefore the displayed values should be seen as estimates. In general, the optimum ε value for magnitude clustering is greater at lower

SNRs, conversely it is greater at higher SNRs for argument clustering.

Several classifiers were trained and tested using features extracted with various combinations of ε values which can be seen in Figure 6.16, additionally a classifier was trained using features extracted with the optimum ε pair at every SNR. The average classification accuracy against SNR results are displayed in Figure 6.17.

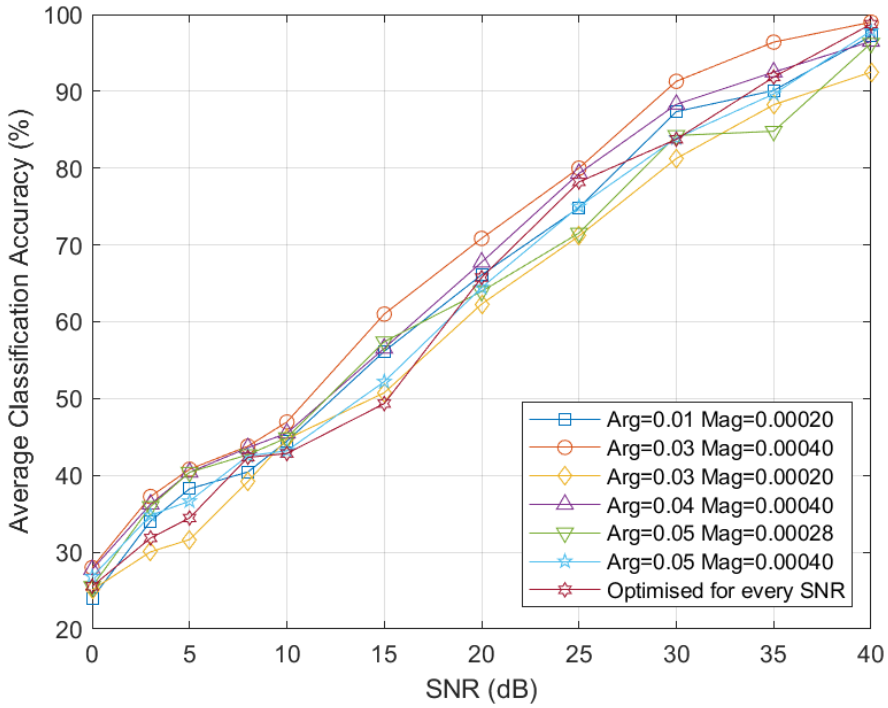


Figure 6.17: *The Average Accuracy Across QAM Orders 2 to 1024 Achieved by Utilising ε Combinations Shown in Figure 6.16*

The argument and magnitude ε pair which was found to be optimal at 10dB was also found to provide the highest accuracy at every other SNR. Furthermore, training the system on data extracted with ε pairs found to be optimal at every SNR in many cases resulted in an average accuracy which was worse than every other ε pair tested. This was contrary to the expected results, optimising for every SNR should provide the greatest accuracy at every SNR, the 10dB optimised ε should result in the strongest accuracy at 10dB and nowhere else. Following these findings, it was decided that employing the elbow point method is not a viable optimisation strategy for this system.

6.4.5 Issues with Using the Elbow Point Method

The analysis in the previous sections to find the optimum values for ε was performed using the elbow point method, which is suggested by the DBSCAN authors [34]. Figure 6.17 showed that when utilising the elbow point method to optimise the system for every SNR the accuracy which was achieved was in some cases lower than every other example. It is thought that this is the case due to the utilised ε values not being the actual optimum and thus the elbow point method is inadequate.

The elbow point method has been demonstrated to involve manual human interaction which inhibits the creation of automated DBSCAN classifiers as well as introducing uncertainty as the location of an elbow point may be subjective. The uncertainty is particularly pronounced when the elbow point does not lie upon a curve with an instantaneous change of gradient but rather a gradual change in gradient as in this case. Due to this curved nature, the elbow point can be seen as a range of values depending upon how the curve is viewed, for example the same 1-distance graph of 4QAM with a single point highlighted is shown Figure 6.18 in three different graphs.

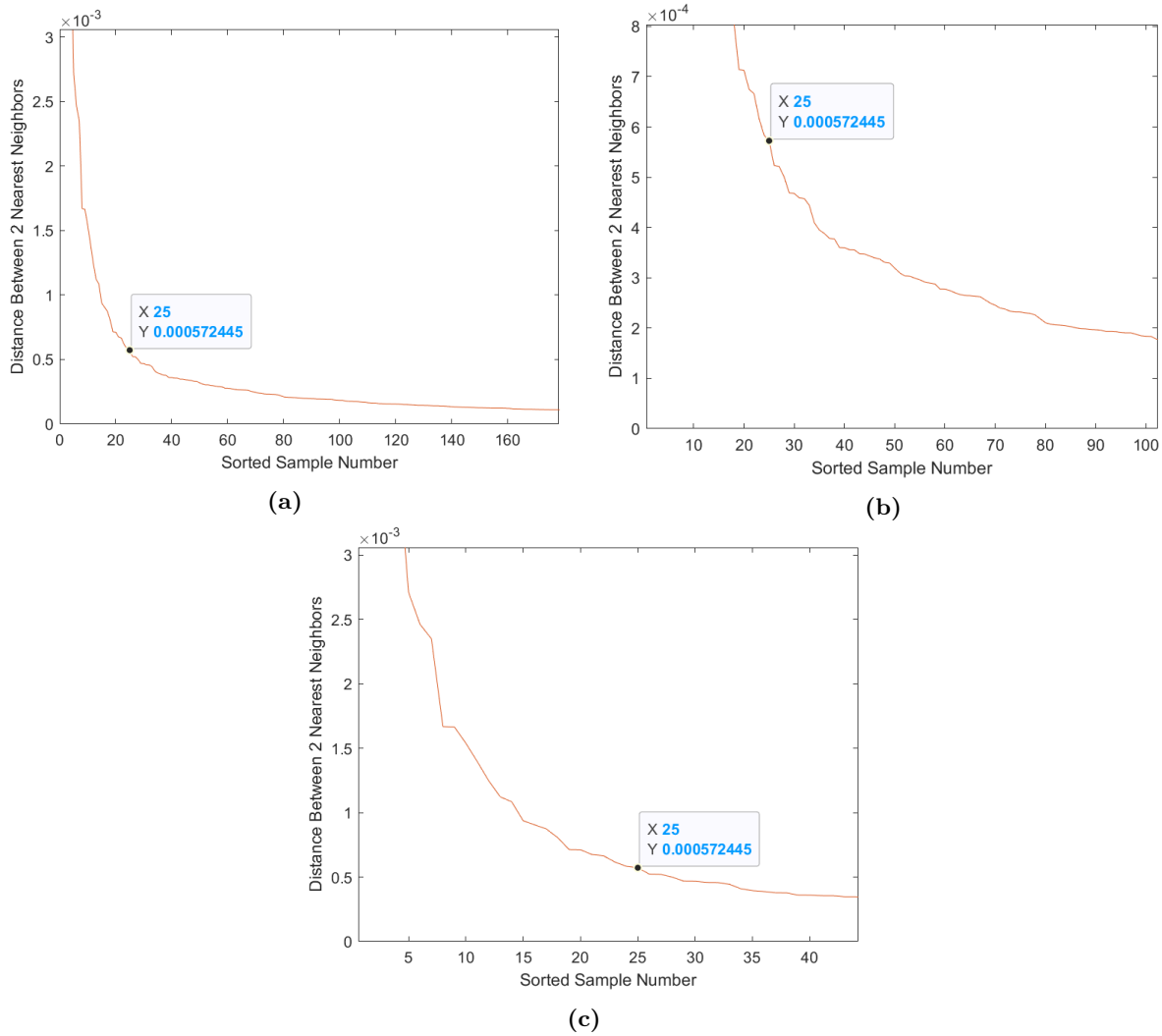


Figure 6.18: *How the X and Y-axis Scale Affects where the Elbow Point May Subjectively Appear*

Each figure above is of the same curve but with varying scales on the X and Y-axis, a point which could be considered the elbow point of Figure 6.18 (a) has also been highlighted in all figures. The scale of the axis can change where the elbow point is observed to lie as in Figure 6.18 (b) the highlighted point lies above what may be considered the elbow point

and conversely in Figure 6.18 (c) the highlighted point lies below what may be consideredthe elbow point.

To solve the problem of elbow point subjectivity, a trial an error approach could be employed in which a grid search algorithm takes a range of ε values which are manually input by the designer and iteratively performs feature extraction as well as classifier training and testing to find which combinations result in optimum performance. As there are 2 values to be optimised, the algorithmic complexity of this strategy quickly explodes, for a range of 10 different magnitude and argument ε values, the feature extraction and subsequent classifier training and evaluation must be performed 100 times to test each value. Furthermore, the subjectivity and requirements for human intervention are also not eliminated, even if a strong value was found, the selected range of points to be evaluated may not include the ε pair which would provide the strongest accuracy, the precision of the search may also not be fine enough to enable the optimal values to be found. Even further compounding the scale of this search is the fact that varying dataset sizes, *minPts* values, and employed sets of modulation schemes have different optimum ε values, meaning that this operation must be performed whenever each of these variables are modified.

It was therefore decided that the problem is best solved by approaching it from the opposite direction. Rather than testing ε pairs to find which values result in a good separation in feature space, feature space would be designed and ε values which resulted in said feature space could be found automatically.

6.4.6 The $d-1th$ Difference Value Method

When plotting the 1-distance graph the distances between points are sorted to generate the distinctive curve. If instead the distances are viewed unsorted, they appear as in Figure 6.19.

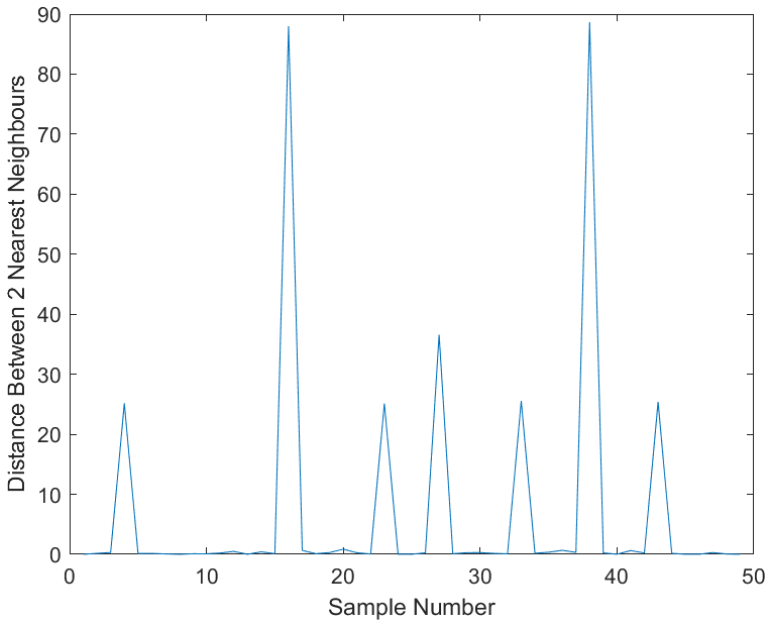


Figure 6.19: *The Unsorted 1-Distance Graph of the 8QAM Arguments*

Figure 6.19 shows the unsorted 1-distance graph of the arguments of 8QAM at 40dB SNR, the peaks represent the large distance between clusters, there are 7 peaks which implies 8 clusters as is expected for 8QAM. In this case an ε value which is smaller than the smallest peak but larger than the small distances between cluster points will lead to a strong DBSCAN performance; therefore, in this case an ε which is just smaller than the 7th largest value in the dataset will enable for accurate clustering of 8QAM's arguments. In general, this rule holds for the argument and magnitude clustering for all modulation schemes, it can be expressed as follows:

For a dataset of point differences X, sorted in descending order:

$$X = \{x_1, x_2, x_3, \ldots, x_m\} \tag{6.4}$$

Where m denotes the number of elements in the dataset. Set d as the expected number of clusters formed by the modulation scheme's argument or magnitude values represented by dataset X. The range in which the optimum ε value lies is therefore:

$$x_{d-1} > \epsilon > x_d \tag{6.5}$$

Using this method the optimum ε may be algorithmically determined as long as the expected number of clusters is known. The method for determining the expected number of clusters is detailed in the following section.

6.4.7 Determining the Expected Number of Clusters

While the expected number of argument and magnitude clusters of low-order modulation schemes can be determined by inspection, this method is impractical for high-order modulation schemes such as 512QAM and 1024QAM. Fortunately, DBSCAN can be utilised to determine the number of expected clusters algorithmically.

Applying the magnitude and argument data taken from noiseless I/Q samples to the 1D DBSCAN algorithm produces an exact number of expected clusters. Selecting a value of ε which can facilitate this process can be performed by plotting the 1-distance graph of a sorted magnitude or argument graph of the highest order modulation scheme in the dataset, any ε value between 0 and the smallest non-zero value will be adequate.

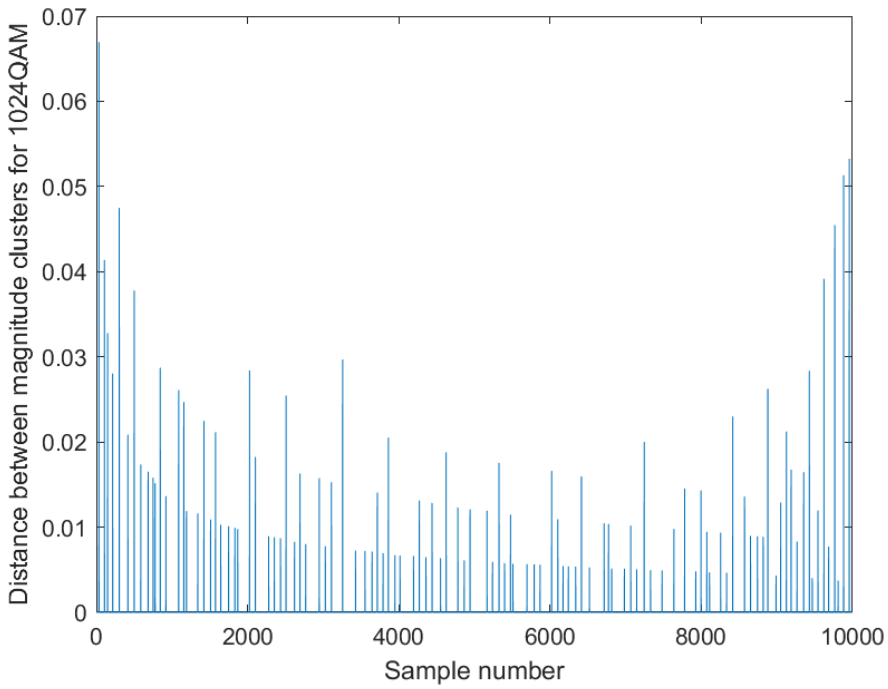


Figure 6.20: *The Unsorted 1-Distance Graph of 1024 Magnitude Data without Noise Impairment*

Figure 6.20 shows the distances between magnitude clusters for pure 1024QAM, the smallest non-zero value is 0.003, therefore an ε of between 0 and 0.003 will produce accurate clustering of every modulation scheme in the dataset. When DBSCAN is executed with ε values obtained in this manner the values for the expected number of clusters shown in Table 6.1 are obtained:

Modulation Scheme	Magnitude clusters	Argument Clusters
2QAM	1	2
4QAM	1	4
8QAM	2	8
16QAM	3	12
32QAM	5	28
64QAM	9	52
128QAM	16	108
256QAM	32	196
512QAM	56	420
1024QAM	109	847

Table 6.1: *Cluster counts for various QAM modulation schemes*

6.4.8 Applying the Expected Number of Clusters to Find an Optimal ε

As the expected number of magnitude and argument clusters d is known, the sorted 1-distance graphs may once again be plotted, the Y-axis value held in the $d-1th$ datapoint represents

the upper limit ε, the Y-axis value held in the dth datapoint represents the lower bound. Any ε value between these two extremes will provide a number of argument and magnitude clusters which is equivalent to the values shown in Table 6.1, for a specific modulation scheme at a particular SNR and assuming that there is enough samples to result in at least $minPts$ datapoints per constellation point.

Figures 6.21 and 6.22 show how the size of the $d-1th$ difference value for the magnitude and argument data respectively varies between modulation schemes and across different SNRs. In general, the lower the order of the modulation schemes, the greater the difference value and thus the greater the optimum ε. The SNR has a similar effect upon the difference value, as SNR increases, so too does the $d-1th$ difference value. Both trends are to be expected as the density of constellation spacing increases as modulation order increases, similarly, increasing noise increases the radius of constellation points, therefore reducing separation between them.

Figure 6.21 also shows that the variation between the $d-1th$ difference value of high-order modulation schemes with respect to the SNR is minimal in comparison to low-order modulation schemes. Furthermore, at low SNRs the $d-1th$ difference values of all modulation schemes converge. As low SNR data and high-order modulation schemes are the most difficult to accurately classify and as it has been shown in Section 6.4.3 that selecting the ε value which is optimum for the highest order modulation schemes provides more predictable feature cluster positioning, it is advantageous for the purposes of modulation classification that the optimum ε values for both low SNR and high-order modulated signals are similar. This allows for an ε selected at these values to perform well in the majority of the most difficult cases. Similarly, in Figure 6.22 (b) most signals show little variation in the optimum ε, discounting the peaks seen at around 5-15dB SNR by the lowest order modulated signals, this again allows for an ε pair to be chosen which will provide strong performance in most cases. The following section will detail an easier method of arriving at these ε values and then draw comparisons to the accuracy achieved with ε values chosen via the elbow point method.

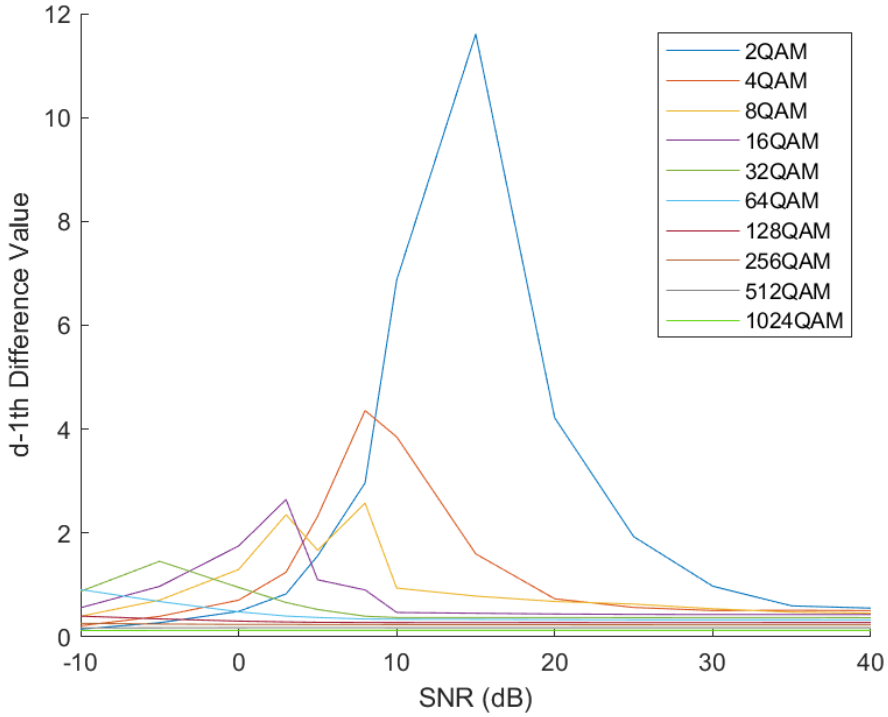


Figure 6.21: *How the $d-1$th Difference Value of Argument Data Varies with Respect to SNR for QAM orders 2 to 1024*

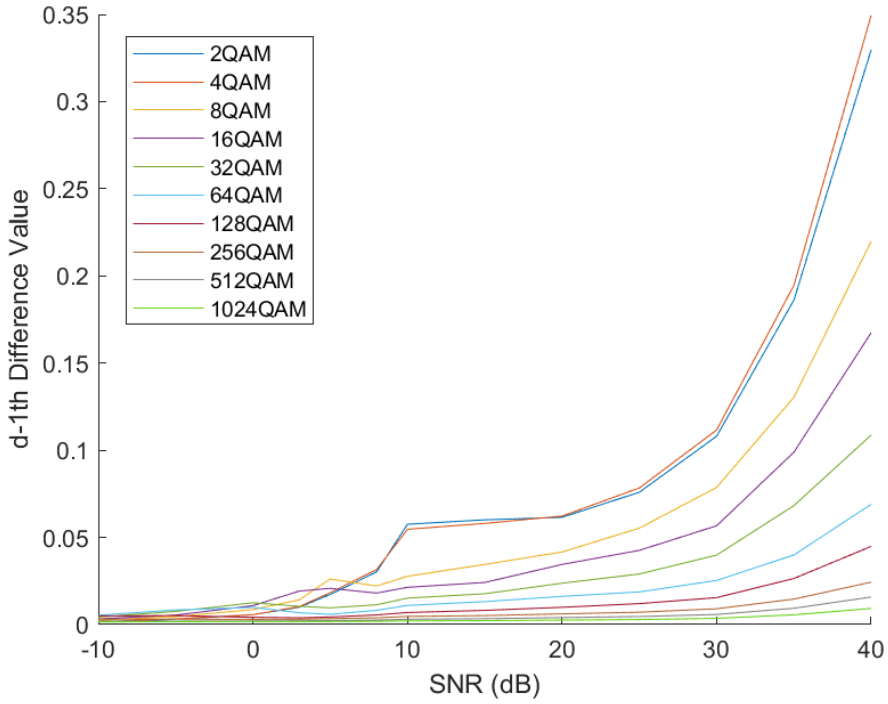


Figure 6.22: *How the $d-1$th Difference Value of Magnitude Data Varies with Respect to SNR for QAM orders 2 to 1024*

6.4.9 The RMS Method

The method of taking the $d-1th$ difference value has been shown to provide a means of selecting ε which reduces subjectivity. However, it is time consuming to tune ε in this manner as the expected number of clusters needs to be found and then this must be used to obtain the $d-1th$ difference value, in addition, tuning so finely to a particular dataset risks overfitting. A similar but quicker and more generalised method for finding an optimum ε is to find the Root Mean Square (RMS) of the 1-distance graph, this works as a proxy for determining a strong threshold. An example of the RMS and $d-1th$ difference value overlaid on the 1-distance graphs of 16QAM may be seen in Figures 6.23 and 6.24.

Figure 6.23 shows the unsorted 1-distance graph of the 16QAM argument data at 40dB SNR, there are 11 peaks which therefore implies 12 clusters should be found. Any point smaller than the smallest peak yet greater than the distances between cluster points will result in a strong ε value, the RMS is guaranteed to find a value such as this as the large difference values between the clusters are squared and therefore contribute more to the RMS value.

In the case of Figure 6.24, the RMS is significantly lower than in 6.23 yet as the RMS still lies between the value of the peaks and the cluster data difference values, again this value for ε would still prove to be effective. Both 1-distance graphs demonstrate that RMS can be used as a tool to find an effective threshold, but it was also found that finding the RMS of each 1-distance graph approximated the $d-1th$ difference value and that the RMS value and $d-1th$ difference value converged as the SNR decreased and modulation order increased. This can be seen in Figures 6.25 and 6.26.

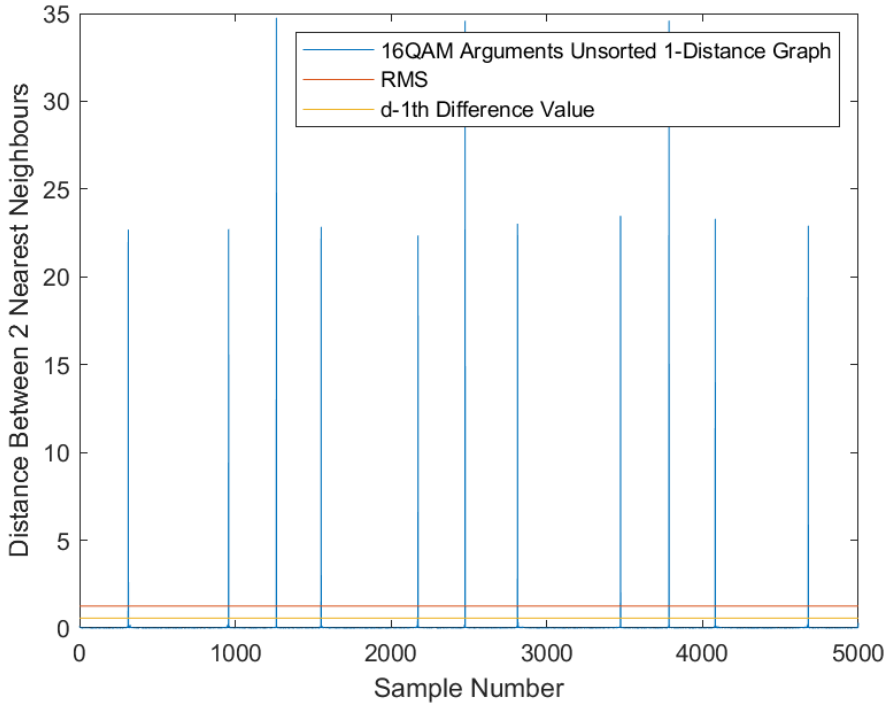


Figure 6.23: *The RMS and $d-1th$ Difference Value Overlaid on an Unsorted 16QAM Argument 1-Distance Graph*

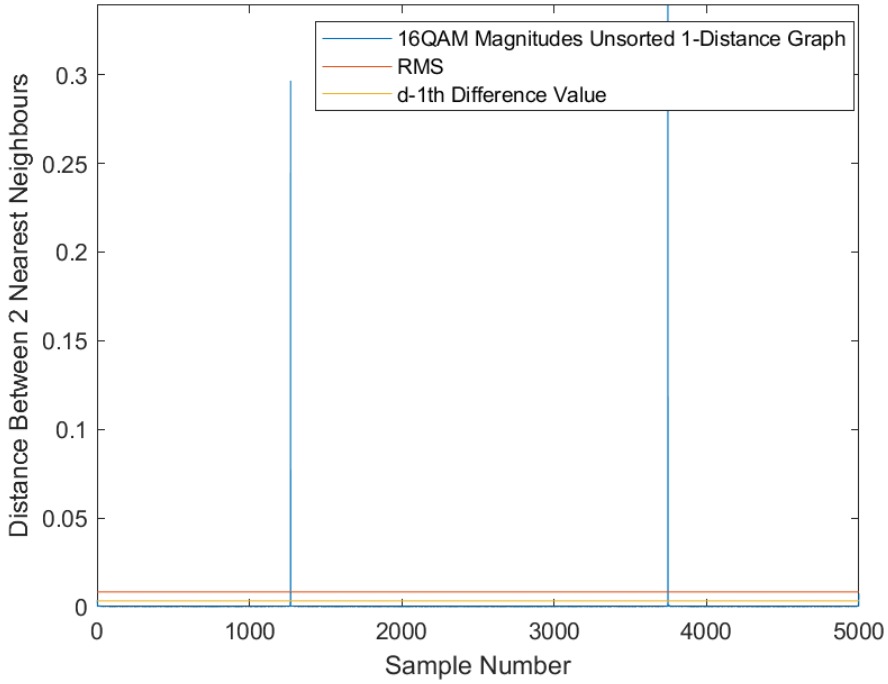


Figure 6.24: *The RMS and $d-1$th Difference Value Overlaid on an Unsorted 16QAM Magnitude 1-Distance Graph*

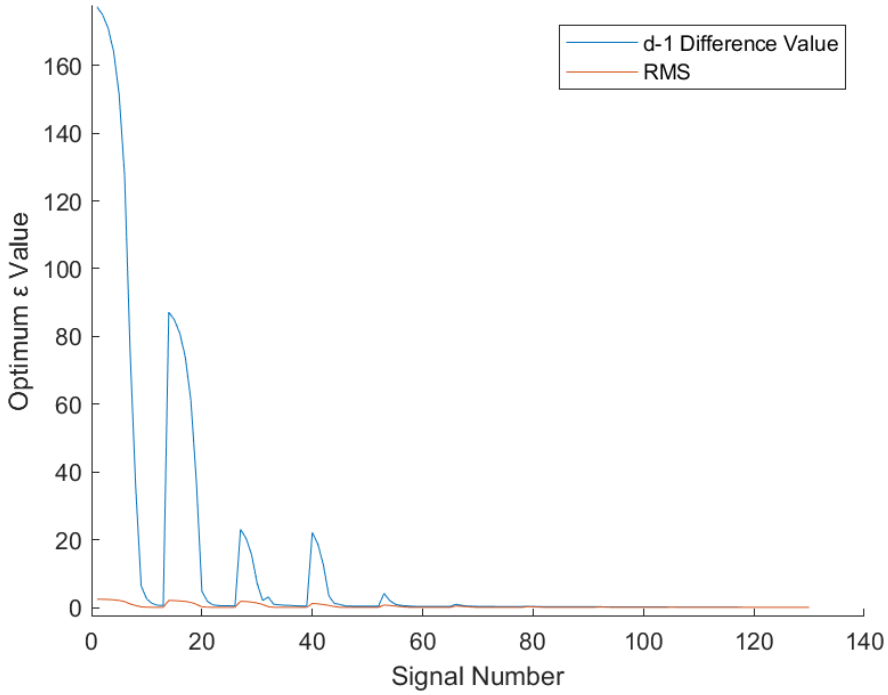


Figure 6.25: *How the $d-1$th Difference Value and RMS of the Argument Data Varies Across QAM Orders and SNRs*

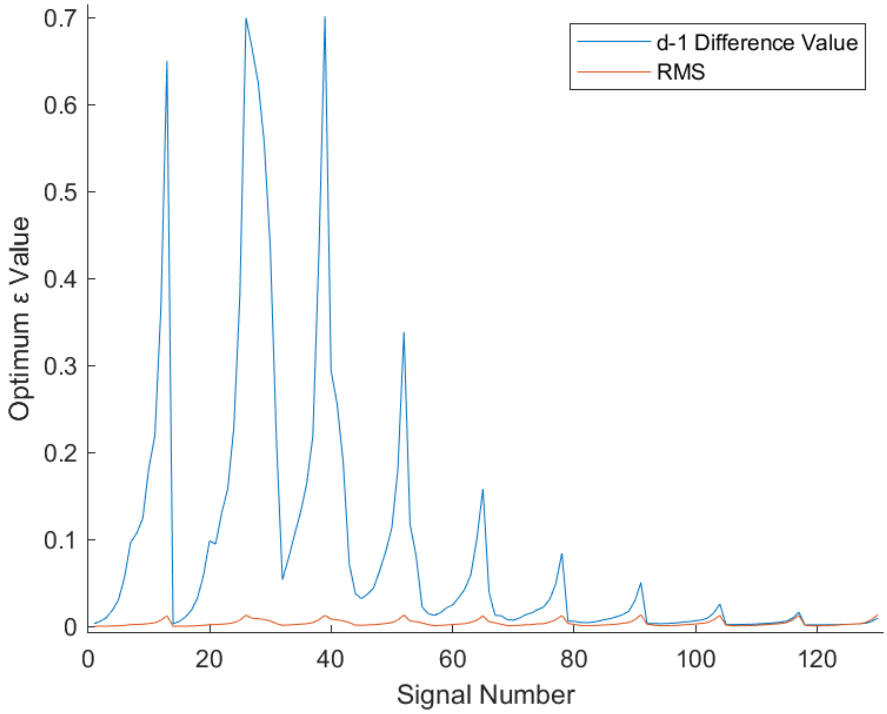


Figure 6.26: *How the $d-1th$ Difference Value and RMS of the Magnitude Data Varies Across QAM Orders and SNRs*

Figure 6.25 shows how the $d-1th$ difference value and the RMS for the argument data change with respect to the modulation scheme and SNR. Figure 6.26 shows the same but for the magnitude data. In these figures the X-axis is labelled as the signal number, each signal is modulated and at a specific SNR, there are 10 modulation types with 13 different SNRs, therefore there are 130 different signals, the graphs may be seen as the same as the graphs shown in Figures 6.21 and 6.21 but arranged as a single trend rather than overlapping one another. The modulation schemes are arranged from lowest to highest order, the SNRs are arranged from highest to lowest SNR, therefore every 13 signals the modulation scheme increases in order after the lowest SNR signal of a particular modulation scheme.

The actual modulation scheme and SNR combinations are unimportant, what the graphs illustrate is that the $d-1th$ difference value and the RMS have a relationship. In Figure 6.25 and 6.26 the $d-1th$ difference value line peaks much higher than the RMS value on low-order modulation schemes but both metrics follow the same trend of peaks and troughs. Figure 6.25 shows that as the modulation order increases this relationship becomes even clearer, above a dataset number of 60 which means 64QAM and upwards, the $d-1th$ difference value and the RMS value hold very similar values, converging around a value of 0.1.

Figure 6.26 shows a similar relationship, the $d-1th$ difference value and the RMS trace a follow a similar trend above a signal number of 92 (256QAM and greater), in the majority of cases the values obtained via both methods are nearly identical.

Should a designer require, when employing the $d-1th$ difference value method or the RMS method, the ε selection process can be automated to choose a strong value with no human intervention. For example, consider the RMS curves in Figures 6.25 and 6.26, an

automated process of extracting a strong ε could be written which uses the minimum value of each curve, another designer may desire the minimum value from a particular curve and the maximum value from another, extracting any strong value from these graphs is trivial and as each point on the curve will provide a strong ε hyperparameter for a specific use case.

It was found in Section 6.4.3 that selecting ε values which are optimal for the highest order modulation scheme allows for predictable feature cluster placement. Therefore in this case it is advantageous for determining the optimum ε that both the $d-1th$ difference value method and RMS curves converge when high order modulated signals are employed, additionally the curve variation with respect to SNR is minimal at high modulation orders which means that selecting an ε from this region will provide strong performance across all SNRs for high-order modulated data. It was noted in the previous section that the low SNR signals of the lower-order modulated signals also had an optimum ε in the same range as the high-order signals, the same can be said from the RMS curves in this case, as the high-order and low SNR signals are the most difficult to classify accurately, selecting the ε value which is optimum for the high-order modulation schemes should provide strong performance across all modulation schemes and SNRs. It was therefore decided that the 130th value of both RMS curves would be the ε values used for DBSCAN in the modulation classifier.

It should be noted that, just as discussed with the elbow point method, changing the DBSCAN dataset size also changes the optimum ε value produced by this method, which is to be expected as the optimum ε changes with the DBSCAN dataset size. The general shape of the curves remains unchanged, the difference lies with the magnitude of the produced values, for instance Figure 6.27 below is produced from a DBSCAN dataset size of 1000, whereas Figures 6.25 and 6.26 were created with a DBSCAN dataset size of 5000.

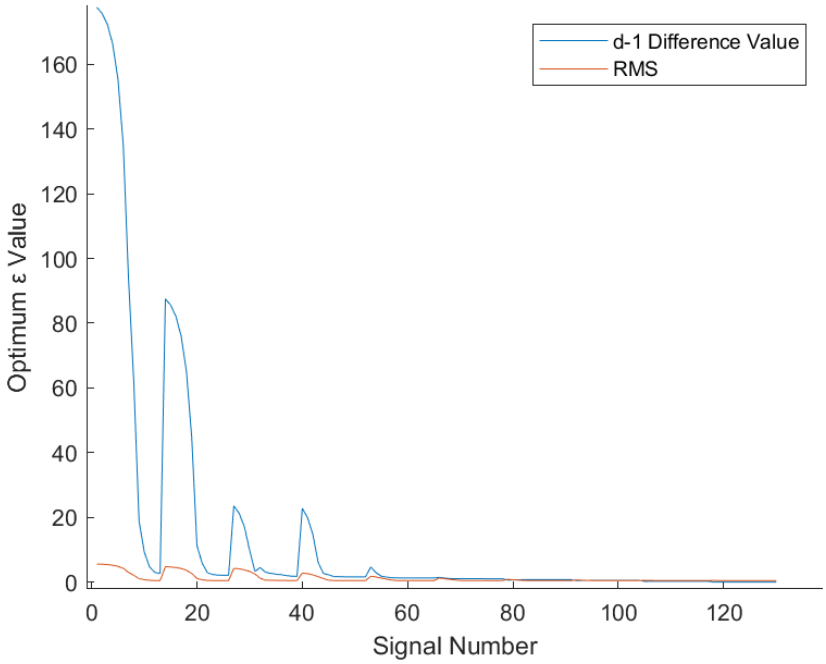


Figure 6.27: *How the $d-1th$ Difference Value and RMS of the Argument Data Varies Across QAM Orders and SNRs with a Dataset Size of 1000*

Figure 6.27 shows the same pattern of peaks and troughs as Figure 6.25 and the two curves converge for high order modulated data, however in this case the value at which the two graphs converge at is 0.5 rather than 0.1, demonstrating that for a dataset size of 1000 an ε value of 0.5 would be optimal.

Due to the RMS method providing similar values to that of the $d-1th$ difference value method for the more difficult to accurately classify modulation schemes and SNRs, it can be concluded that the RMS method can be employed as a proxy for the more labour-intensive $d-1th$ difference value method, the achieved accuracy of both methods in comparison to the elbow point method is found in the following section.

6.4.10 The $d-1th$ Difference Value and RMS Method Classification Results

Figure 6.28 shows a comparison curve of the classification accuracy achieved when the ε values found using the $d-1th$ difference value method and the RMS method are employed in the DBSCAN classification system, the curves from Figure 6.16 which were found with the elbow point method are included for comparison.

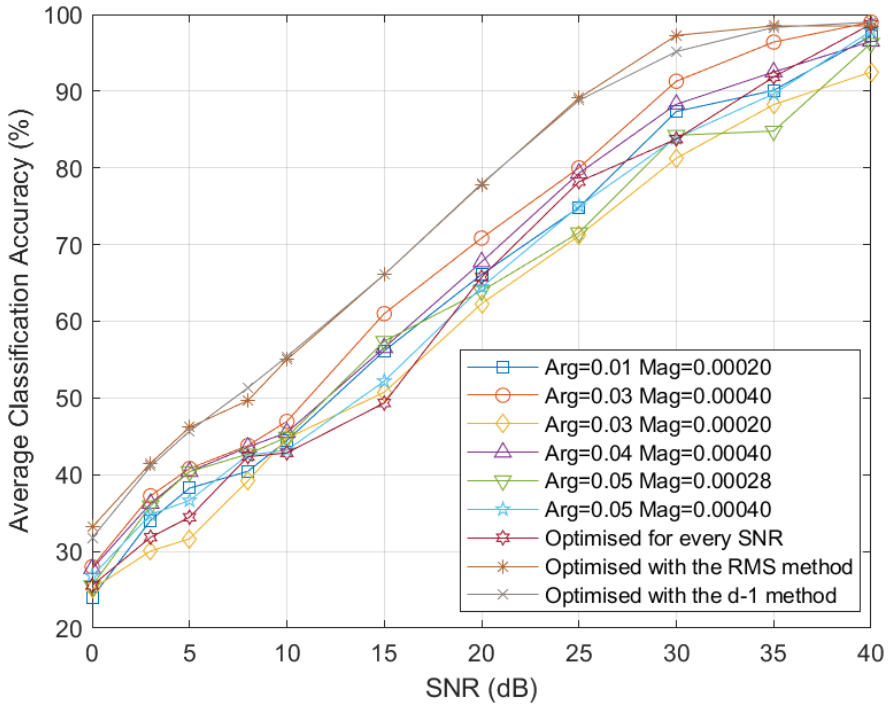


Figure 6.28: *The Average Classification Accuracy (%) Achieved with Various ε Values Selected Via the Elbow Point Method as well as the RMS and $d-1th$ Difference Value Methods*

Figure 6.28 shows that across every SNR the $d-1th$ difference value method and the RMS method find an ε value which outperforms the elbow point method, the only SNR at which an ε combination is found via the elbow point method which matches the two automated method's performance is at 40dB SNR. The smallest gains in performance are seen at the lowest and highest SNRs, for example at 0dB SNR the RMS method provides an ε which

allows for a classification accuracy of 33.46%, the elbow point method at a maximum achieves 28.0% classification accuracy, this is only a gain of 5.46% in terms of raw classification accuracy however in terms of increase in performance the RMS method has increased the classification accuracy of the system by 19.5%. In the midrange SNRs the proposed ε selection methods maintain a significant gap in classification accuracy, varying between a 5% and 9.8% raw classification accuracy increase, the largest delta in classification accuracy can be found at 25dB SNR, at this point the elbow point method achieves 79.3% classification accuracy and the RMS method achieves 89.1% classification accuracy, a raw improvement of 9.8% accuracy. The $d-1th$ difference value method and RMS method fail to reach 100% average classification accuracy at a high SNR, only reaching 99% at 40dB SNR, however the two methods do reach above 97% classification accuracy at an SNR of 30dB, in comparison the elbow point method only reaches this level of accuracy at 40dB SNR, a difference of 10dBs.

Finding the optimum ε value with the $d-1th$ difference value method or the RMS method has thus been shown to provide stronger classification performance across all SNRs and modulation schemes compared to when the elbow point method is employed. In addition to increased classification accuracy, the $d-1th$ difference value and RMS methods can be automated should that be required or at least can provide a semi-automated method that provides engineers working with 1D DBSCAN a greater understanding of the range of optimum ε across the entire dataset. Should a designer not wish to automate the ε selection process, the RMS and $d-1th$ value against SNR curves provide a means of gaining insight into how the optimum ε value changes with respect the SNR and signal modulation scheme.

6.4.11 The Effects of ε Conclusion

This section detailed the different methodologies for finding the optimum ε value for a given dataset size and *minPts* value. To begin, the elbow point method for selecting ε was outlined, this is the methodology for selecting ε given by the DBSCAN authors and is the generally accepted methodology used by DBSCAN engineers. The elbow point method involves plotting a k-distance graph, where k is the *minPts* value minus 1, in this case 1. The 1-distance graph determines the distance from each point in a dataset to its closest neighbour point, these distances are sorted and plotted as a curve. The point of the curve at which the gradient sees the largest rate of change is known as the elbow point and may be taken as a strong value for the ε hyperparameter. As the problem of modulation classification involves a range of modulated signals at various SNRs, there was found to be a large range of elbow points to choose from. Selecting two ε values which were of the highest value in the range shown by the elbow point method was shown to produce predictable feature cluster spacing.

Issues with the elbow point method were outlined, these issues were the fact that where the elbow point lied could be subjective depending upon the scale of the graph's axis. It was shown that the subjectivity combined with the multivariate SNR and modulation scheme optimisation problem led to difficulties in finding ε values which were optimal. A comparison between the accuracy achieved by systems using various ε pair combinations showed that when ε was optimised for each SNR the performance decreased. As this finding was contrary to the expected result it was concluded that the uncertainty introduced by the elbow point method's subjectivity was the issue.

An automated method of determining a strong ε value was developed, which was coined the $d-1th$ difference value method. It was shown that operating DBSCAN on noiseless

constellation diagrams can determine the expected number of magnitude and argument clusters. As the expected number of magnitude and argument clusters can be determined, the 1-distance graph can be examined for the value of the $d-1th$ largest point, where d is the number of expected clusters. Setting ε at or below this value will result in accurate clustering of a particular modulation scheme. The value of the $d-1th$ point is different for each modulation scheme and SNR; two graphs were plotted which showed that the values of the $d-1th$ point for high-order and high SNR low-order modulation schemes were similar which allows for an ε to be chosen which will provide strong performance across the most difficult signals to classify. Rather than undergo the process of determining what the expected number of clusters is and then subsequently writing code to determine the values of the $d-1th$ points, it was shown that taking the RMS of the 1-distance graph can produce similar results. Figures 6.25 and 6.26 showed that the RMS and the values of the $d-1th$ points followed a similar trend, for high order modulated data and low SNR data the value of the RMS and the values of the $d-1th$ points converged, therefore taking the RMS of the 1-distance graphs can be used to select strong values of ε. To confirm these results, curves representing the aggregate classification accuracy of systems trained with ε values found with the RMS and the $d-1th$ difference value method were plotted alongside the classification accuracy trained with ε values found via the elbow point method. It was found that the RMS and the $d-1th$ difference value method found ε values which provided stronger classification accuracy at every SNR, achieving approximately a 10% improvement in classification accuracy in some cases. It was also discussed that the optimum ε value varies depending on the DBSCAN dataset size, therefore different ε pairings will be required to be found for differently sized models, as the RMS method was found to provide the strongest ε values it will be utilised to select the ε parameter pairings in the following chapters.

6.5 10-Bit Datapath Considerations

The analysis in this chapter has been based upon the usage of 64-bit floating point precision as is the default in MATLAB [94]. The implemented design features a 10-bit datapath, and therefore some hyperparameters must be optimised for this scenario.

The dataset size and *minPts* parameters do not require any changes. The dataset size sets the number of shift registers in the Insertion Sort module and therefore this value remains the same irrespective of the datapath precision. The *minPts* value determines the number of datapoints within ε to constitute a cluster forming meaning that the value also remains the same regardless of the datapath precision. However, due to the logic of the DBSCAN implementation in hardware, the variable must be set to the desired *minPts* value minus 1, this is due to the *PointCount* register having a default value of 0. Algorithmically, *minPts* remains at a value of 2 in this case.

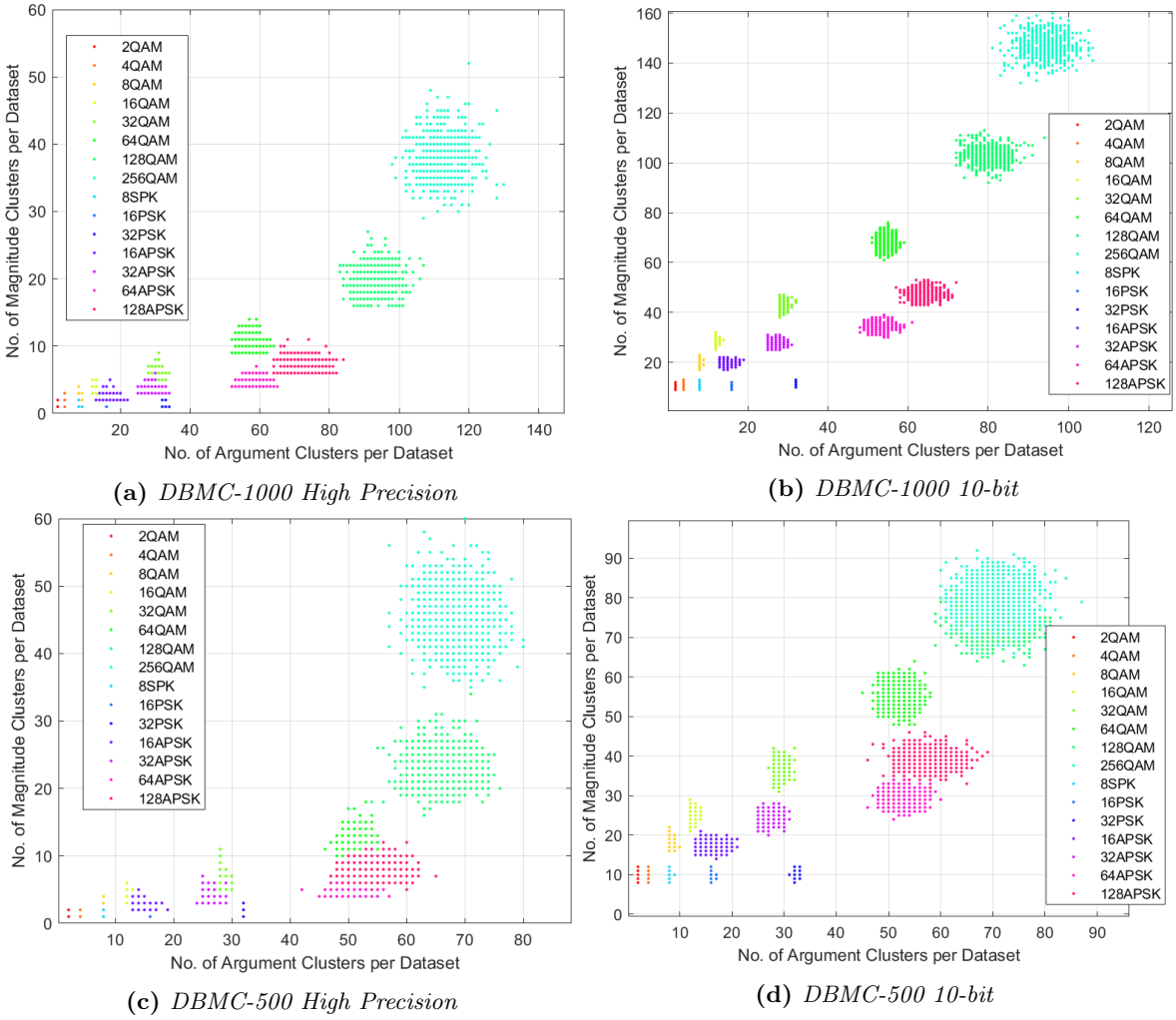
Where precision must be accounted for is with the ε parameters. To illustrate why this is the case take the optimum ε values obtained via the RMS method for DBMC-1000. Figure 6.27 showed that the optium argument ε value was 0.5, not shown was the optimum magnitude ε value of 0.003. Chapter 5 discussed how the 10-bit fixed point datapath assumed a maximum argument value of 360 and therefore required 9 integer bits with a single fractional bit, a maximum magnitude value of 3 was assumed, therefore requiring 2 integer bits and 8 fractional. The maximum precision able to be represented by data with this format is

therefore 0.5 and 0.0039065 for the argument and magnitude data respectively. It was found that due to the truncation which occurs following the CORDIC module, points within the minimum precision hold the same value. Therefore in the case of an argument ε of 0.5, as is optimal for a 1000-point dataset, an ε of 0 is actually optimal in hardware as all of the points within 0.5 of each other hold the same value. Similarly, for the optimum software magnitude ε of 0.003, the optimum ε was again found to be 0. This illustrates a general rule which was found: The optimum ε value to choose for the hardware implementation was 1-bit of the minimum precision lower than the closest value which can be reached with the 10-bit precision to what was found to be optimum at higher precision. To illustrate, Table 6.2 demonstrates both the optimum ε values found in software and the quantised values which were utilised in the hardware implementation.

Table 6.2: *The Optimum ε Values found with Software and the Quantised Approximations Utilised in the Hardware Implementations*

Implementation	**Raw Arg ε**	**Raw Mag ε**	**Quantised Arg ε**	**Quantised Mag ε**
DBMC-50	9.6	0.04	9.0	0.0351585
DBMC-250	2.04	0.0079	1.5	0.00390625
DBMC-500	1	0.003	0.5	0
DBMC-1000	0.5	0.003	0	0

It was found that with these quantised ε values feature clusters in feature space exhibited a high degree of separation, in some cases separation was superior to feature space obtained on 64-bit floating point values. Figure 6.29 (a) and (b) display the obtained feature space of DBMC-1000 on high precision data and quantised data respectively, Figure 6.29 (c) and (d) also display the obtained feature space of DBMC-500 on high precision data and quantised data respectively.



(a) *DBMC-1000 High Precision*

(b) *DBMC-1000 10-bit*

(c) *DBMC-500 High Precision*

(d) *DBMC-500 10-bit*

Figure 6.29: *A Comparison Between the Obtained Feature Spaces of High-Precision Data and 10-bit Quantised Data at 40dB SNR*

It can be seen in Figure 6.29 that the feature spaces are different in each case, this is due to the ε values being slightly different. However, despite the differences in each case there is still a high degree of separation between clusters and strong performance is expected. The performance differential illustrated in Figure 6.30 which shows the variation in the obtained average accuracy between using high precision data and ε values and quantised data and ε values. In each case the obtained accuracy trends are somewhat different but not by a significant degree.

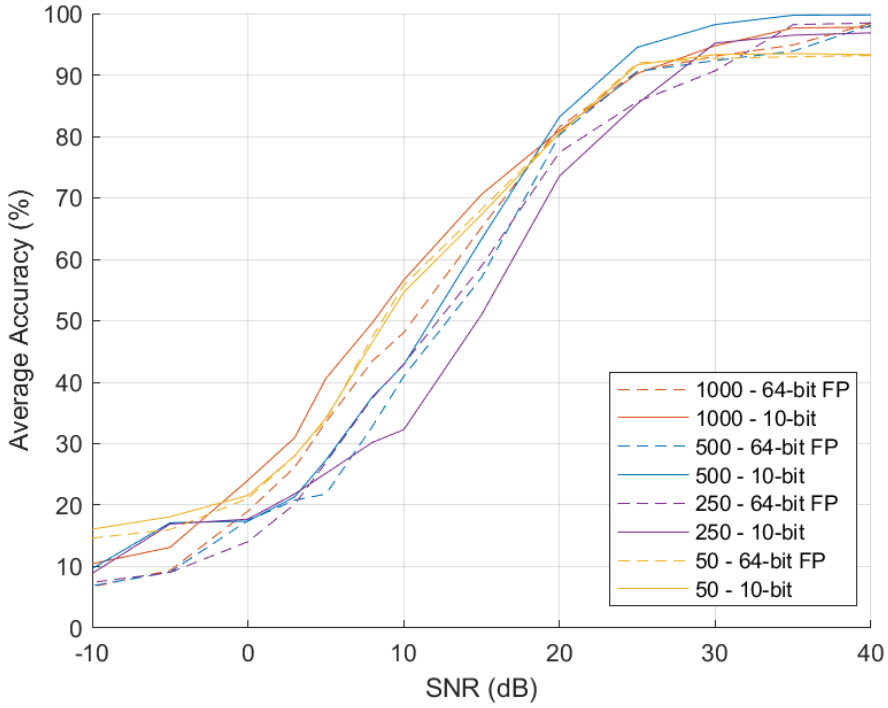


Figure 6.30: *The Average Classification Accuracy (%) Against SNR (dB) Achieved with 10-bit Fixed Point and 64-bit Floating-Point ε Values and Data*

Thus, with a 10-bit datapath appropriate ε values can be obtained by approximating the values obtained with higher precision calculations and taking away a value equal to 1-bit of the minimum precision. It was also found earlier in this chapter that optimum ε values for a dataset size of 5000 were smaller than for a dataset size of 1000. Implementation of a 5000-point classifier would necessitate an increase in the datapath precision to accommodate the smaller ε values.

6.6 Hyperparameter Selection Conclusion

This chapter has discussed methods for optimising the various hyperparameters which are required to operate the DBSCAN feature extractor. First, the DBSCAN dataset size was discussed, it was shown that the minimum dataset size should be set to at least the value of *minPts* multiplied by the largest expected number of arguments or magnitudes. It was also shown that larger datasets result in larger feature spaces, which enables greater feature cluster separation and therefore increases classification accuracy. However, as the hardware implementation utilisation scaled with dataset size it was decided to keep this parameter at the minimum required value.

It was also shown that *minPts* has a similar effect on the scale of the feature space to the DBSCAN dataset size. It was also shown to result in feature clusters of reduced radii, improving feature separation. Increasing the value of *minPts* necessitates an increase in the dataset size for performance to be maintained, increasing both values may lead to further gains in feature separation, but again it was decided to set *minPts* to the minimum suggested

value of 2 to maintain a small implementation size.

The elbow point method in conjunction with 1-distance graphs were used to demonstrate how the optimum ε values vary across SNRs and modulation schemes. It was shown that the elbow point method is flawed when the 1-distance graph does not feature an instantaneous change in gradient. The d-1 difference value and RMS methods were proposed as a stronger selection algorithm, they were shown to result in ε values which provide increased classification accuracy across all SNRs whilst also decreasing the subjectivity of the selection process in comparison to the elbow point method. The newly proposed methods are recommended for optimising the DBSCAN feature extraction algorithm.

Finally, it was discussed how the 10-bit datapath of the hardware implemented systems necessitated the usage of quantised ε values. It was shown that 10-bit values allowed for enough precision to closely match the achieved performance when utilising 64-bit floating point ε values.

Chapter 7

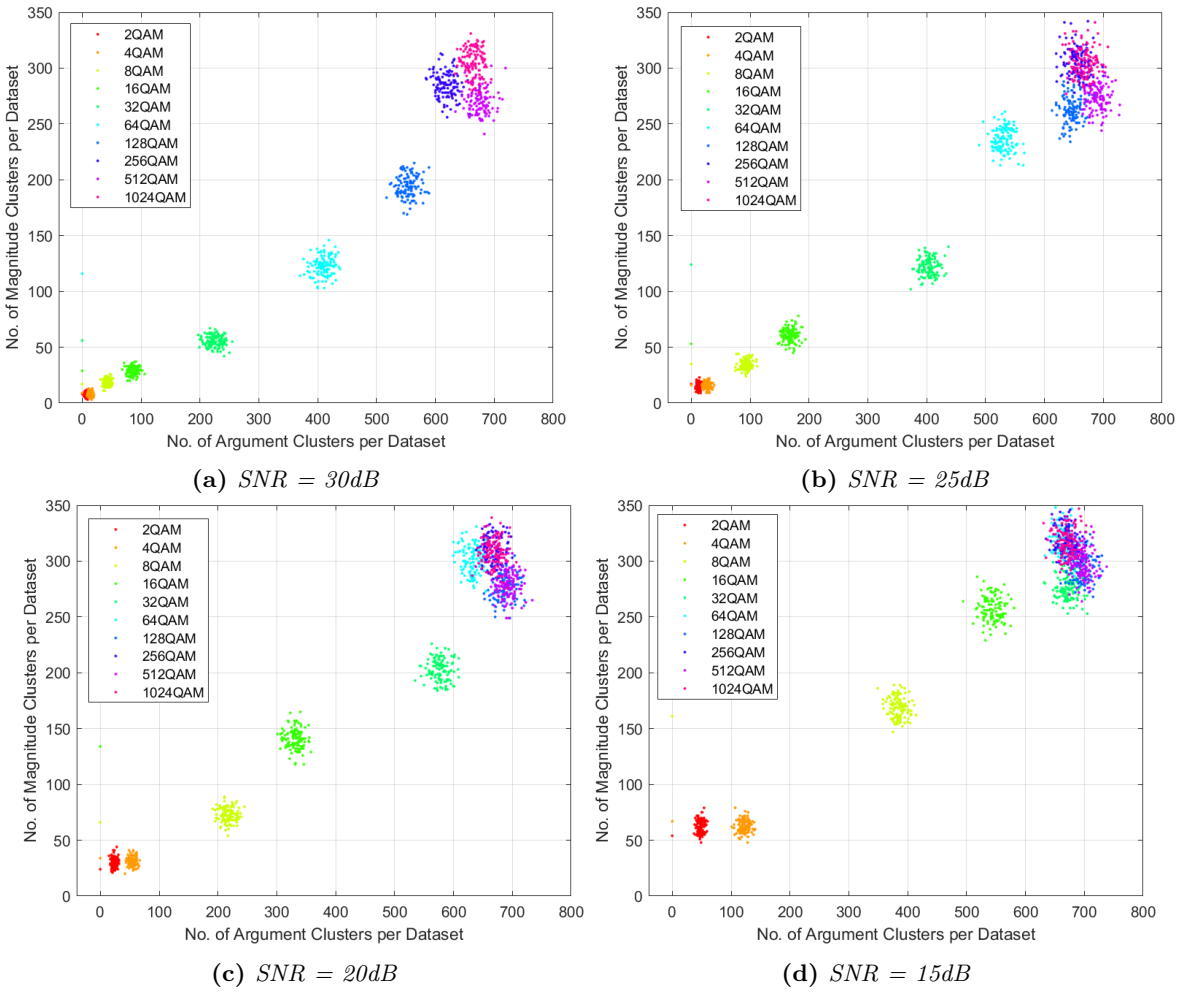
Modulation Classification Performance

The effects of each hyperparameter required to operate the classification system have now been quantified, thus allowing for the classification performance to be discussed in the context of these parameters. This chapter will present the classification accuracy against SNR performance for each DBSCAN dataset size while including only the recommended modulation schemes for each dataset size. For each of the following graphs the *minPts* hyperparameter was set to 2 and the values of ε were set to values found via the RMS method which will be listed at the beginning of each individual section.

7.1 Classification Mechanism and Accuracy Degradation

Before the results of each DBMC configuration are provided, it is first necessary to provide an explanation of the mechanism of classification and accuracy degradation. Throughout this thesis the relative positioning of feature clusters in feature space has been used as a tool to demonstrate the operation of the system. Perfect classification accuracy is obtained when all feature clusters exhibit strong separation in feature space, thereby allowing for the classifier to find strong decision boundaries. Accuracy is therefore reduced when feature clusters overlap. In general, feature cluster overlap first occurs between high-order signals, as the SNR falls feature clusters extracted from modulation schemes of progressively lower orders begin to overlap with those of higher orders. In feature space this may be seen as one large super feature cluster gradually absorbing feature clusters of progressively lower order modulation schemes as the SNR falls.

Figure 7.1 (a), (b), (c), and (d) show the resulting feature space obtained on a dataset size of 5000 at 30dB, 25dB, 20dB, and 15dB SNR respectively. The feature space shown in Figure 7.1 (a) shows the lowest SNR at which there is separation between the 3 high-order feature clusters. In (b) there is little separation between the feature clusters of the two highest order feature clusters, 256QAM has significant overlap with 512QAM and 1024QAM, 128QAM also begins to overlap with the three highest order feature clusters. In (c) it can be seen that all modulation schemes of orders 64 and greater overlap. Finally, in (d) 32QAM begins to overlap with the feature supercluster.



(a) *SNR = 30dB*

(b) *SNR = 25dB*

(c) *SNR = 20dB*

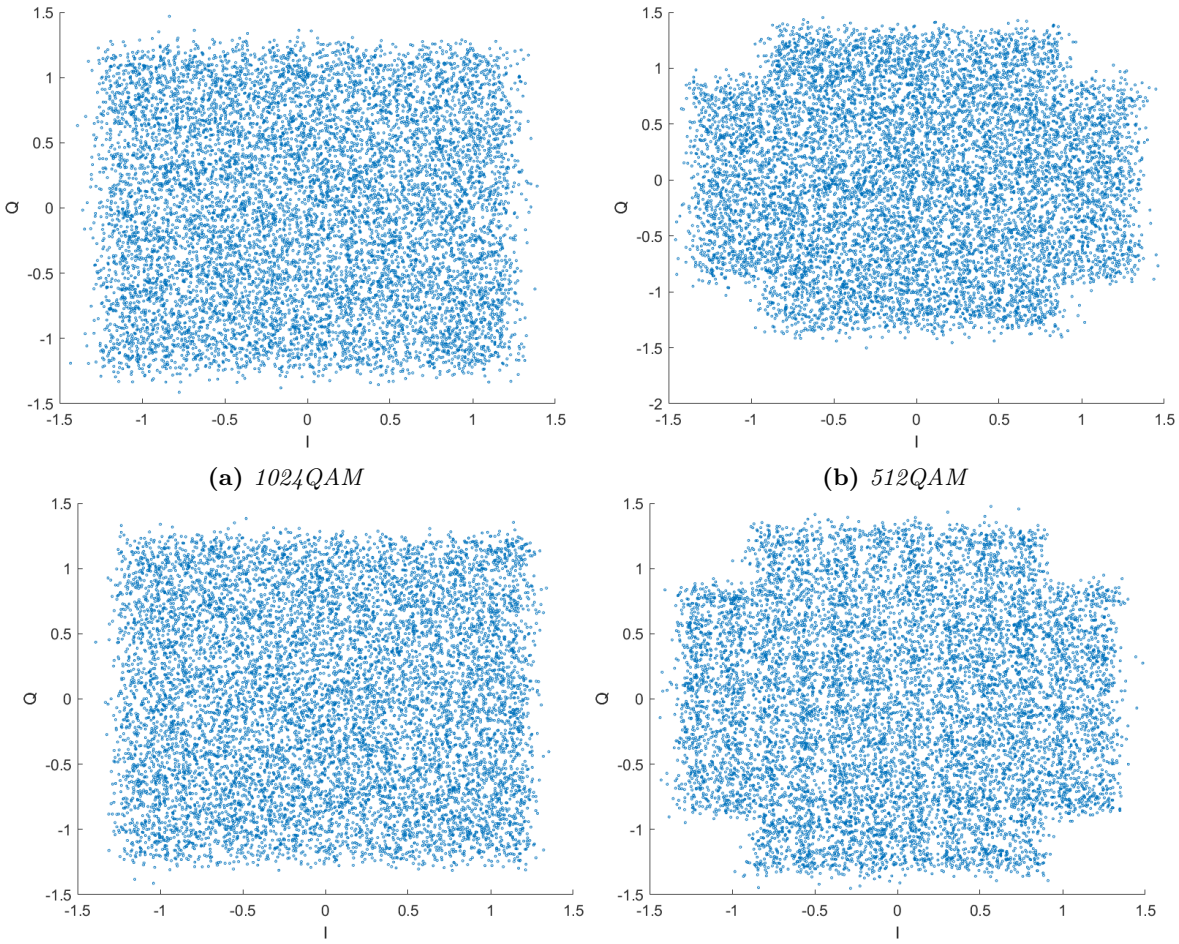
(d) *SNR = 15dB*

Figure 7.1: *How QAM Feature Cluster Arrangement Varies Between SNRs 30dB to 15dB*

This is the mechanism of accuracy reduction, as the SNR decreases feature clusters begin to overlap with increasing severity. At extremely low SNRs such as -10dB feature space will appear to be one large overlapping cluster. Perfect classification accuracy could be maintained to lower SNRs by selecting a dataset which consists of lower-order signals. For instance, in this example 30dB is the lowest SNR at which near perfect accuracy is achieved. At 25dB SNR 128QAM begins to overlap with the three highest order signals. However, should the three highest order signals be removed from the dataset, perfect classification accuracy could be maintained as low as 15dB SNR, as this is the SNR at which 64QAM and 128QAM first see overlap in feature space.

The reason for the feature supercluster formation is that as SNR decreases the appearance of the constellation diagrams of various modulation schemes becomes increasingly similar. For example, at 20dB SNR the four highest order modulation schemes all lose clearly defined constellation point separation and appear as a noisy cross or square, therefore the information which the DBSCAN algorithm extracts is lost below certain SNRs. This leads to the clustering algorithm finding a similar number of argument and magnitude clusters for constellation

diagrams which reach this level of noise. The SNR where each modulation schemes constellation diagram appears as shown in Figure 7.2 is dependent upon the order, higher-order schemes begin with reduced separation between constellation points in a noiseless scenario, they therefore require less noise corruption to reach this appearance.



(a) *1024QAM* **(b)** *512QAM*

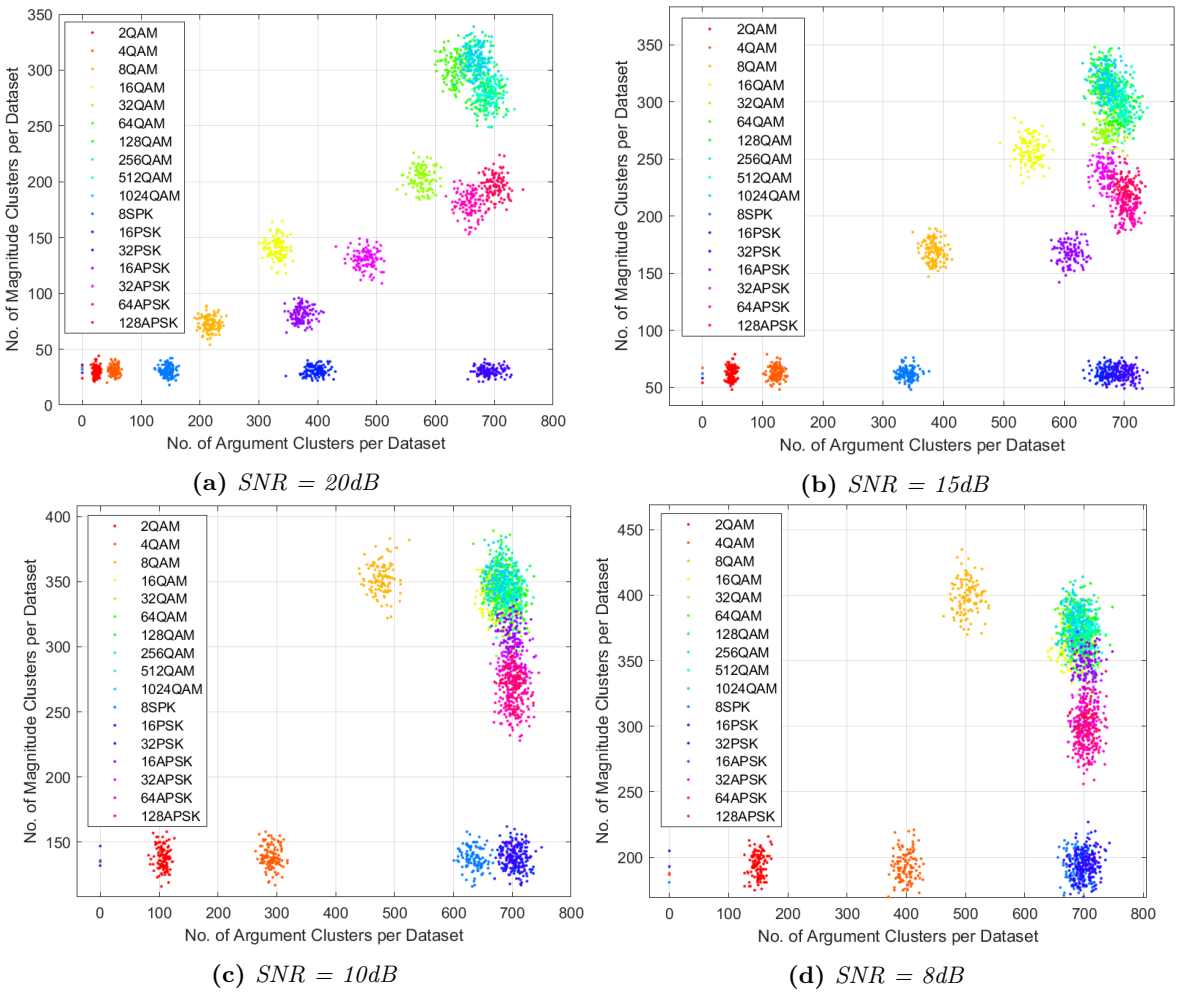
(c) *256QAM* **(d)** *128QAM*

Figure 7.2: *The Constellation Diagrams of High-Order QAM Signals at 20dB SNR*

7.1.1 PSK and APSK Accuracy Degradation Mechanism

Figures demonstrating feature space shown thus far in this thesis have primarily included only QAM signals for visual clarity. However, the DBMC system is also capable of classifying PSK and APSK signals, this section briefly describes the mechanism of accuracy degradation when these signals are included in the training and testing dataset.

Similarly to QAM, PSK and APSK feature clusters begin to overlap as the SNR decreases. However, these modulation formats tend to first form their own overlapping supercluster and then superclusters begin to overlap at very low SNRs. Figure 7.3 illustrates the feature cluster overlap from 20dB to 8db SNR.



(a) *SNR = 20dB*

(b) *SNR = 15dB*

(c) *SNR = 10dB*

(d) *SNR = 8dB*

Figure 7.3: *Feature Cluster Arrangement of QAM, PSK, and APSK Signals at SNRs 20dB, 15dB, 10dB, and 8dB*

Figure 7.3 (a), (b), (c), and (d) shows the resulting feature space when a dataset size of 5000 is employed to extract features from signals of 20dB, 15dB, 10dB, and 8dB respectively. In (a) the high-order QAM signals exhibit significant overlap, 128APSK and 64APSK also begin to exhibit minor overlap, the PSK signals are well separated. In (b) 16PSK and 32PSK begin to overlap, as do the 3 highest order APSK signals. In (c) the 8PSK feature cluster moves closer to 16PSK and 32PSK and all APSK signals now exhibit at least some degree of overlap. In (d) all PSK feature clusters overlap (other than 2QAM and 4QAM which could also be described as BPSK and QPSK). Similarly, all APSK signals bar 16APSK overlap. 16APSK does not overlap significantly with other APSK signals yet does overlap with the QAM supercluster purely by chance, in this case 100% classification accuracy is lost on 16APSK due to the inclusion of QAM signals in the dataset.

7.2 Dataset Description

The dataset employed for training and testing the DBMC systems (as well as the SNR estimation functionality) utilised both simulated and lab-captured signals. Simulated data was used as extremely large datasets could be generated and managed easily; it was used as test and training data. The lab-captured dataset was used solely as test data, no difference in classification accuracy was observed between datasets generated by either method. Simulated data was created for 17 different modulation schemes at SNRs from -10dB to 40dB, 6,000,000 I/Q samples were generated per SNR/modulation scheme combination. Simulated data was generated with MATLAB R2021b [94], AWGN was the only signal impairment. QAM and PSK signals were created using the wireless waveform generator toolbox, APSK signals were created with a custom MATLAB script.

Lab-captured data was generated with the Rohde & Schwarz SMW100A [117] and captured with a Keysight N9030B PXA signal analyser [118]. Signals were generated at 50MSamples/s with an intermediate frequency bandwidth of 160MHz, the signal analyser sampled at a rate of 200MSa/s. Sample rate conversion was performed afterwards with MATLAB R2021b [94]. Signals were radiated between horn antennae at a distance of 6cm and 75cm. No signal impairments were applied other than AWGN. APSK signals were not captured in the laboratory as the signal generator did not offer the functionality to do so, as such all utilised APSK signals were generated with MATLAB R2021b with constellation point positioning set according to the DVB-S2X standard [119]. Figures 7.4 and 7.5 respectively show a photograph and block diagram of the lab capture setup.

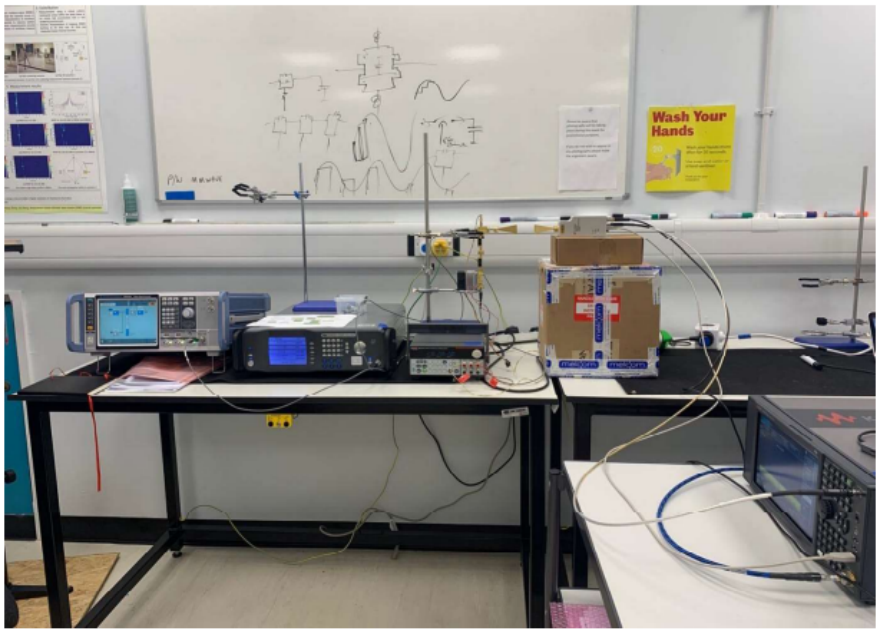


Figure 7.4: *Photograph of Horn Antennae Lab Data Capture Setup*

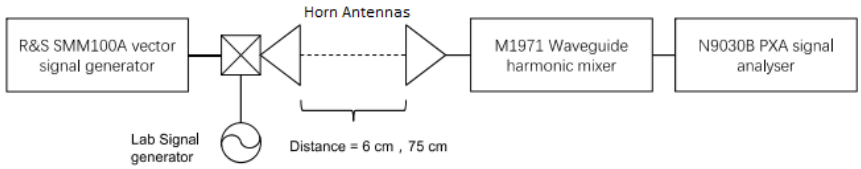


Figure 7.5: *Block Diagram of Horn Antennae Lab Capture Setup*

7.3 DBMC Modulation Classification Performance

Following the explanation of the mechanism of the DBMC accuracy degradation, the description of the employed dataset, and the details of the testing setup, every aspect of the modulation classification system has been outlined. This section provides statistics for the performance of each DBMC configuration.

In Chapter 5 the 4 different DBMC structures were discussed, each configuration has a particular feature extraction dataset size and number of MLP output nodes. Chapter 6 discussed how the feature extraction dataset size limits the maximum order of modulation schemes which can be classified accurately, therefore each DBMC configuration is limited to classifying a particular set of modulation schemes. The maximum modulation order which a configuration may classify accurately is governed by the number of expected argument clusters, for instance DBMC-50 cannot accurately classify 32QAM as there is an expected number of argument clusters of 28, thus the feature extraction dataset size is not large enough to accommodate accurate feature extraction of this modulation scheme.

Each configuration has the feature extraction dataset size denoted in the name of the system, all systems use a *minPts* value of 2, ε values were obtained using the RMS method. Hardware-implemented models had their ε values set according the methods outlined in Section 6.5, specific values will be provided at the beginning of the subsections that concern each configuration.

Throughout the coming comparisons it can be seen in the provided figures that the achieved classification accuracy on particular modulation schemes is highly variable, the reason for this phenomenon is twofold. Firstly, as explained in Section 6.1 and throughout this thesis, classification accuracy is wholly dependent on the spacing of the feature clusters in feature space, as the SNR decreases the positioning of feature clusters varies therefore it can sometimes be the case that two clusters with a high degree of overlap at a particular SNR cease to overlap significantly at a lower SNR thus increasing the achieved accuracy on each class. The second reason is that separate classifiers were trained at each SNR, when feature clusters overlap each trained classifier tended to prioritise a specific class over others, therefore the results graphs may show temporary spikes in accuracy of particular classes at particular SNRs. The reason for one class being prioritised over another between SNRs when feature clusters overlap is due to the randomness of weight initialisation at the beginning of each training run. Results were averaged over 5 test runs to smooth out accuracy curves but complete smoothness was not obtained. It should be assumed that once a class is classified with below 40% accuracy it is classified with an equivalent accuracy to all classes with a similar accuracy. It can be seen that the average classification accuracy in general follows a consistent trend even despite this variability in classification accuracy, this is because

when one class is prioritised by the decision boundaries found by the classifier another is deprioritised thus resulting in a consistent average.

7.3.1 DBMC-50 Classification Performance

The first DBMC configuration to discuss is the smallest. DBMC-50 is limited to operating on QAM, APSK, and PSK signals of orders up to and including 16. The argument and magnitude ε employed to obtain the following results was 9 and 0.0351585 respectively. Figure 7.6 displays the accuracy of the system on solely QAM signals.

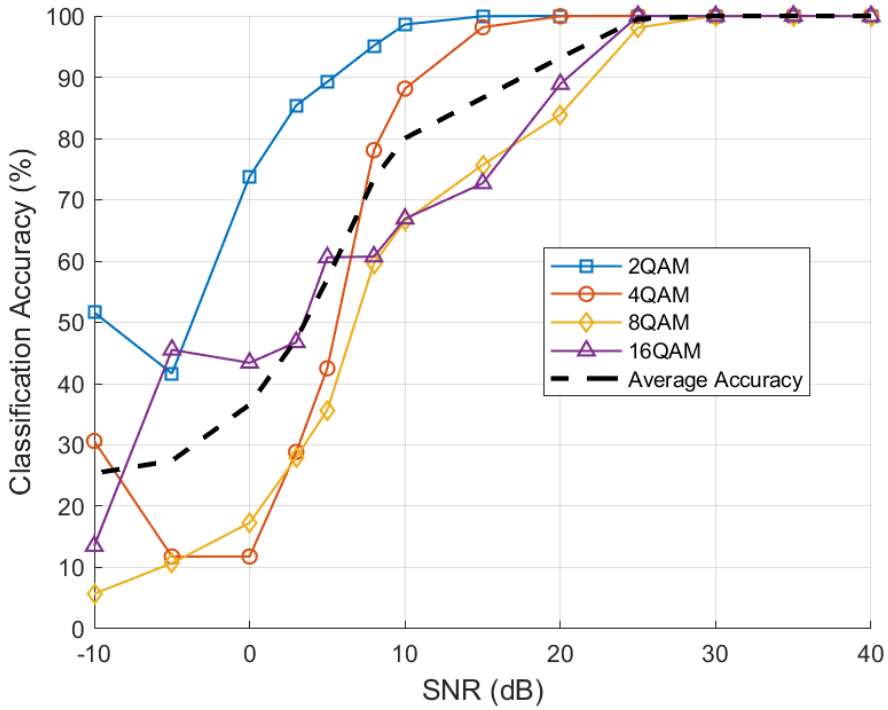


Figure 7.6: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-50 on the QAM only Dataset*

The system is shown to capable of achieving 100% classification accuracy at an SNR of 30dB and greater. At 25dB a reduction in overall accuracy to 99% is obtained because of slight feature cluster overlap between 8QAM and 16QAM. This overlap becomes increasingly severe as the SNR decreases, as a result the accuracy of both highest order schemes drops at a similar rate. At and below 10dB SNR 4QAM begins to significantly overlap with the higher order supercluster, similarly 2QAM sees slight overlap at this SNR. At -5dB SNR the accuracy of the system is 27%, an approximately equivalent rate to a random guess.

Figure 7.7 and 7.8 exhibit the accuracy of the DBMC-50 system when QAM, PSK, and APSK signals up to an order of 16 are input.

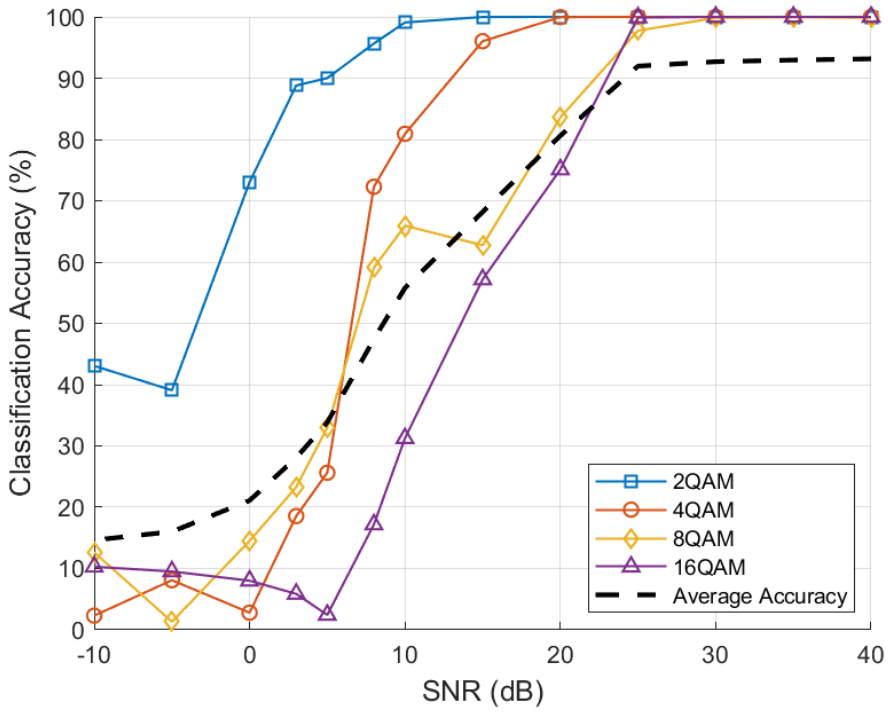


Figure 7.7: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-50 on the Full Dataset*

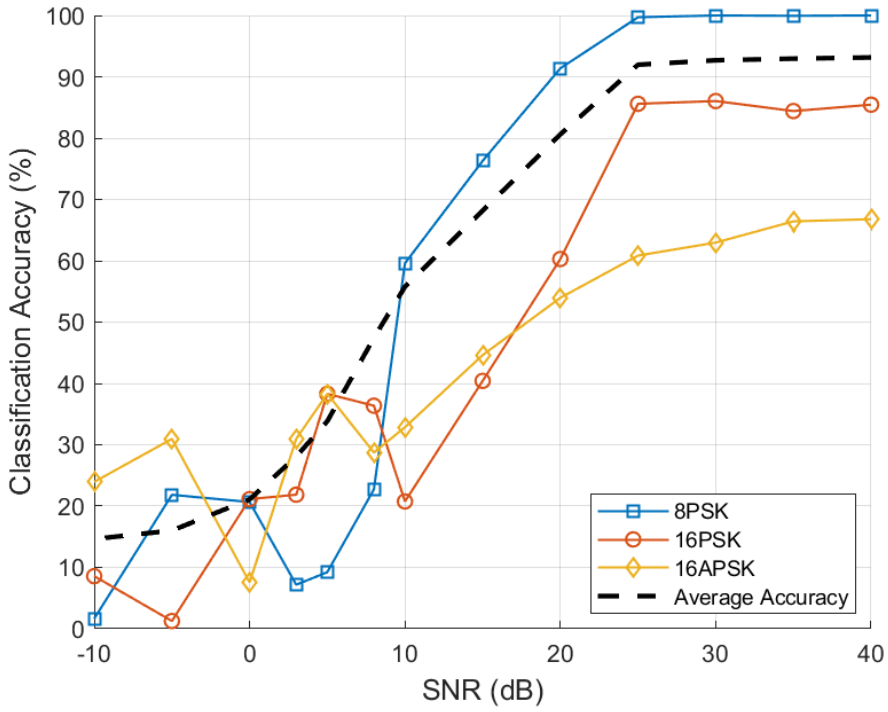


Figure 7.8: *PSK and APSK Classification Accuracy (%) Against SNR (dB) of DBMC-50 on the Full Dataset*

The peak average accuracy is reduced by the inclusion of 16PSK and 16APSK, the model misclassifies these two modulation schemes as each other even at the highest SNRs. At all SNRs the inclusion of these signals results in a decrease in average accuracy in comparison to using only QAM signals, this is shown by 2 of the 3 APSK and PSK signals lying below the average accuracy curve. The system's performance on the QAM signals remains similar to the performance seen in Figure 7.6, perfect classification accuracy of individual modulation schemes is lost at identical SNRs, the rate of accuracy degradation is also comparable to the previous results. Peak average accuracy is lost at 25dB SNR and approximate equivalency to random guess accuracy is also obtained at -5dB SNR. The general trend of average accuracy degradation is therefore similar across both comparisons, but the inclusion of 16PSK and 16APSK imposes a consistently lower average classification accuracy.

7.3.2 DBMC-250 Classification Performance

DBMC-250 is the second smallest configuration, it is limited to classifying modulation orders up to 128 due to the feature extraction dataset size. The argument and magnitude ε values employed were 1.5 and 0.00390625 respectively. Figure 7.9 shows the classification performance of the system on only QAM signals of orders up to and including 128QAM.

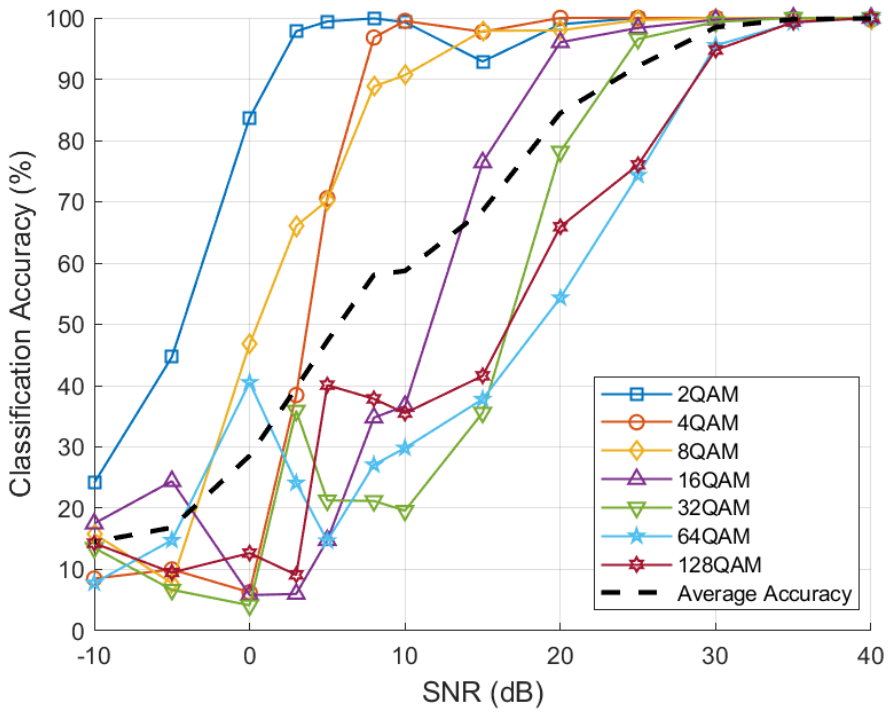


Figure 7.9: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-250 on the QAM only Dataset*

The peak accuracy of the system is 100% but this accuracy is only achieved at 40dB SNR, at 35dB it falls to 99.68%, at 30dB it falls further to 98.31%. Below 30dB SNR the rate of average accuracy reduction is in general consistent until -5dB. The reduced performance in comparison to DBMC-50 is wholly explained by the inclusion of higher order modulation

schemes. 64QAM and 128QAM are responsible for the initial reduction in average accuracy from 100%. 32QAM begins to impose a reduction in average accuracy at SNRs of 20dB and below. It can be seen that the system maintains an accuracy greater than 90% on the 3 lowest order QAM signals to a lower SNR than achieved by DBMC-50, therefore while the average accuracy of DBMC-250 is reduced by including higher order modulation schemes, the performance on low order modulation schemes is improved over the smaller model.

Figures 7.10 and 7.11 show the performance of the system when PSK and APSK signals up to a respective order of 32 and 128 are included in the dataset. Like with DMBC-50, a peak accuracy of 100% is not achieved. The inclusion of APSK and PSK signals results in an accuracy greater than 98% only being achieved at 35dB and 40dB SNR. Furthermore, the rate of average accuracy reduction with respect to the SNR is increased by the inclusion of these signals. The performance of the system on QAM signals remains largely unaffected, the majority of the reduction in performance is a result of the relatively weaker performance of the system on 64APSK and 128APSK bringing the average accuracy downwards.

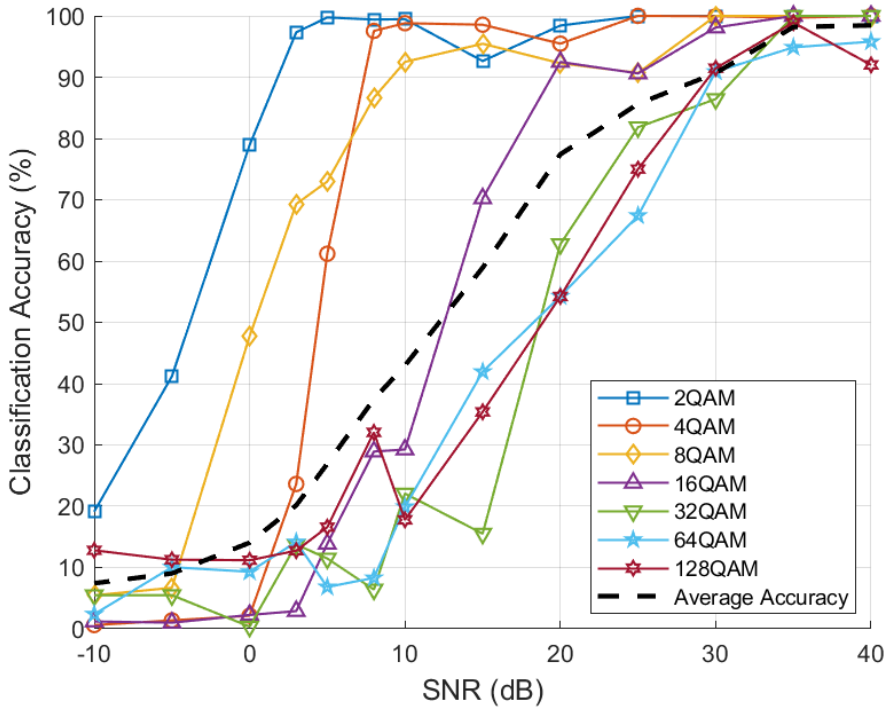


Figure 7.10: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-250 on the Full Dataset*

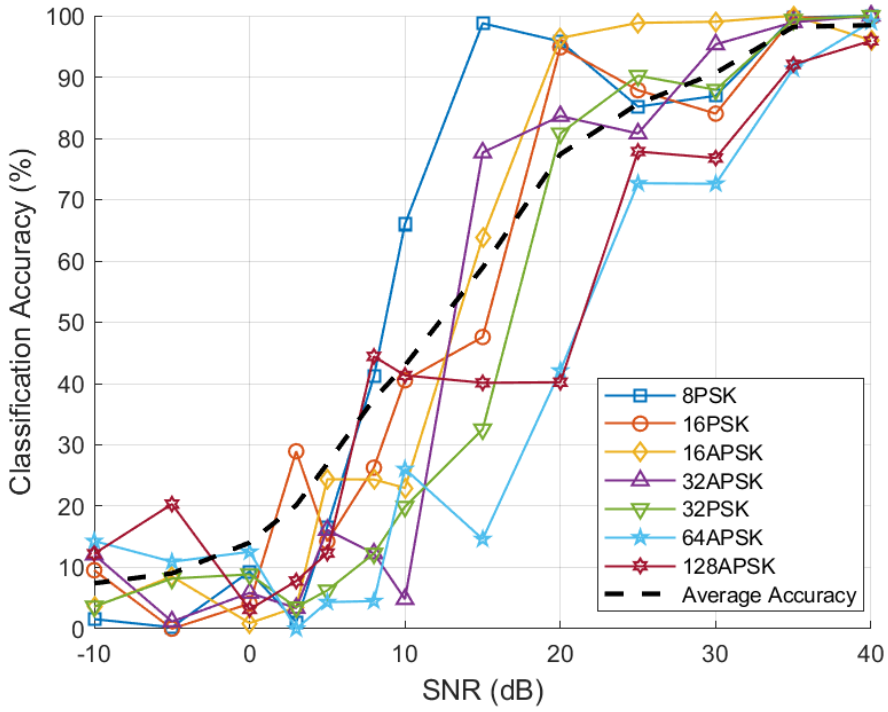


Figure 7.11: *PSK and APSK Classification Accuracy (%) Against SNR (dB) of DBMC-250 on the Full Dataset*

7.3.3 DBMC-500 Classification Performance

DBMC-500 is the second largest implemented DBMC classifier, it is limited to classifying modulation schemes of orders up to and including 256. The utilised argument and magnitude ε were 0.5 and 0 respectively. Figure 7.12 shows the classification accuracy achieved by the system on purely QAM signals.

The system is shown to once again reach 100% classification accuracy at 40dB SNR, over 99% accuracy is also achieved at 35dB SNR. Below this SNR value the average accuracy sees a linear decline as the SNR reduces until -5dB SNR where a rate close to a random guess is reached. The reduction in average accuracy is primarily driven by the inclusion of 256QAM which the system struggles to differentiate from 128QAM. At 25dB 64QAM begins to contribute to the average accuracy reduction, then at 20dB SNR the accuracy of 32QAM begins to rapidly decline. Every 5dB below 25dB SNR another modulation scheme sees a rapid reduction in accuracy. However, despite the reduced average accuracy performance, the accuracy of the system on signals which are common to both DBMC-50 and DBMC-250 is improved. Figures 7.13 and 7.14 show the accuracy when PSK and APSK signals up to respective orders of 32 and 128 are included in the dataset.

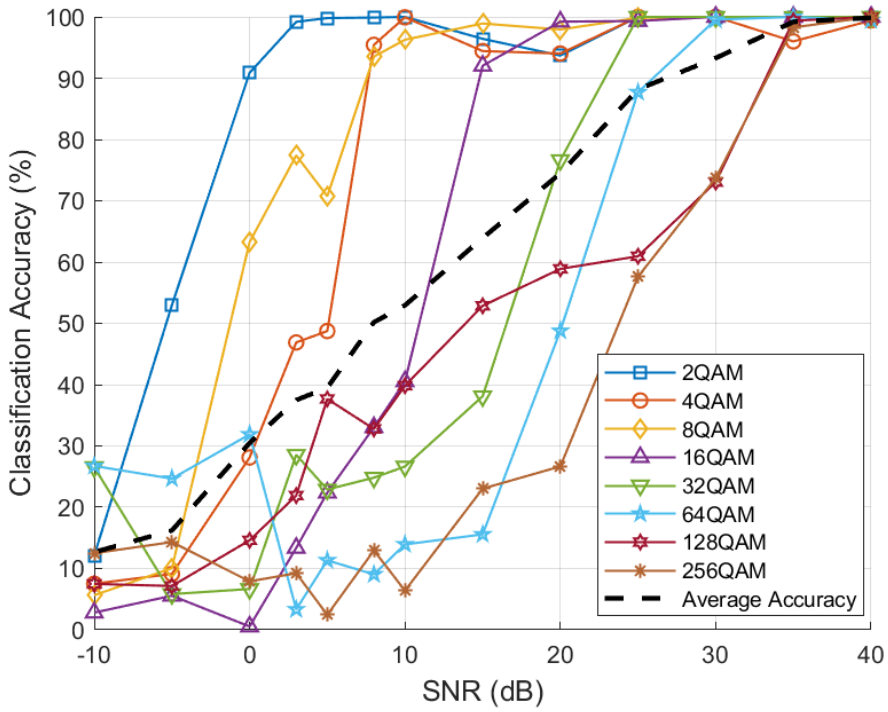


Figure 7.12: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-500 on the QAM only Dataset*

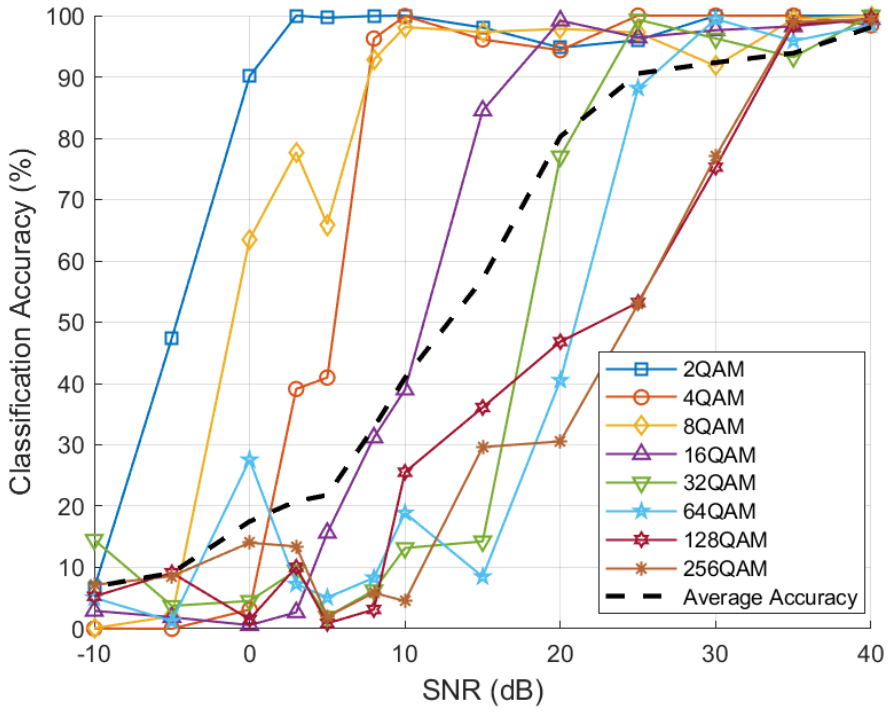


Figure 7.13: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-500 on the Full Dataset*

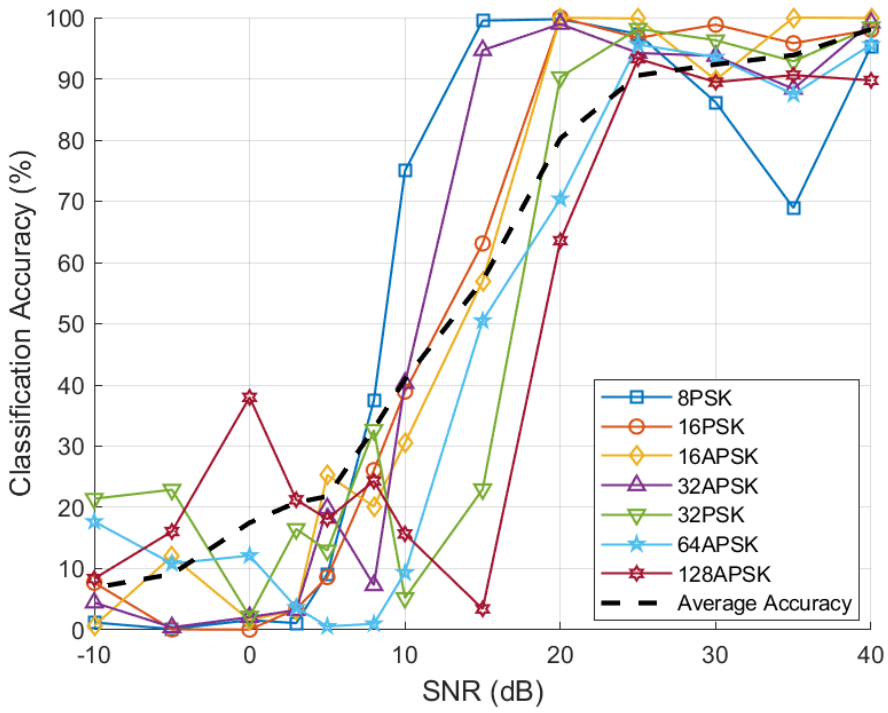


Figure 7.14: *PSK and APSK Classification Accuracy (%) Against SNR (dB) of DBMC-500 on the Full Dataset*

Perfect classification accuracy is not achieved by the system on this dataset, at 40dB an accuracy of only 98.37% is obtained. This is due to the weak performance of the system on the APSK signals at this SNR. However, the accuracy of the system on PSK and APSK signals remains in general above 90% until below 25dB SNR, this contributes to an average classification accuracy of greater than 90% being maintained until this SNR. Between 25dB and 10dB SNR most modulation schemes fall from over 90% accuracy to below 30%, as such there is a steep decline in average accuracy in this region. At 5dB SNR, all signals bar the 3 lowest order QAM signals are classified with sub 30% accuracy. Once again, the performance of the system on QAM signals remains largely unaffected by the inclusion of PSK and APSK signals.

7.3.4 DBMC-1000 Classification Accuracy

DBMC-1000 is the largest implemented classification system, the feature extraction dataset size imposes a limit to operating on signals with a maximum expected number of argument clusters of 500, therefore the highest order modulation scheme included is 512QAM. The utilised argument and magnitude ε values were both 0. Figure 7.15 shows the results obtained when a dataset consisting of only QAM signals is used as the training and test data.

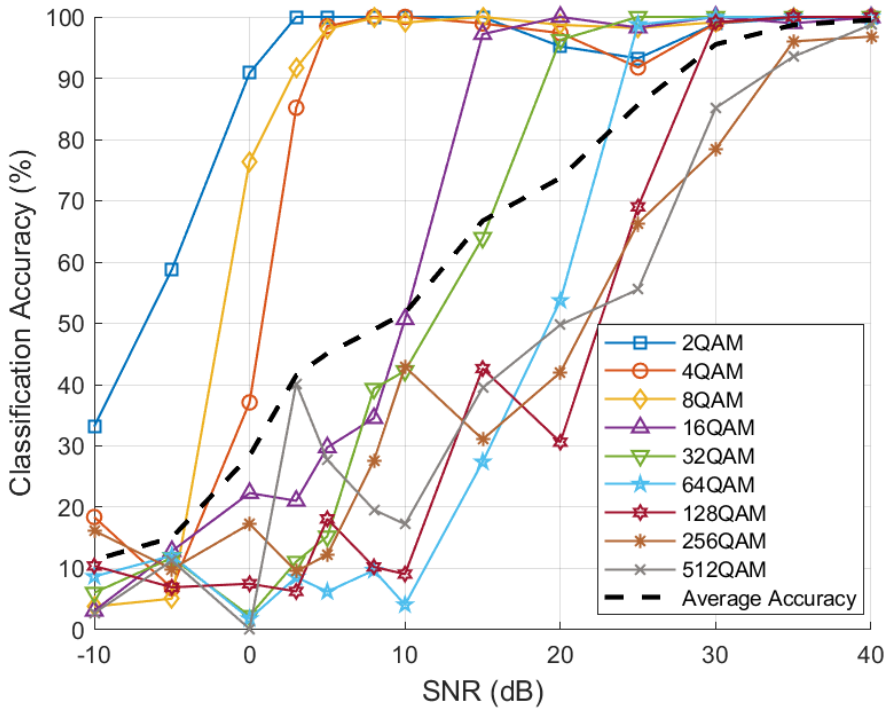


Figure 7.15: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-1000 on the QAM only Dataset*

While 100% accuracy is not achieved, 99.49% average accuracy is obtained at 40dB SNR, an average accuracy greater than 95% is maintained at and above 30dB SNR. Below 30dB SNR the typical linear decline in average accuracy can be seen. At all SNRs the system struggles to differentiate between 256QAM and 512QAM, at an SNR of 30dB and below, 128QAM is also classified with a similar accuracy to these two higher order schemes. The performance on signals shared with DBMC-50, 250, and 500 is improved in all cases. Finally, an accuracy of near equivalence to a random guess is obtained at -5dB SNR. Figures 7.16 and 7.17 show the performance of the system when PSK and APSK signals are included in the dataset.

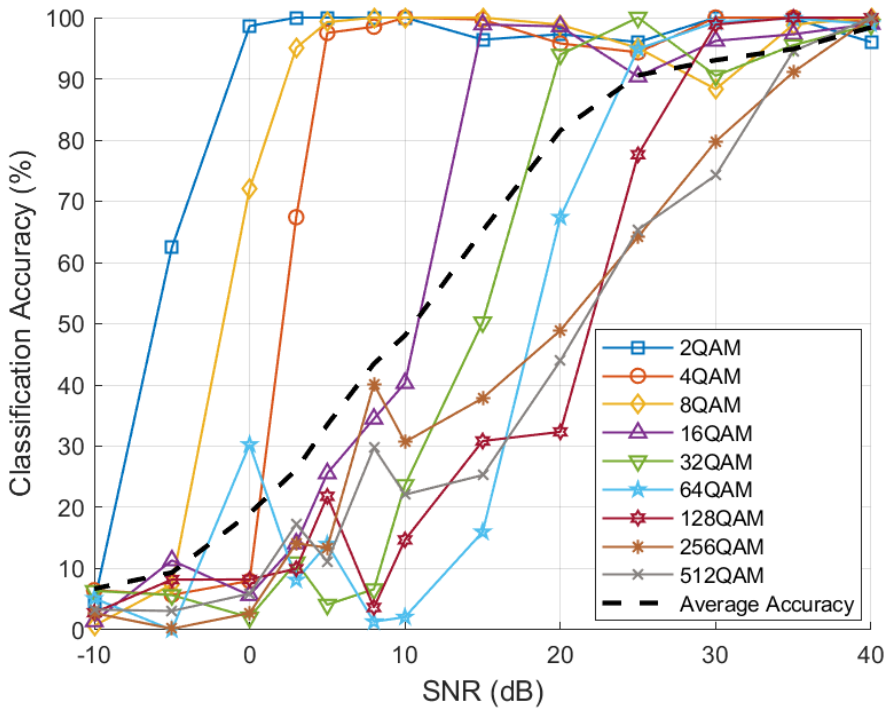


Figure 7.16: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-1000 on the Full Dataset*

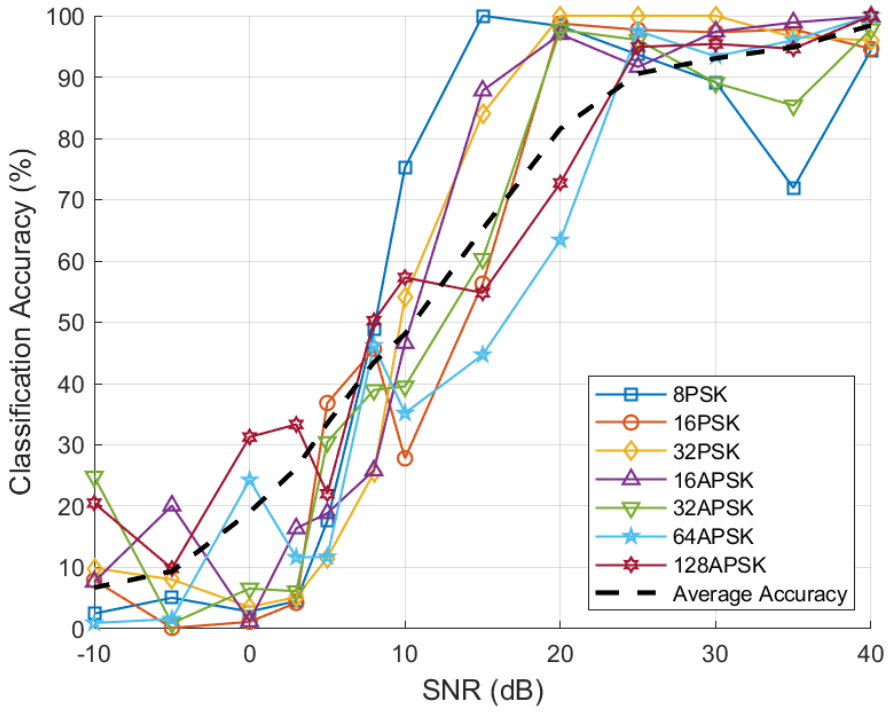


Figure 7.17: *PSK and APSK Classification Accuracy (%) Against SNR (dB) of DBMC-1000 on the Full Dataset*

Again 100% accuracy is not achieved, even at 40dB SNR. In this case the reason is due to the performance on the system on low-order signals. Chapter 5 discussed how the inclusion of very high-order signals in the dataset necessitated optimising ε values for performance on these signals, this is a clear example where optimisation of the system for high-order performance imposes a performance penalty on low-order signals at high SNRs. It was also shown that optimum ε values for high-order signals at high SNRs were similar to that of low-order signals at low SNRs. Therefore, the accuracy of the low-order signal climbs again at SNRs below 20dB. An average accuracy of above 90% is maintained until 25dB, below which the accuracy undergoes the characteristic linear decrease. The accuracy of the high-order QAM signals is unaffected by the inclusion of PSK and APSK signals, however the accuracy of the low-order QAM signals is somewhat affected by misclassification as PSK or APSK at higher SNRs, at SNRs below 20dB this misclassification ceases to occur. Equivalent accuracy to a random guess is once again obtained at approximately -5dB SNR.

7.3.5 DBMC-5000 Classification Accuracy

This is the largest DBMC model which was tested. This model was not implemented in hardware due to being too large to synthesise on the target FPGA, therefore these results were obtained via software simulation and provided as an example of the upper limit of performance of the DBMC classification system. The utilised argument and magnitude ε values were 0.1 and 0.0007 respectively. Figure 7.18 displays the obtained accuracy on a QAM only dataset.

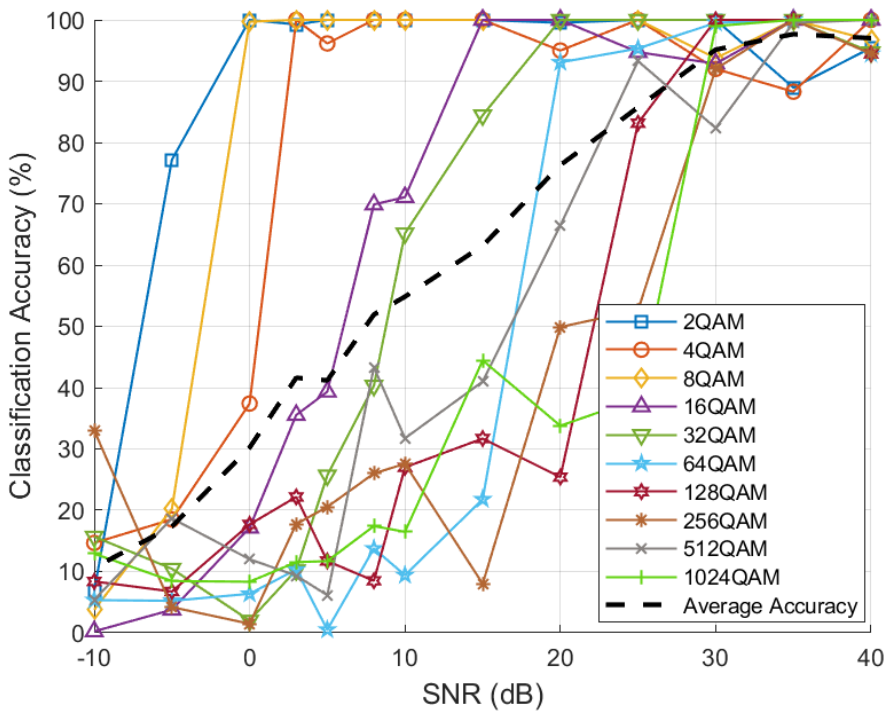


Figure 7.18: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-5000 on the QAM only Dataset*

Peak accuracy once again fails to reach 100%, the highest average accuracy is 97.76% which is obtained at 40dB SNR. The reason for imperfect accuracy is primarily due to 4QAM, 8QAM, and 16QAM exhibiting approximately 95% accuracy at 40dB SNR. This is due to the ε values being optimised for high-order performance at high SNRs. 1024QAM is the first signal to see a large performance reduction, unexpectedly it is 128QAM which is the second. 512QAM maintains over 90% accuracy as low as 25dB SNR, 256QAM momentarily dips at 25dB SNR before returning to 100% accuracy at 20dB SNR and then subsequently falling to 20% accuracy at 15dB SNR. All QAM signals common to the smaller configuration see increased SNR resilience in this case. Equivalent average accuracy to a random guess is obtained at -10dB. Figures 7.19 and 7.20 display the achieved classification accuracy when the full dataset is employed for training and testing

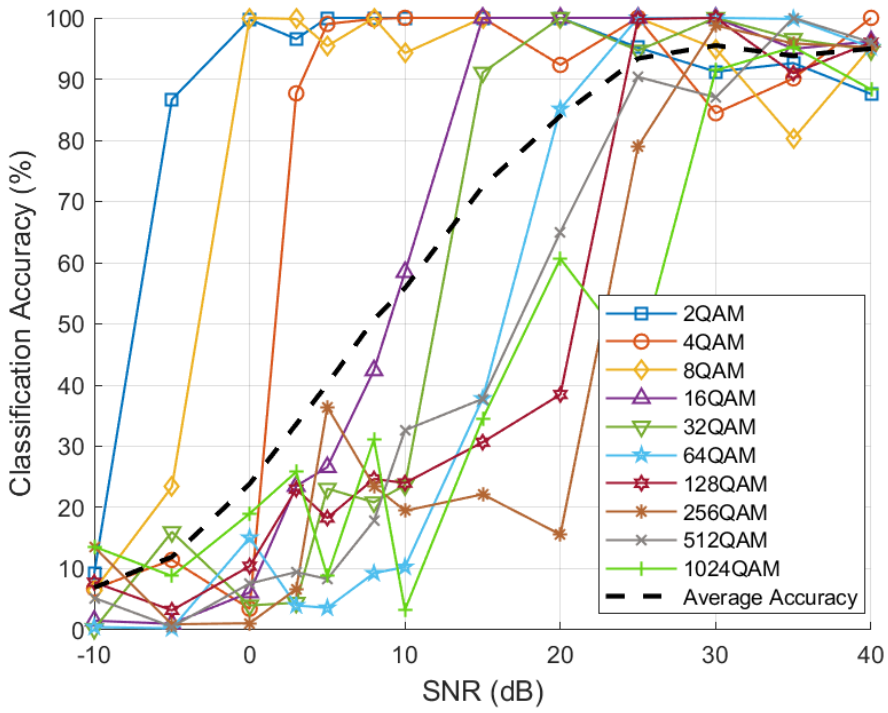


Figure 7.19: *QAM Classification Accuracy (%) Against SNR (dB) of DBMC-5000 on the Full Dataset*

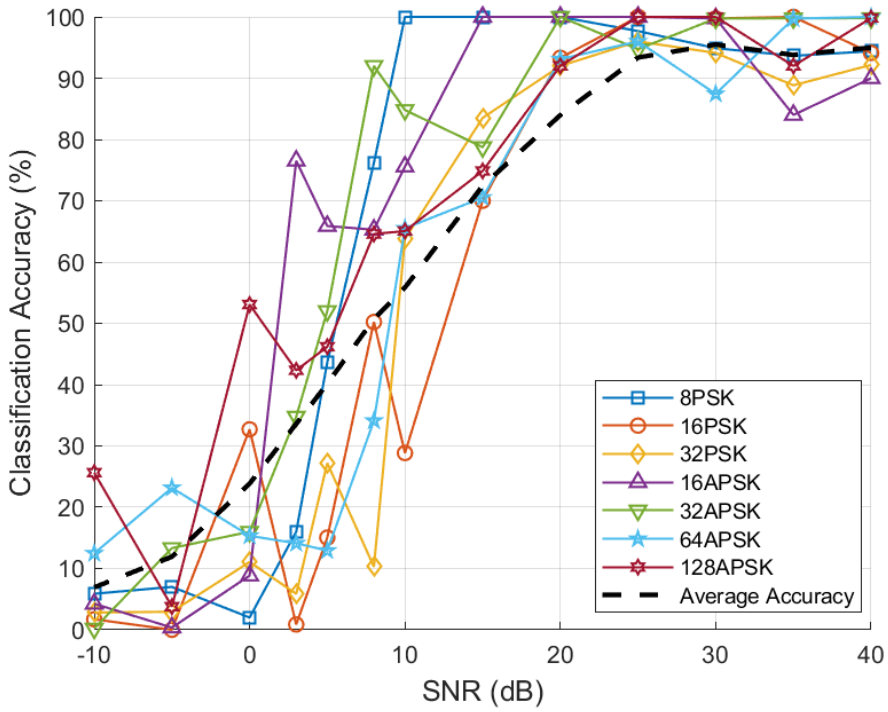


Figure 7.20: *PSK and APSK Classification Accuracy (%) Against SNR (dB) of DBMC-5000 on the Full Dataset*

The peak average accuracy is obtained at 30dB SNR which again is due to the reduced accuracy on low-order signals at 35dB and 40dB SNR, as in other cases this is due to the ε values being tuned to maximise high-order accuracy. 100% accuracy is not obtained, with the maximum achieved average accuracy being 97.15%. The average accuracy remains above 80% as low as 20dB SNR, it is primarily the accuracy of the system when classifying high-order QAM signals bringing the average down, the high-order APSK signals apply upwards pressure to the average accuracy, even as low as 10dB 128APSK is classified correctly with greater than 80% accuracy. The SNR at which the accuracy is equivalent to a random guess is -5dB. The accuracy obtained on the QAM signals is similar to that of Figure 7.18, indicating the addition of PSK and APSK signals did not affect the classifiers performance on QAM signals.

7.3.6 DBMC Performance Discussion

The average results of each DBMC configuration when utilising each dataset are shown in Figure 7.21, Figure 7.22 displays the same results but magnified between 15dB and 40dB SNR.

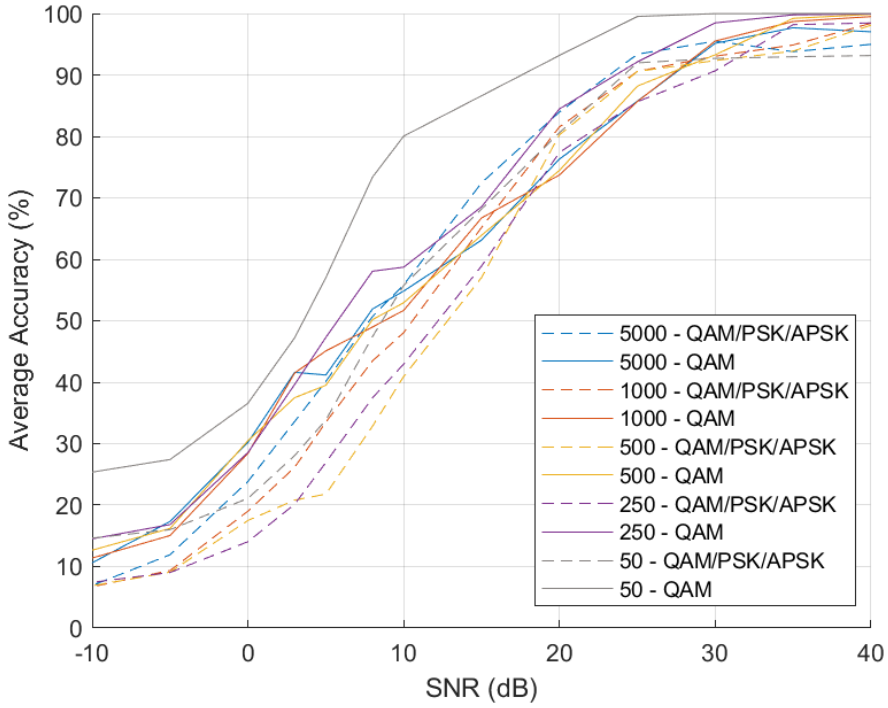


Figure 7.21: *Average Classification Accuracy (%) Against SNR (dB) Achieved by each DBMC Configuration on each Dataset*

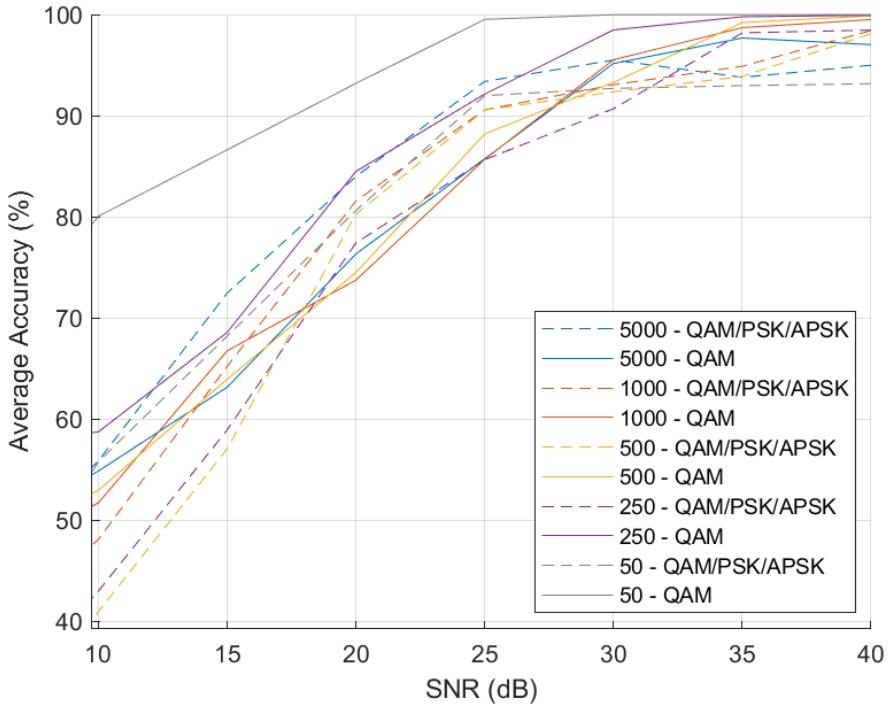


Figure 7.22: *Average Classification Accuracy (%) Against SNR (dB) Achieved by each DBMC Configuration on each Dataset, in the SNR range 10dB to 40dB*

The first trend which can be identified is that all DBMC configurations in general follow a similar sigmoid shaped pattern of accuracy degradation as the SNR decreases. Between 25dB and 40dB SNR the accuracy is in general greater than 90%, below 25dB SNR the accuracy linearly declines until 0dB where the rate of accuracy degradation decreases. The primary exception to this trend is DBMC-50 which achieves significantly greater accuracy at all SNRs than other configurations. This trend demonstrates that when using the rule of including only modulation schemes with expected numbers of argument clusters which are fewer than half the dataset size results in comparable performance across systems. It was found in the previous sections that progressively larger configurations achieved stronger performance on modulation schemes which were common to the datasets of smaller configurations, it would seem that this stronger performance balances the inclusion of comparatively more difficult higher order schemes in the dataset.

Figure 7.22 shows clearly that the inclusion of PSK and APSK signals in the dataset always results in a reduction in peak average accuracy, in every case each configuration obtains a greater peak average accuracy when only QAM signals are utilised. However, in some cases when PSK and APSK signals are included the peak average accuracy or a value close to the peak average accuracy is maintained to a lower SNR than with a QAM only dataset. Therefore, while the QAM only results may achieve a higher peak accuracy at 35dB and 40dB SNR, at 15dB, 20dB and 25dB the PSK and APSK results are higher than their QAM only counterparts. This trend is only observed for the three largest DBMC configurations, suggesting that these configurations are particularly effective at classifying PSK and APSK signals whereas DBMC-50 and DBMC-250 struggle. The trend again reverses below 15dB SNR where the PSK and APSK signals are once again shown to result in reduced average accuracy compared to when only QAM signals are classified. Figure 7.20 clearly illustrates why this is the case, between 30dB and 15dB SNR nearly all PSK and APSK signals are classified with an accuracy greater than the average accuracy curve, above and below these SNR values the inverse is the case.

It can also be seen that while the average accuracy trend of the DBMC systems demonstrate a consistent degredation in accuracy as the SNR decreases, the accuracies of the system on various modulation schemes can increase or decrease between SNRs. This is due to the classifier prioritising particular classes when classes overlap because of random weight value initialisation, a thorough explanation may be found in Section 7.3.

7.3.7 Selecting the Optimum DBMC System

When evaluating which DBMC system is the optimum choice for a desired application, the maximum order of modulation scheme which is expected to be classified should be determined. The maximum number of expected magnitude or argument clusters should then be determined using the methods outlined in Section 6.4.7, the minimum DBMC configuration size is double this value. The hardware utilisation requirements of the target platform should then be evaluated, smaller configurations will naturally result in reduced utilisation, however, the results shown in this chapter demonstrate that larger configurations always provide increased classification accuracy. Therefore, to maximise classification accuracy, the largest DBMC configuration which is compatible with the target hardware platform should be utilised.

7.4 Classification Accuracy Comparison

This section draws comparisons to other hardware implemented modulation classifier structures which have shown to be effective in the literature. The various DBMC sizes will have the average classification accuracy across all compatible modulation schemes for a given system determined, these averages will be plotted against SNR from -10dB to 40dB. It is important to note that a comparison between two modulation classifiers may not be completely fair due to different modulation schemes being included in the training and testing data, the inclusion of lower order modulation schemes and the discounting of modulation schemes of orders upwards of 64 will naturally increase average classification accuracy.

Each DBMC configuration will be compared with the systems which utilise similar datasets, idealised performance of certain DBMC configurations, i.e. when only certain modulation schemes are used in the dataset, will also be provided.

7.4.1 A Note on the Usage of RadioML Datasets

Throughout the literature review and the upcoming classification results comparison, the RadioML.2016.10A and RadioML.2018.01A [41] datasets are frequently discussed as being the datasets which many of the works in the literature utilised as their training and testing data. Ideally this dataset would be utilised in the same manner for this thesis to make comparison to other works easier and more direct, but this cannot be done for a number of reasons.

Firstly, both RadioML datasets contain modulation schemes which are incompatible with the algorithm on which DBMC relies upon, as DBMC relies upon a constellation diagram clustering algorithm, naturally the modulation scheme requires a distinct constellation diagram in order for it to be classifiable, the AM analogue modulation schemes along with OOK, and BPSK all produce the same result out of the two 1D DBSCAN modules making them impossible for the classifier to distinguish between. Secondly, the RadioML dataset is now not recommended by the creators due to the following: “These datasets are from early academic research work in 2016/2017, they have several known errata and are NOT currently used within DeepSig products. We HIGHLY recommend researchers develop their own datasets using basic modulation tools such as in MATLAB or GNU Radio, or use REAL data recorded from over the air!” The data used for our training and testing follows the advice of the RadioML creators as both recorded data and MATLAB generated data has been used. Finally, and most importantly, both RadioML datasets are recorded with significant sample rate offset which results in significant corruption of the constellation diagrams. It was attempted to resample the dataset, but each sample was found to have a varying degree of offset, and with 2,555,904 frames in total performing the resampling process this many times was determined to be impossible.

Due to these issues, it was decided that generating a dataset of the same digital modulation schemes which can be found in RadioML.2018.01A was the optimal method for providing a fair comparison.

7.4.2 Low-Order Comparison

The first comparison to draw is with systems which utilised a maximum modulation order of 16. Included in the comparison are two hardware implementations: HISTO-SVM [20] and the feature-based DT [21], a software classifier by Swami et al. which also utilised cumulants is included due to strong performance [47]. All hardware implemented DBMC configurations are included in the comparison, they were trained and tested on a dataset consisting of BPSK, QPSK, 8PSK, and 16QAM as this array of signals most closely matches what was employed by the works from the literature. Figure 7.23 displays the average classification accuracy against SNR curves for each system.

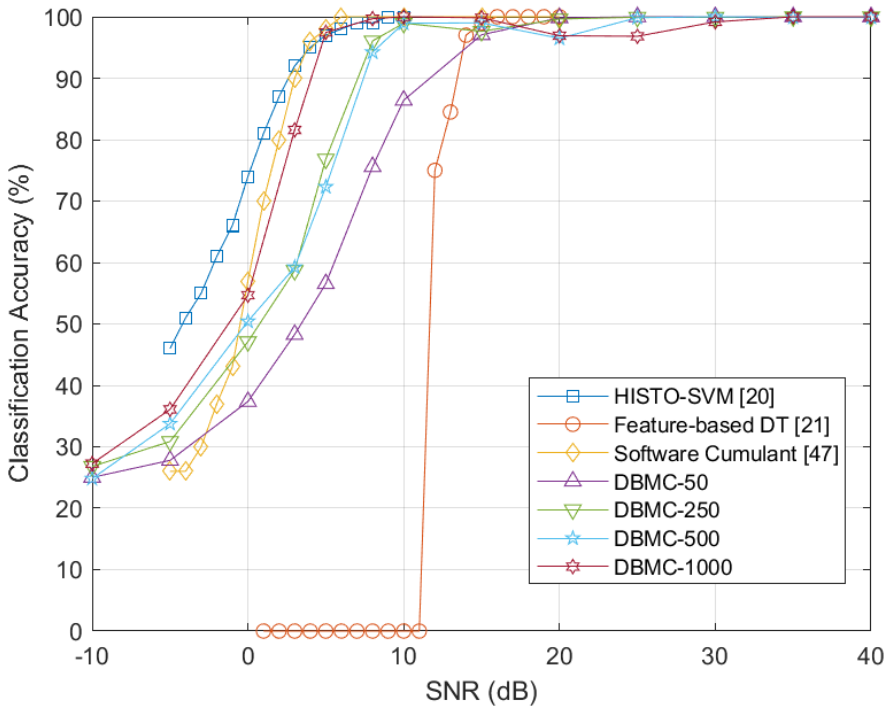


Figure 7.23: *Average Classification Accuracy (%) Against SNR (dB) Achieved by each Hardware-Implemented DBMC Configuration and the Strongest Low-Order Classifiers from the Literature on a Dataset with a Maximum Modulation Order of 16*

Beginning the comparison with the proposed systems, the performance of each DBMC configuration is relative to the feature extraction dataset size as expected. DBMC-50 sees a significant decline below 15dB SNR, DBMC-250 and DBMC-500 exhibit a similar decline below 10dB SNR, DBMC-1000's decline is severe below 5dB SNR. There is no significant difference between the performance of DBMC-250 and DBMC-500. It can also be noticed that DBMC-500 and DBMC-1000 suffer a slight reduction in accuracy between 15dB and 30dB SNR, this is due to the ε parameters being tuned to optimise low SNR performance.

The hardware-implemented feature-based DT [21] is the weakest performing system from the literature in the comparison. DBMC-50 and this system both exhibit their first decline from perfect accuracy at 15dB SNR, however the performance degradation of DBMC-50 is

far less drastic than the cumulant classifier. Overall, it can be concluded that DBMC-50 is the superior classifier due to losing perfect accuracy at the same SNR and having superior SNR robustness. As the other DBMC configurations are superior to DBMC-50 in terms of noise robustness it can be said that they are also superior classifiers to the cumulant classifier.

DBMC-1000 is the only configuration of the proposed method which achieves comparable performance to HISTO-SVM [20] and the software-implemented cumulant classifier [47]. Discounting the temporary loss of perfect accuracy of DBMC-1000 between 30dB and 15dB SNR, both DBMC-1000 and HISTO-SVM lose 100% accuracy at 8dB SNR, but only by a single percentage. At 5dB SNR all three models achieve an average accuracy in the region of 97%. At 3dB SNR DBMC-1000 has an accuracy which is 8% lower than the cumulant classifier and 10% lower than HISTO-SVM. At no point below this SNR does DBMC-1000 outperform HISTO-SVM, however below 0dB DBMC-1000 achieves a higher accuracy than the cumulant classifier. Despite the weaker SNR robustness to the two models from the literature it can be said that DBMC-1000 is at least on par in terms of the SNR at which perfect classification accuracy is lost.

7.4.3 Medium-Order Comparisons

This section compares the performance of DBMC systems to classifiers from the literature which provided results when utilising a dataset with a maximum modulation order of 64. There were no examples of hardware-implemented classifiers which utilised a dataset such as this, therefore this comparison will be to the strongest performing software-based models. In the literature review the LSTM [15] was found to be the best performing I/Q accepting deep learning model, FiF-Net [57] was determined to be the best image-based deep learning classifier, and the software-implemented cumulant classifier proposed by Zhou et al. [48] was found to be the strongest feature-based classifier. Therefore, each of these models are included in the comparison figure. The dataset employed to generate the DBMC results was designed to match the digital signals found in the RadioML.2016.10A dataset, it featured 2QAM, 4QAM, 8PSK, 16QAM, and 64QAM. DBMC-250 is the smallest configuration which is capable of classifying a dataset which includes 64QAM. Figure 7.24 displays the results of the DBMC systems alongside the systems from the literature.

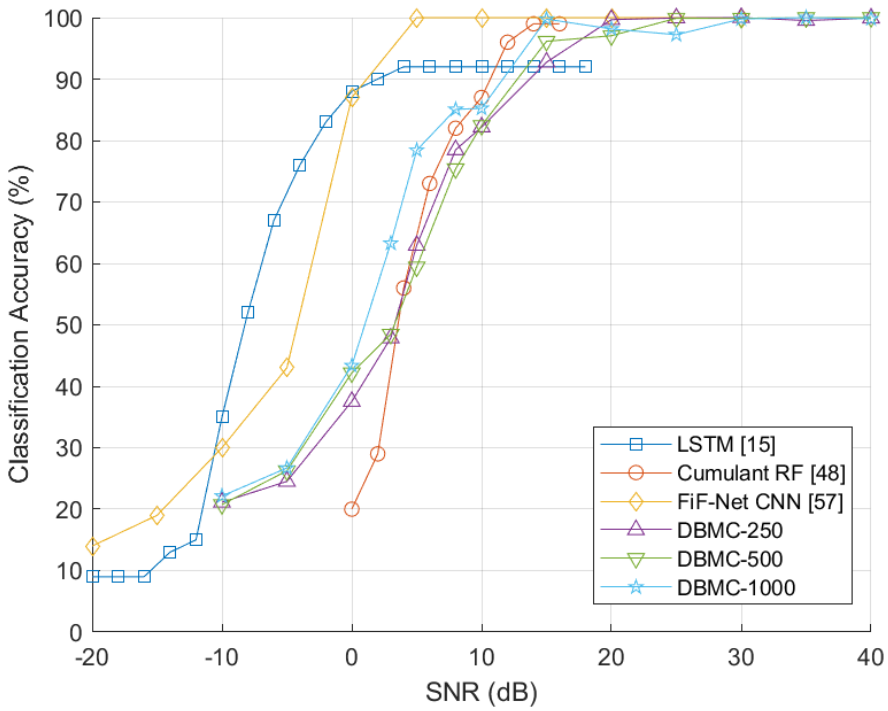


Figure 7.24: *Average Classification Accuracy (%) Against SNR (dB) Achieved by the 3 Largest Hardware-Implemented DBMC Configurations and the Strongest Medium-Order Classifiers from the Literature on a Dataset with a Maximum Modulation Order of 64*

There is no DBMC configuration which maintains its peak accuracy to as low an SNR as either DL-based classifier as both examples maintain this level of performance until approximately 4dB SNR. Where DBMC models have the advantage over the DL model is the value of the peak accuracy, the I/Q accepting CNN can only reach a maximum average accuracy of 92% whereas DBMC reaches 100% in all cases. The results exhibited by the DBMC models are more comparable to the feature-based classifier [48]. All DBMC configurations are demonstrated to be capable of reaching 100% accuracy whereas the cumulant classifier only reaches 99%, a minor difference overall. The two smaller DBMC configurations lose 100% accuracy 6dB earlier than the system from the literature, discounting the slight reduction in accuracy shown by DBMC-1000 above 15dB, this system maintains peak accuracy to an SNR 1dB greater than the cumulant classifier. The rate of accuracy degradation is similar between all feature-based classifiers, [48] has superior accuracy between 15dB and 10dB SNR, below 10dB SNR DBMC-1000 exhibits a higher accuracy.

To conclude this comparison, no DBMC configuration achieves the low SNR performance exhibited by the DL-based classifiers, but all configurations improve upon the peak accuracy achieved by the LSTM. DBMC-1000 achieved comparable results to the feature-based classifier, DBMC-250 and DBMC-500 matched the high SNR and low SNR performance but lost their peak accuracy 6dB and 11dB higher. ModNet is the strongest classifier in the comparison owing to achieving 100% accuracy and maintaining this level of performance above 3dB SNR.

7.4.4 High-Order Comparisons

Next the DBMC systems will be compared with models from the literature which are demonstrated to capable of classifying orders up to 256. The dataset used to generate the DBMC results used all QAM, PSK, and APSK signals which are included in the RadioML.2018.01A dataset as this is the dataset used by the majority of systems in the comparison. The smallest DBMC model included in the comparison is DBMC-500 as this was the smallest model found to be capable of classifying signals up to an order of 256.

The results from the literature are taken from the strongest performing software and hardware models. RUNet [39] and MobileNetV3 [18] were found to be the hardware implemented models which achieved the strongest performance. ResNet, CNN/LSTM with attention, and the Subtractive Clustering algorithm-based classifier [51] were the only software models that utilised a dataset which included modulation schemes of orders this high. Figure 7.25 displays the average accuracy against SNR of each system in the comparison.

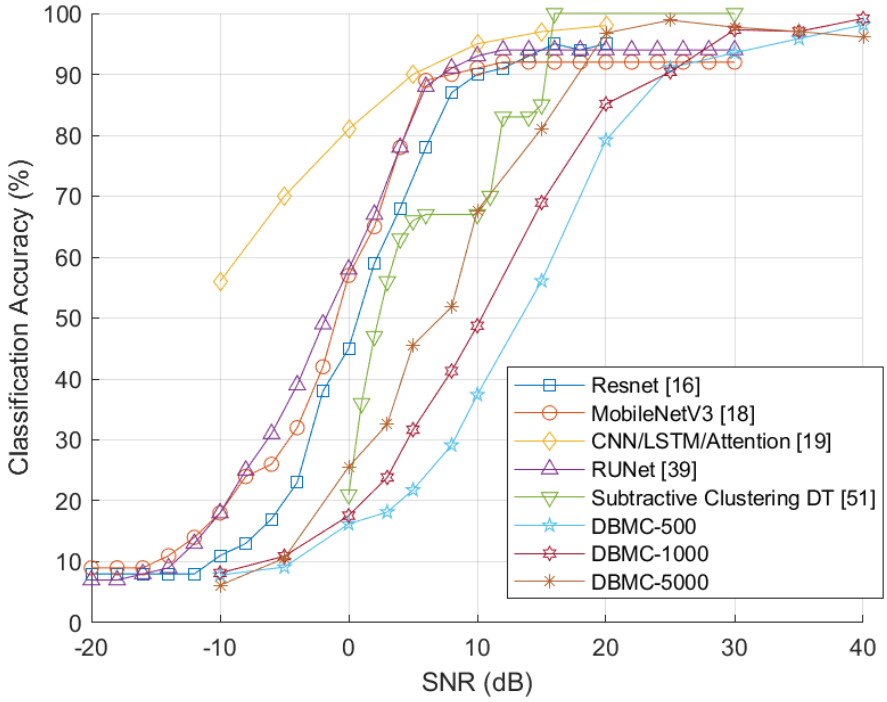


Figure 7.25: *Average Classification Accuracy (%) Against SNR (dB) Achieved by the 2 Largest Hardware-Implemented DBMC Configurations, DBMC-5000, and the Strongest High-Order Classifiers from the Literature on a Dataset with a Maximum Modulation Order of 1024*

Beginning with the hardware implemented models, DBMC-500 and DBMC-1000 both achieve a higher peak average accuracy than RUNet and MobileNetV3, however this advantage is limited to SNRs of 30dB and higher. Below 30dB each DBMC configuration exhibits a significant reduction in average accuracy, demonstrating inferior robustness to noise. The DBMC-5000 model which was not implemented in hardware maintains a higher accuracy above 20dB SNR but again is shown to have reduced robustness to noise. The same conclusions may be drawn when comparing to ResNet and the CNN/LSTM with attention, compared to both

models the DBMC systems all achieve a higher peak accuracy but do so at very high SNRs. The model with which there is a greater similarity in terms of accuracy across SNR is the clustering-based model, here the clustering-based model achieves 100% accuracy and maintains this performance level above 15dB SNR. The trend of accuracy degradation is similar to that of DBMC-1000 but generally a higher classification accuracy is achieved by the model from the literature. At all SNRs the clustering-based classifier outperforms DBMC-500 and DBMC-1000, although it does so using a more limited dataset consisting of only QAM signals with orders 4 to 256. A more direct comparison between clustering-based methods and DBMC models will be provided in the following section.

7.4.5 5G Dataset Performance Comparisons

The comparisons provided in the previous sections utilise DBMC results where the dataset is designed to match what was utilised by the systems in the literature to enable a more direct comparison. This section explores realistic performance of DBMC as if were utilised within a 5G or 6G CR scenario. 5G currently employs 4QAM, 16QAM, 64QAM, and 256QAM, each modulation scheme is utilised for differing channel conditions and data rate requirement scenarios [2]. The comparisons provided in this section are not necessarily fair as the systems from the literature were tested with a wider array of different signals, these results are primarily to demonstrate the maximum level of performance which can be achieved by the DBMC system. Figure 7.26 shows the obtained average accuracy when DBMC-500, DBMC-1000, and DBMC-5000 are trained and tested on the QAM signals utilised in 5G alongside the results of some of the strongest high-order classifiers from the literature.

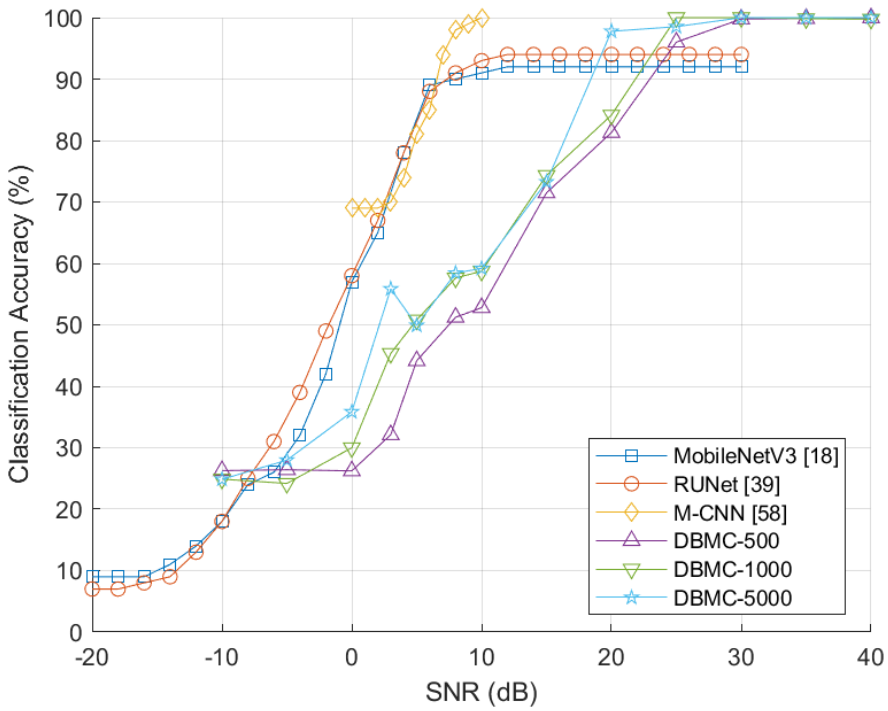


Figure 7.26: *Average Classification Accuracy (%) Against SNR (dB) Achieved by the 2 Largest Hardware-Implemented DBMC Configurations, DBMC-5000, and the Strongest High-Order Classifiers from the Literature on a Dataset which includes 4QAM, 16QAM, 64QAM, and 256QAM*

When employing the 5G modulation scheme dataset the average accuracy achieved by the DBMC systems are improved at all SNRs. All systems achieve 100% accuracy at an SNR of 30dB and higher, superior accuracy to the two hardware-implemented CNNs is maintained as low as 25dB SNR for DBMC-500 and DBMC-1000 and 20dB for DBMC-5000. However, the robustness to noise is still inferior to the systems from the literature by a significant margin. However, in Section 7.1.1 it was demonstrated that QAM, PSK, and APSK signals form their own overlapping clusters in feature space. By using 16PSK rather than 16QAM and 64APSK rather than 64QAM the results shown in Figure 7.27 are obtained.

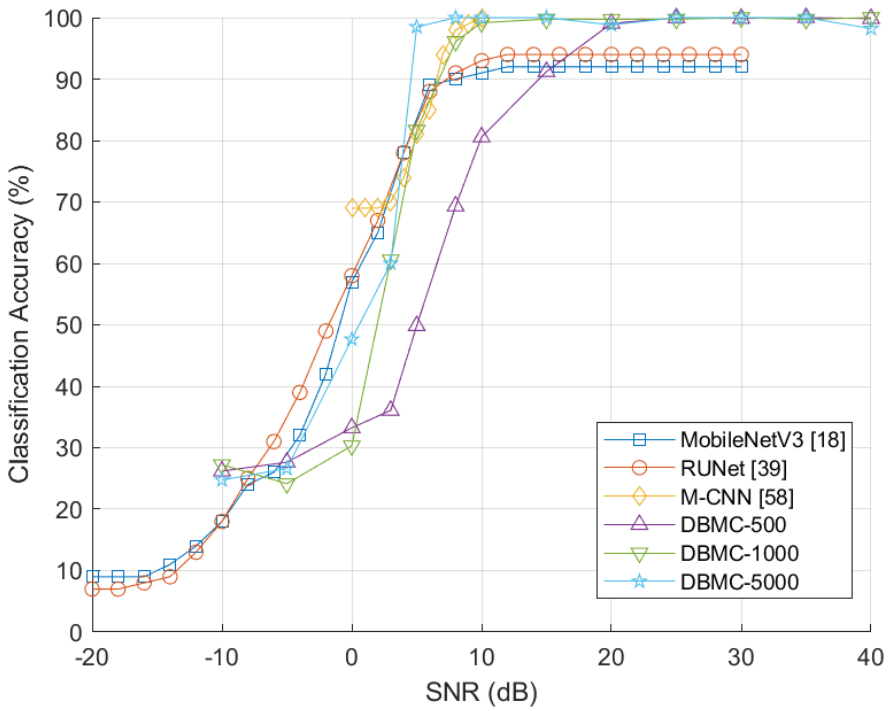


Figure 7.27: *Average Classification Accuracy (%) Against SNR (dB) Achieved by the 2 Largest Hardware-Implemented DBMC Configurations, DBMC-5000, and the Strongest High-Order Classifiers from the Literature on a Dataset which includes 4QAM, 16PSK, 64APSK, and 256QAM*

In this case 100% accuracy is maintained as low as 10dB SNR for DBMC-1000 and 8dB DBMC-5000. These results mean that the two largest DBMC configurations outperform the two I/Q accepting CNNs at all SNRs greater than 5dB and maintain peak accuracy to a lower SNR. DBMC-5000 is even found to maintain over 95% accuracy to a lower SNR than any high-order classifying system which can be found in the literature. DBMC-1000 is shown to closely match the accuracy achieved by M-CNN, the high-order classifier which maintains 100% accuracy to the lowest SNR of any system.

The caveat to the results shown in Figure 7.27 is that they require the usage of a particular set of modulation schemes. As 5G systems are currently only equipped with the hardware to transmit data with QAM, upgrades to infrastructure would be required to accommodate the usage of 64APSK and 16PSK. The results however do demonstrate that performance equivalent to the strongest DL-based classifiers can be matched or even improved upon by DBMC systems if these modulation schemes are employed. Therefore, perhaps if future generations of communications standards include the functionality to utilise the stated modulation schemes, then DBMC would be an extremely viable candidate technology for achieving lightweight, quick, and accurate CR functionality.

7.4.6 Clustering Classifier Comparisons

The performance of the various DBMC configurations has now been compared to the strongest classifiers from the literature in multiple test scenarios. This final comparison section will

evaluate how the proposed modifications made to the DBSCAN algorithm have affected the classification performance by drawing comparisons to other clustering-based modulation classifiers. The 5G dataset was used to obtain the results presented in this section. Figure 7.28 compares the performance of DBMC-500, DBMC-1000, and DBMC-5000 with the 2D DBSCAN [37] and Subtractive Clustering classifier [51].

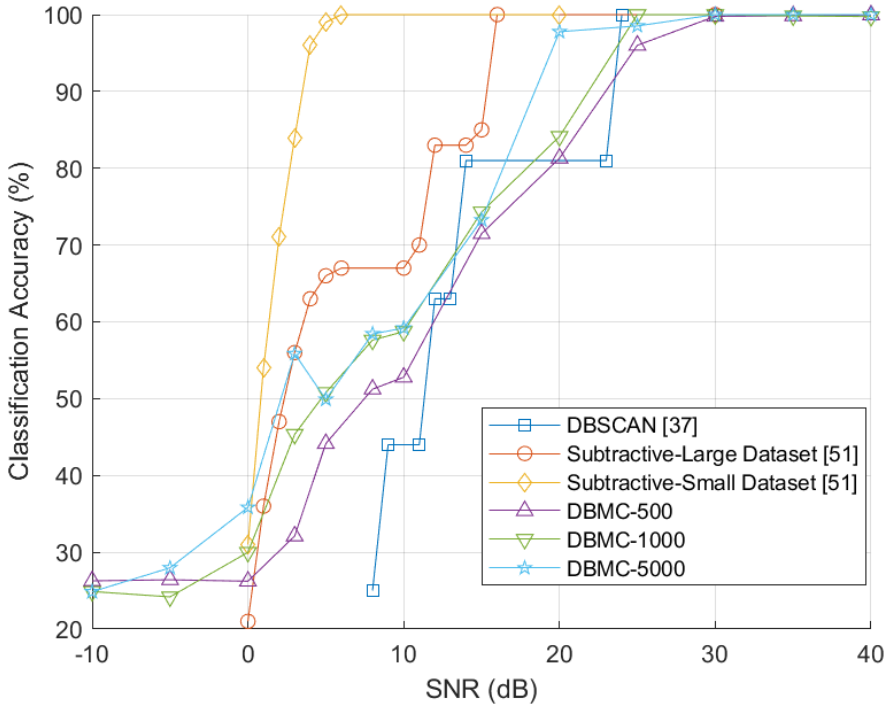


Figure 7.28: *Average Classification Accuracy (%) Against SNR (dB) Achieved by the 2 Largest Hardware-Implemented DBMC Configurations, DBMC-5000, and the Clustering-Based Classifiers from the Literature on a Dataset which includes 4QAM, 16QAM, 64QAM, and 256QAM*

Figure 7.28 shows that the average classification accuracy all DBMC configurations are approximately equal to that of the 2D DBSCAN modulation classifier. DBMC-1000 maintains 100% accuracy to an SNR 1dB higher than the 2D DBSCAN classifier, although it may be the case that both systems achieve this level of accuracy at the same SNR as there is no datapoint at 24dB SNR for DBMC-1000. DBMC-5000 achieves 97.8% accuracy at 20dB whereas the 2D DBSCAN classifier drops to 81% accuracy at this SNR. DBMC-500 loses perfect accuracy below 30dB SNR. Below 20dB SNR all DBMC systems follow a rate of accuracy degradation which is approximately equivalent, the 2D DBSCAN classifier's rate of degradation is stepped which results in either a greater or inferior accuracy being achieved a certain SNRs. The 2D DBSCAN algorithm reaches an equivalent accuracy to a random guess at 8dB SNR, 8dB higher than DBMC-500.

The fact that all DBSCAN based systems lose 100% accuracy at a similar SNR suggests an inherent limitation of the algorithm, despite the differing mechanisms for feature extraction and optimisation techniques all systems achieve similar performance in this regard. The difference between the rates of accuracy degradation may be attributed to the

employed classifier model, the 2D DBSCAN system used a decision tree with hard decision thresholds, once the extracted number of clusters for a particular modulation scheme passed below a certain threshold the accuracy of the system at classifying that particular modulation scheme drops to 0% which explains the stepped nature of the accuracy reduction. Perhaps by employing a classifier capable of finding nonlinear decision boundaries the 2D DBSCAN system could match the low SNR performance of DBMC. DBMC was designed to be capable of classifying PSK and APSK signals as well as QAM signals which is a capability that the 2D DBSCAN algorithm does not possess, this comparison demonstrates that the inclusion of this functionality does not sacrifice performance on QAM signals. Furthermore, the 2D DBSCAN algorithm used feature extraction dataset sizes of 10,080 which is over double that of DBMC-5000, DBMC is therefore demonstrated to have the capacity to achieve comparable performance with a dataset size which is over 10x smaller in the case of DBMC-1000.

Despite matching the performance achieved by the DBSCAN clustering algorithm from the literature, no DBMC configurations are able to match the performance achieved by the subtractive clustering algorithm. Even with the larger dataset which includes 32QAM and 128QAM the subtractive clustering algorithm system achieves equal or superior accuracy above an SNR of 3dB. When utilising the small dataset, which is equivalent to what was used by DBMC, the performance far exceeds that of the DBMC systems, notably 100% accuracy is maintained as low as 6dB SNR. However, similarly to the results shown in the previous section, if 16PSK and 64APSK are used instead of 16QAM and 64QAM, the accuracy achieved by DBMC-5000 can be made to approximately match that of the subtractive clustering algorithm, this is shown in Figure 7.29.

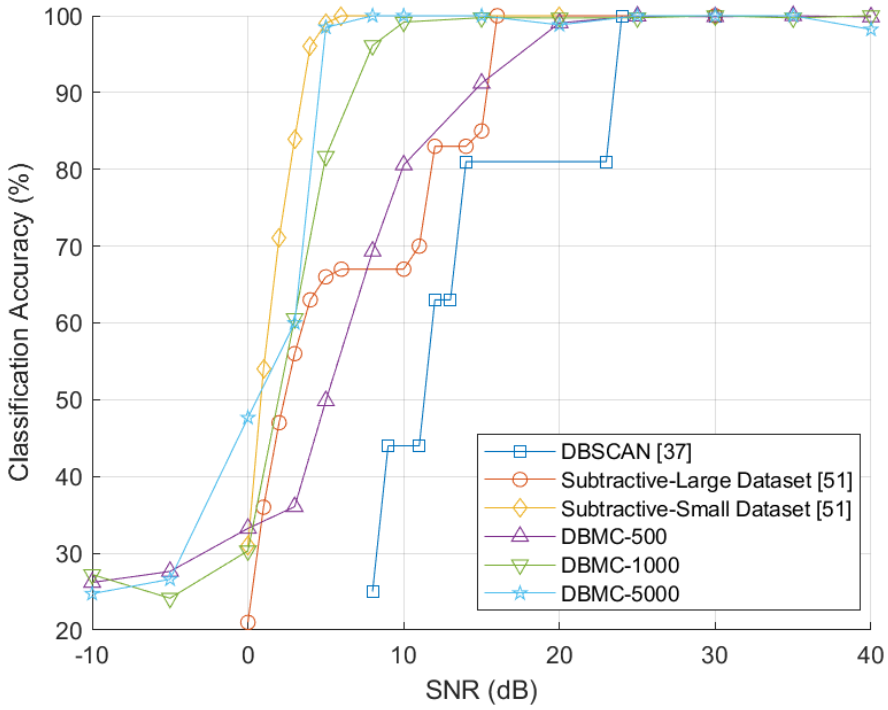


Figure 7.29: *Average Classification Accuracy (%) Against SNR (dB) Achieved by the 2 Largest Hardware-Implemented DBMC Configurations, DBMC-5000, and the Clustering-Based Classifiers from the Literature on a Dataset which includes 4QAM, 16PSK, 64APSK, and 256QAM*

Figure 7.29 demonstrates that when employing the dataset consisting of 16PSK and 64APSK the average accuracy achieved by DBMC-5000 can match that of the subtractive clustering algorithm. DBMC-1000 and DBMC-500 cannot achieve 100% accuracy to as low an SNR as the subtractive algorithm, however DBMC-1000 does come close. While the datasets are different, the orders of the employed modulation schemes are equivalent. These results would not be possible without the modifications to the DBSCAN algorithm proposed in this thesis, this fact shows that proposed modifications enable the DBSCAN algorithm to achieve a level of performance which would otherwise be impossible. Thus, the proposed algorithm results in a marked improvement over the traditional algorithm under certain conditions.

7.5 Classification Comparison Conclusion

The various DBMC configurations have now had their average classification accuracy compared with the best performing systems found in the literature.

No hardware implementations could be found which employ a maximum modulation order of 64 in the training and testing dataset, therefore comparisons were drawn to the strongest software-based implementations. It was found that DBMC-250, DBMC-500, and DBMC-1000 could match the accuracy of the strongest feature-based system at all SNRs. However, no DBMC configuration could match the noise robustness exhibited by DL modulation classifiers, although 100% accuracy was obtained which the LSTM [15] failed to reach.

Comparisons with systems from the literature tested with high-order datasets found that while DBMC achieved a higher peak accuracy than the DL models, the robustness to the effects of noise were once again inferior. DBMC-5000 achieved a greater accuracy than RUNet [39] and MobileNetV3 [18] above 20db SNR, DBMC-500 and DBMC-1000 only achieved this feat above 30dB SNR. The algorithm to which the performance of the DBMC systems was most similar was the subtractive clustering feature extraction classifier.

Section 7.5.5 presented another high-order comparison, but the DBMC results were obtained using a dataset consisting of the QAM signals utilised in 5G communications. The performance of DBMC-500 and DBMC-1000 was somewhat improved but an accuracy greater than that of the I/Q accepting CNN classifiers was only achieved at an SNR of 25dB and higher. It was then shown that by utilising 16PSK and 64APSK instead of 16QAM and 64QAM the average accuracy of DBMC-1000 was greater than that of the I/Q accepting CNNs at all SNRs above 5dB, the performance as also equivalent to the constellation diagram accepting M-CNN [58] at all SNRs greater than 3dB.

The final comparisons were made with other clustering feature extraction methods to evaluate how the proposed changes to the DBSCAN algorithm affected performance. When using the 5G QAM dataset it was found that all DBMC configurations of sizes greater than 500 achieved approximately equivalent classification performance to a 2D DBSCAN modulation classifier [37] as 100% accuracy was maintained to an SNR of 24/25dB. Despite the DBSCAN modifications and proposed enhancements to the hyperparameter optimisation process, the accuracy of DBSCAN could not be made to match that of the Subtractive Clustering algorithm. However, by once again employing 16PSK and 64APSK the peak accuracy achieved by DBMC-5000 was maintained to an equivalent SNR as the subtractive clustering algorithm. DBMC-1000 also saw a large improvement when utilising this dataset as near perfect accuracy was maintained as low as 10dB SNR.

The findings from these comparisons show that regardless of the maximum order of modulation scheme included in the dataset, the largest DBMC systems can match or outperform all feature-based classifiers apart from the subtractive clustering algorithm. With a maximum order of 16 DBMC-1000 matched the accuracy of HISTO-SVM [20] and the strongest software-based cumulant classifier [47]. With a maximum order of 64 DBMC-1000 matched the accuracy achieved by a cumulant classifier [48]. DBMC-5000 came close to matching the performance of the Subtractive Clustering algorithm [51] with a maximum modulation order of 256.

The comparisons have also shown that DL-based classifiers can classify at their peak accuracy to a much lower SNR than the DBMC systems. DL classifiers generally maintain peak accuracy down to 10dB SNR whereas DBMC loses peak accuracy between 20dB and 35dB. DBMC has been shown to be capable of reaching 100% classification accuracy while I/Q accepting DL classifiers only reach a maximum of 92-94%, however constellation diagram accepting CNNs are shown to be able to achieve 100% accuracy as low as 10dB SNR. The tests when utilising the 5G QAM signal set showed a slight improvement in accuracy for the DBMC systems, only when the dataset was specifically constructed to maximise accuracy by including 16PSK and 64APSK did DBMC-1000 and DBMC-5000 match the noise robustness of the DL systems as 100% accuracy was maintained until 10dB and 8dB SNR respectively. Therefore, DL classifiers have been found to be more versatile classifiers than DBMC owing to the stronger performance on more varied and larger datasets, however the performance of

DBMC with the 16PSK and 64APSK 5G dataset demonstrates that equivalent performance can be achieved by the proposed model in a 5G CR scenario.

Chapter 8

NDA SNR Estimation Performance

The DBSCAN algorithm was selected as the focus for development in part because it had previously been shown to provide SNR estimation functionality. As discussed in the introduction to this thesis, SNR estimation is a key function which a CR enabled system should be capable of performing. This SNR estimator reuses the hardware of the modulation classifier and therefore a single implementation of the DBMC system can perform both tasks, the following section will outline the minor but necessary modifications to realise this functionality, comparisons to other SNR estimator's hardware implementations will be provided in case this system is required to be implemented in the absence of DBMC functionality. Following this, the operation of the SNR estimator will be described and explained. Finally, the accuracy of the SNR estimator in comparison to other methods found in the literature will be presented.

8.1 SNR Estimator Hardware

SNR estimation functionality may be realised with the same hardware structure as modulation classification; therefore Chapter 5 provides a detailed description of the structure of the SNR estimation implementation.

Despite the reuse of hardware, two modifications to stored values within the system are required to enable SNR estimation functionality:

- Tuning ε to optimise between SNR Estimation/modulation classification functionality, specific ε values are required for each modulation scheme in SNR estimation mode.
- Tuning MLP weights between SNR Estimation/modulation classification functionality, again specific weight values are required for each modulation scheme in SNR estimation mode. The MLP operates as a regressor in this case, thus all but one output node weight value is set to 0.

Both modifications are to values which are stored in LUTs, to achieve SNR estimation functionality all that is required is a change in the values contained within the LUTs. Ideally, the proposed system structure would be capable of switching between sets of weight and ε values depending upon the function which is required to be performed, however this functionality was not implemented due to time constraints. Enabling the ability to switch between functionalities requires storing all the ε and weight values simultaneously. Additionally, logic and

control hardware would be required to manage loading of the required parameter values which enables switching between functions. Storing an array of different values has a minor effect upon the hardware implementation size, in comparison the rest of the system the increase is negligible.

To demonstrate the required increase in utilisation, the required additional storage for the largest 17 output node MLP configuration is given: there are 57 weights and biases required to operate the MLP, each stored as a signed 16-bit fixed point value, similarly, the two ε values are stored as unsigned 10-bit fixed point values. In total there is therefore 932 bits of data required to achieve a classification function. However, the SNR estimation operation requires the MLP operate as a regressor, this functionality requires only a single output node and thus only 6 weights, 3 biases, and 2 ε values would be required to be stored per modulation scheme, a total of 164 bits. An SNR estimator with this MLP configuration would be expected to operate on 17 different modulation schemes and estimating the SNR of differing modulation schemes requires a different set of parameters. Therefore, to obtain the functionality to estimate the SNR of 17 modulation schemes requires the storage of 17 sets of 164 bits of data, which is equal to 2788 bits. Including the values required for AMC necessitates an additional 928 bits per SNR, bringing the total to 14,852 bits or 1.8565KB of storage. These additional weight, bias, and ε values could be stored as ROM within the FPGA fabric or in memory, whichever is preferable to the designer.

The complexity of the control hardware required to manage the selection of the required values for a particular SNR estimation or modulation classification task can vary depending on how a user desires the system to operate. For example, an implementation which requires more direct control from a user could feature a method of user input such as a keypad which allows for the selection of their desired functionality, the input would be connected to the LUTs which contain the stored values and outputs of said LUTs would connect to the MLP and DBSCAN modules, the user input would therefore control which stored values are used as MLP weights and ε. A more automatic implementation may alternate between modulation classification and SNR estimation, differing ratios of modulation classification to SNR estimation operations may be performed with this method, for example 10 modulation classification operations could be performed, the most frequently occurring classification result could then be taken and the required weights and ε values for said modulation scheme could then be loaded, 5 SNR estimation operations could then be performed and the average of the results may be taken as the SNR. This could be implemented with a state machine controlling the LUT output as well as additional registers to hold the most recent modulation classification outputs.

8.2 Prior DBSCAN SNR Estimation Mechanisms

One of the reasons why DBSCAN was selected as the feature extraction mechanism in this work was that it had previously been demonstrated to have the ability to provide SNR estimation capabilities as well as AMC functionality. The work which used DBSCAN in this manner was created by Zhao et al. [38], they used the ratio shown in Equation 8.1 as a measure of constellation point density:

$$R = \frac{|\Omega|}{|D|} \tag{8.1}$$

Where Ω represents the number of core points in D, D represents the whole set of datapoints, and $||$ denotes the total number of each variable. As explained in Section 3.6.1, DBSCAN labels core points as being clustered points which have at least minPts other datapoint within their ε neighbourhood. As SNR decreases the density of constellation points also decreases, this ratio therefore provides a measure of the density of the constellation diagram as naturally the number of core points decreases as the density decreases. This is shown in Figure 8.1 where 16QAM at 20dB and 10dB can be seen, it can be seen how the majority of the datapoints at 20dB are found to be core points whereas at 10dB the majority of the datapoints are not.

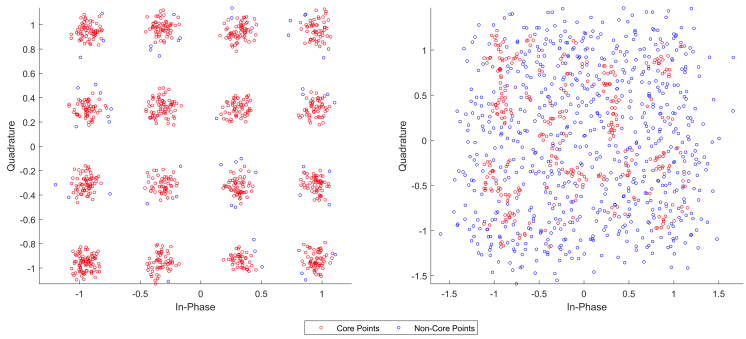


Figure 8.1: *The Core Points Found by DBSCAN on the 16QAM Constellation Diagram at an SNR of 20dB and 10dB*

The authors found the value of R at each SNR and constructed a 4th order polynomial which fit the pattern of R value reduction as the SNR was decreased. Using the obtained R values from Equation 8.1 and the polynomials the system could then estimate the SNR. Different polynomials were required for each modulation scheme.

8.2.1 The SNR Estimation Mechanism of the Proposed System

In Section 4.2 which covered the creation of the optimised DBSCAN algorithm proposed by this work it was stated that the functionality to label core points was not implemented. Furthermore, the proposed DBSCAN feature extractor does not operate on the constellation diagram itself. Due to these two factors, SNR estimation cannot be performed using the same mechanism as in the previously discussed work. However, the ratio R was used as a proxy for determining constellation point density, by finding a different mechanism to use as a measure of the density it is possible to perform blind SNR estimation in a similar manner.

The DBSCAN feature space diagram was used in prior sections to demonstrate how the two 1D DBSCAN operations produce feature clusters which can be used by an MLP for modulation classification purposes. The two 1D DBSCAN algorithms produce two values representing the number of arguments and number of magnitudes of a constellation diagram, these numbers are combined and create feature clusters on the DBSCAN output scatter. Ideally, each modulation scheme was represented by an individual cluster of points on the DBSCAN output scatter chart with clearly defined boundaries and a large amount of separation to nearby clusters, this generally what was produced when DBSCAN was applied to 40dB SNR data. It was explained that deterioration of classification performance occurred when these clusters began to overlap as SNR decreased, this occurred because as the influ-

ence of noise on the constellation diagrams grew, the DBSCAN system became less capable of accurately clustering the argument and magnitude data, finding a lower number of clusters than expected, as noise levels increased the number of clusters found decreased proportionally, causing a drift of clusters towards the origin. This is the mechanism by which DBMC can estimate the SNR of a signal. Therefore, rather than using R as a measure of constellation point density, the obtained numbers of arguments and magnitudes may instead be used.

To illustrate an example, when 40dB data is applied to both 1D DBSCAN algorithms the maximum number of magnitude and argument clusters will be found. As the SNR decreases so too will the number of magnitude and argument clusters, in feature space this can be seen as the feature cluster moving from the 40dB position towards the origin, the location of the cluster is proportional to the SNR of the signal, therefore an MLP classifier can be trained to output the input signal's SNR according to the position of the cluster in feature space. Figure 8.2 illustrates this by showing how the position of the 4QAM cluster moves towards the origin as the SNR is decreased from 40dB SNR to 0dB SNR.

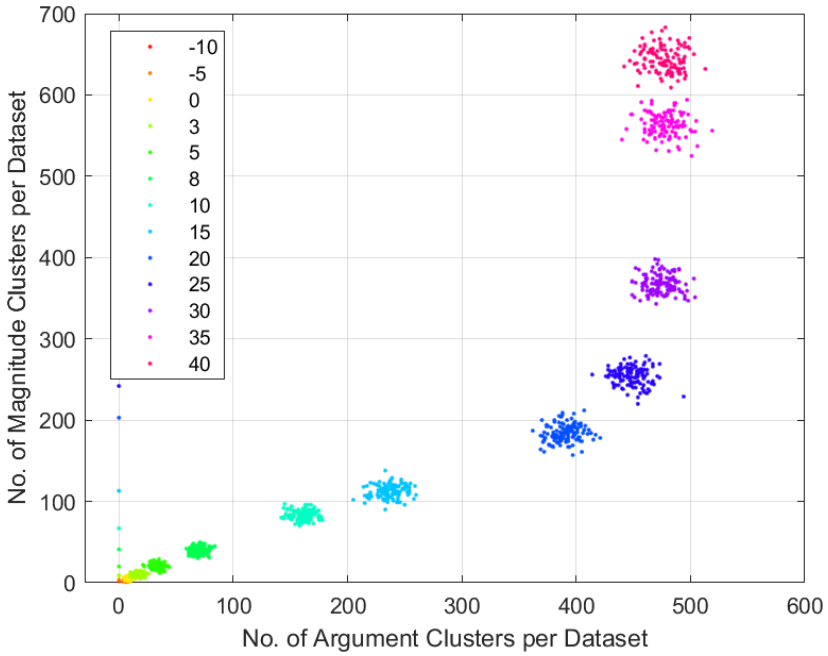


Figure 8.2: *The Resulting Feature Space when DBSCAN is Applied to 4QAM Data of SNRs -10dB to 40dB*

As can be seen in Figure 8.2, the 40dB cluster represents the greatest value of argument and magnitude clusters, every decrease in SNR moves the obtained cluster towards the origin until at 0dB the cluster lies at the minimum value of argument and magnitude clusters. In the case of modulation classification, the feature clusters represented categorical classes, in this case the clusters represent values of a continuous distribution, the SNR of the signal. Furthermore, the relationship between the feature cluster location and the signal SNR also follows a continuous trend. Therefore, rather than training the MLP to classify the feature clusters' location as distinct SNR classes, it can be trained as a regression model to learn the relationship between SNR value and cluster position in feature space. This enables the

system to estimate the SNR of signals which have SNRs which are not represented in the training dataset. For instance, a 4QAM signal of 37.5dB SNR would likely result in a value of the number of argument and magnitude clusters being found which lied in feature space between the 40dB and 35dB cluster. In this case a classifier would provide an SNR estimate of either 35dB or 40dB, guaranteeing a minimum error of 2.5dB. A regression model learns the relationship between SNR and cluster location and would therefore be capable of estimating the SNR as 37.5dB.

Notice how in Figure 8.2 SNR value feature clusters greater than 5dB show strong separation, and SNR values below 5dB are closely spaced towards the origin, much like with classification tasks, a stronger degree of separation results in reduced estimation error, it would therefore be expected that the feature arrangement shown in Figure 8.2 would result in low error at high SNR and a higher error at lower SNR. To compare error the metric MSE in generally used, it is defined in Equation 8.2:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{8.2}$$

Where n is equal to the number of estimation algorithm executions, Y_i is the true SNR, and $(\hat{Y}_i)$is the estimated SNR. With this metric defined, a plot of the obtained MSE over 10 estimation operations is shown in Figure 8.3.

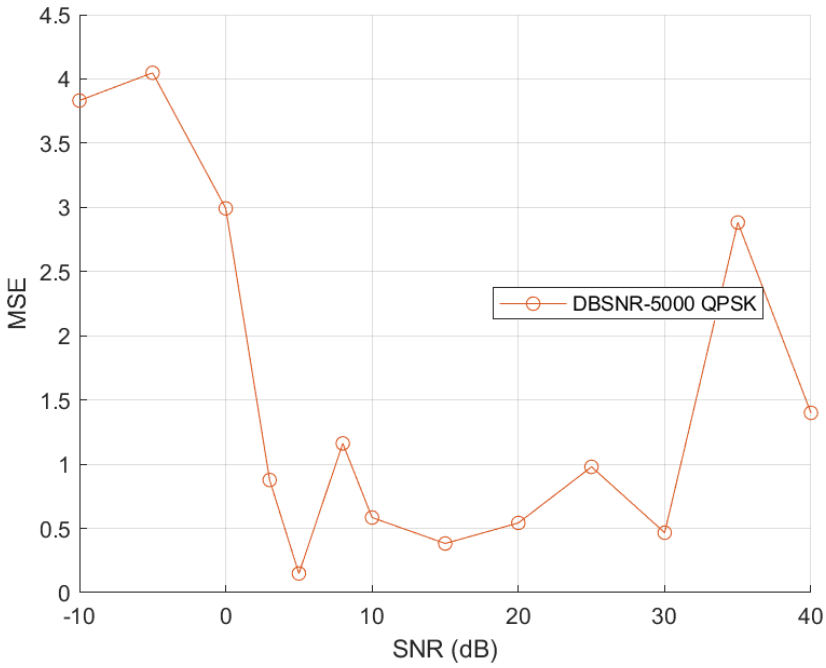


Figure 8.3: *MSE Against SNR (dB) Characteristics for DBSNR-5000 with a 4QAM Input*

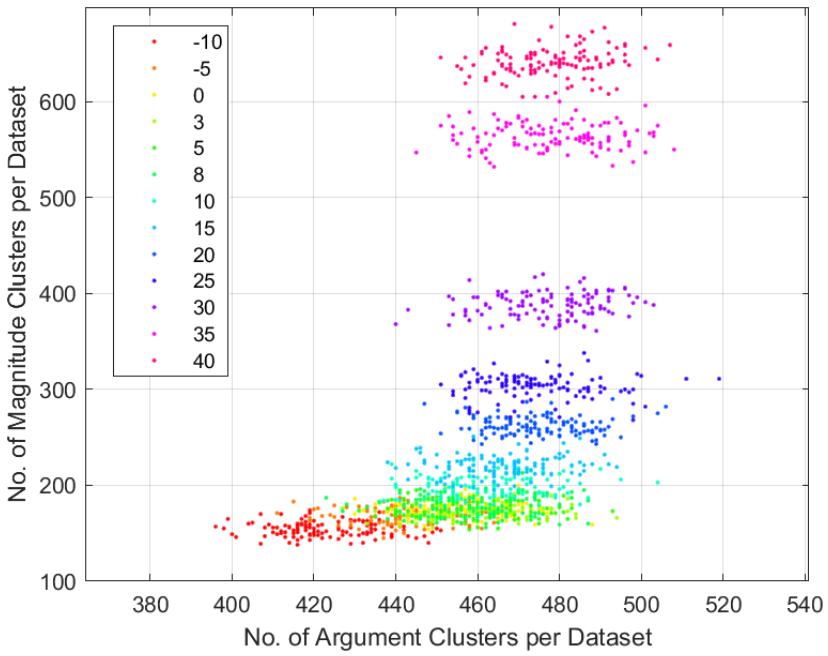


Figure 8.4: *Feature Space Obtained by Applying DBSCAN to 1024QAM at SNRs -10dB to 40dB*

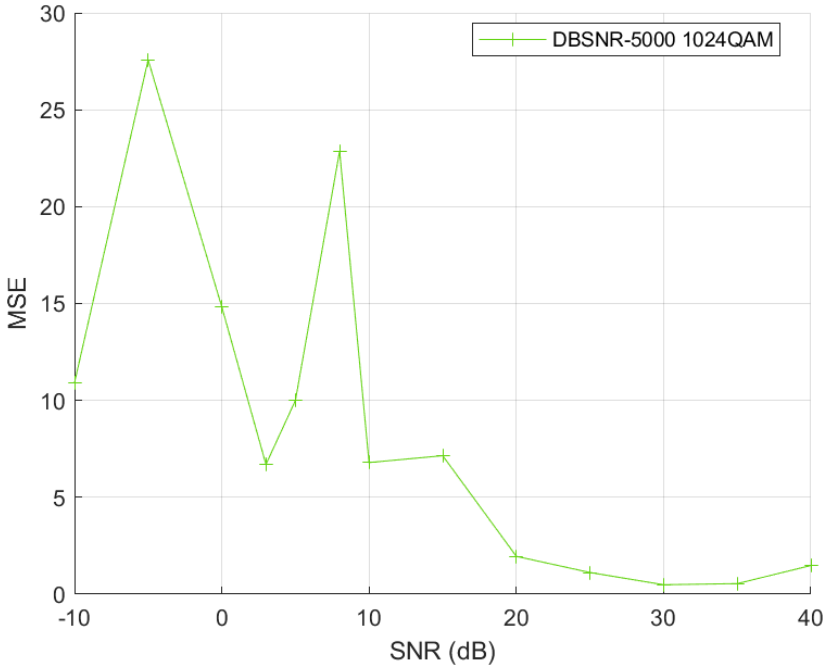


Figure 8.5: *MSE Against SNR (dB) Characteristics for DBSNR-5000 with a 1024QAM Input*

As implied by the feature space diagram in Figure 8.2, Figure 8.3 shows that the reduced feature cluster separation results in a greater error being obtained due to the SNR to feature

cluster relationship being less well defined. A more severe example can be seen when viewing the feature space and MSE against SNR curve of 1024QAM, these may be seen in Figures 8.4 and 8.5 respectively.

In Figure 8.4 there is strong separation between the high SNR clusters, implying a strong estimation accuracy would be achieved. Conversely, for clusters representing SNRs less than 10dB there is near total cluster overlap, there is no clearly defined relationship between SNR and feature cluster location for the MLP to learn in this case, a high degree of error is therefore guaranteed. The -10dB cluster is somewhat separated from the highly overlapping clusters from -5dB to 10dB SNR but still exhibits a small degree of overlap. It would be expected that the error of the system at -10dB would be smaller in comparison but not to the same degree as high SNR data. Figure 8.5 confirms these predictions, SNRs between 10dB and -5dB exhibit a high degree of error whereas at higher SNRs the error remains consistently low.

To summarise the SNR estimation mechanism, the DBSCAN 1D feature extraction mechanism finds a variable number of magnitude and argument clusters which is proportional to the signal SNR. The obtained number of clusters is used as a proxy for determining the density of the constellation diagram. A regression model learns the relationship between the extracted number of argument and magnitude clusters and the SNR to estimate future inputs. Strong separation between SNR clusters is a requirement for the regression model to learn the relationship accurately.

8.3 SNR Estimation Results

Now that the operation of the DBSCAN SNR estimator has been explained the overall results can be presented. In the DBMC results section it was shown that implementations with larger dataset sizes in general provide a higher level of accuracy, up to a limit of approximately a dataset size of 5000. It was also found that the proposed systems in general exhibited stronger performance on low order as well as PSK and APSK signals as opposed to QAM. This section will evaluate the performance of the proposed SNR estimation system and look to identify similar trends.

As with the modulation classifier, the proposed hardware implementation structure did not allow for synthesis of the size 5000 model, the results are provided purely to illustrate the upper bounds of performance of the proposed mechanism. Results will be provided in terms of both MSE and the average estimated SNR against the true SNR. In both cases the number of tests performed to obtain the provided values was 10. Utilised ε values were obtained using the RMS method detailed in Chapter 6, the values used for modulation classification and SNR estimation are consistent across dataset sizes for both operations, the beginning of each dataset size results section will restate the utilised values. minPts values were always set to 2. The dataset itself is the same as was employed for modulation classification, details may be found in Section 7.2.

8.3.1 DBSNR-5000 Accuracy

This is the DBSNR configuration with the largest dataset size which was tested. As has been the case throughout all dataset size 5000 tests in this thesis, these results were obtained

via software simulation as the implementation was too large to synthesise. The argument and magnitude ε values employed were 0.1 and 0.0007 respectively. Figures 8.6 and 8.7 respectively show the obtained MSE against SNR curves as we all the Estimated SNR against True SNR for each QAM modulation scheme.

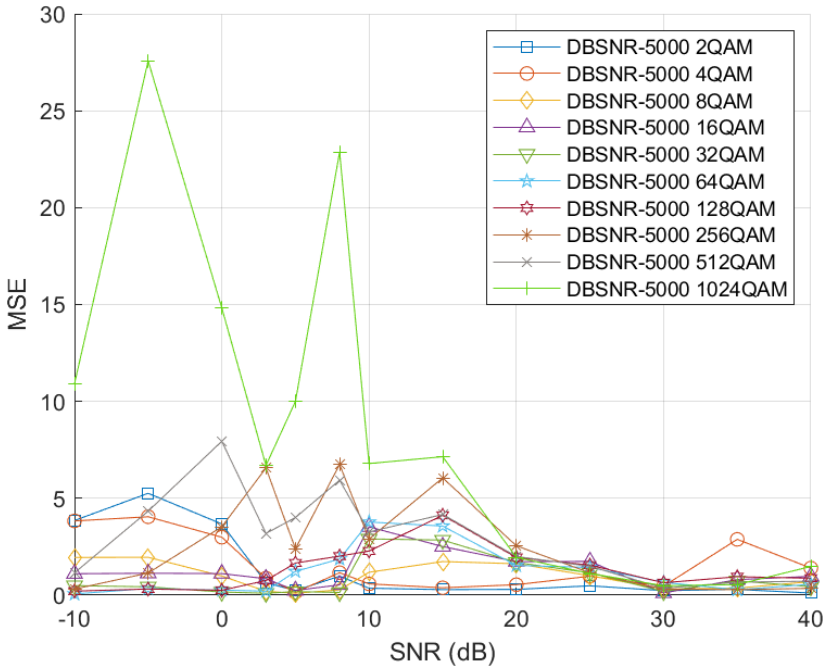


Figure 8.6: *MSE Against SNR (dB) Characteristics for DBSNR-5000 on QAM Signals of Orders 2 to 1024*

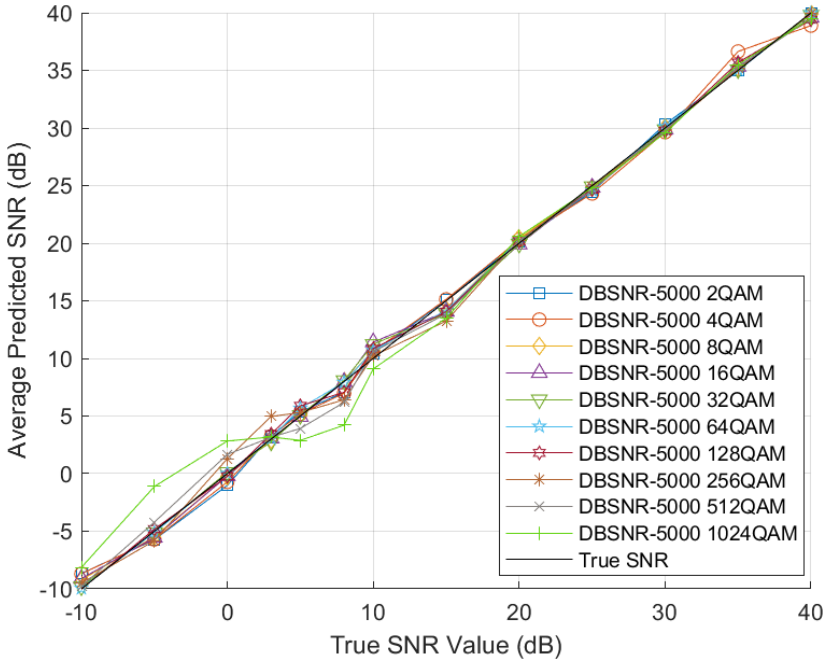


Figure 8.7: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-5000 on QAM Signals of Orders 2 to 1024*

Figure 8.6 displays the MSE against SNR achieved by the system on QAM signals of orders 2 to 1024 in the SNR range -10dB to 40dB. In general, an MSE below 5 is achieved across the entire SNR range, the three highest order modulation schemes are the exception. Below 20dB 256QAM, 512QAM, and 1024QAM each display spikes in MSE which implies a reduced SNR estimation accuracy in this SNR range. The lower order modulation schemes also exhibit a somewhat larger MSE below 20dB in comparison to their MSE at higher SNRs but the MSE remains below a value of 5 in all cases. Above 20dB all MSE values are low, below a value of 3 in all cases. Across the full SNR range there is a trend of higher order signals resulting in a higher MSE being observed.

Figure 8.7 displays the Average Predicted SNR against the True SNR Value. As suggest by Figure 8.6, above 20dB SNR there is strong correlation between the line denoting the True SNR and the Predicted SNR values, showing that highly accurate estimation is achieved in this SNR region. Below 20dB there is more deviation from ideal performance, particularly when inspecting the lines representing the 3 highest order signals, the 1024QAM estimation accuracy is particularly poor. However, despite this deviation the estimated SNR is never more than 4dB away from the true value, the worst result obtained is when 1024QAM is estimated to have an SNR of -1.12dB at -5dB. These results show that despite a degree of inaccuracy, a CR system employing this algorithm for SNR estimation purposes could rely on an estimate accurate to within 4dB across all QAM signals and at all SNRs, the estimated SNR would in general be within a single dB.

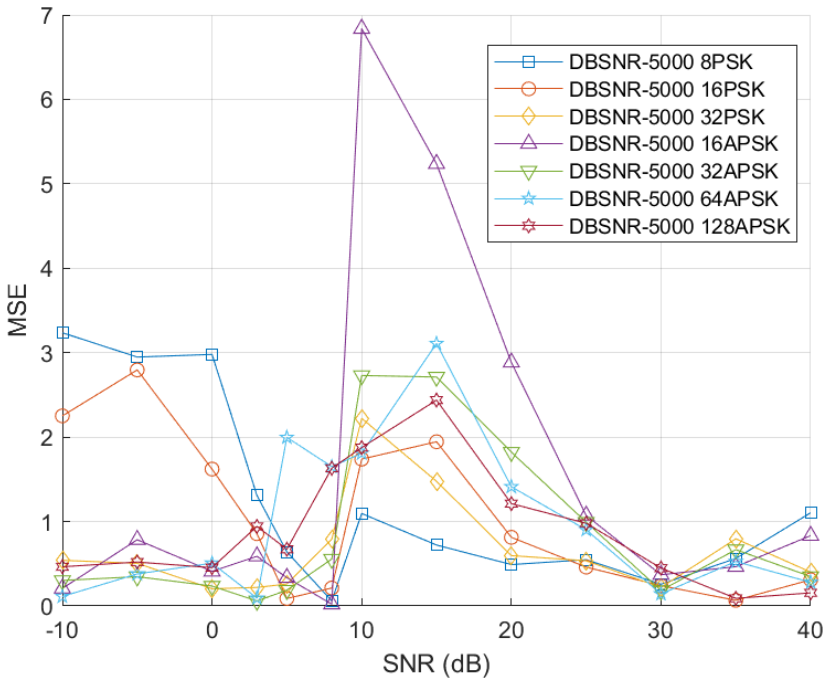


Figure 8.8: *MSE Against SNR (dB) Characteristics for DBSNR-5000 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

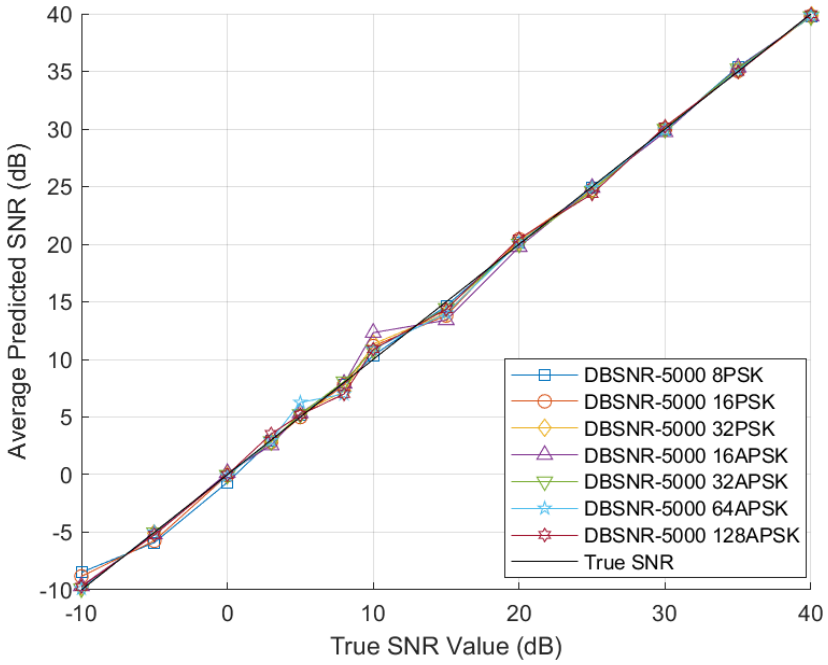


Figure 8.9: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-5000 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

Figure 8.8 displays the obtained MSE against SNR for various PSK and APSK signals

from -10dB to 40dB SNR. In this case the majority of the obtained MSE values are below 3, with the exceptions to this being 16APSK at 10dB and 15dB and 8PSK at 0dB to -10dB. These MSE results suggest that the system is a more capable SNR estimator on PSK and APSK modulation schemes than QAM.

Figure 8.9 confirms that the system is highly effective at estimating the SNR of these signal formats as in the majority of cases the estimated SNR is nearly identical to the true SNR. At an SNR value of 10dB and 15dB there is some slight deviation from the ideal result, but the maximum deviation is never more than 2.32dB which occurs at 10dB for 16APSK. The results shown here that a highly accurate SNR estimation result can be obtained by the DBSNR-5000 system on the employed PSK and APSK signals across the full SNR range.

8.3.2 DBSNR-1000 Accuracy

This system is the largest configuration which was implemented and tested on an FPGA. The testing setup was as described in Section 7.3. An argument and magnitude ε value of 0 and 0 was employed.

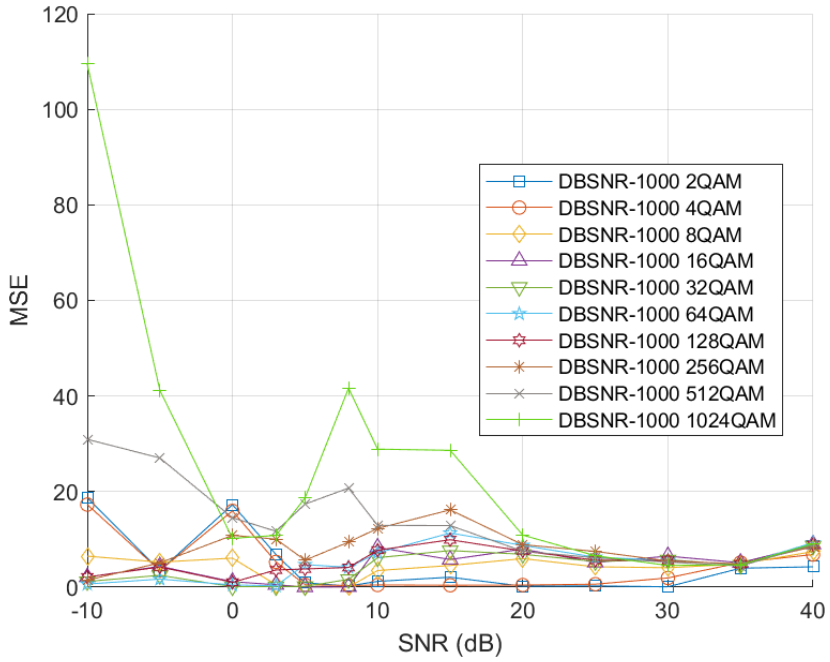


Figure 8.10: *MSE Against SNR (dB) Characteristics for DBSNR-1000 on QAM Signals of Orders 2 to 1024*

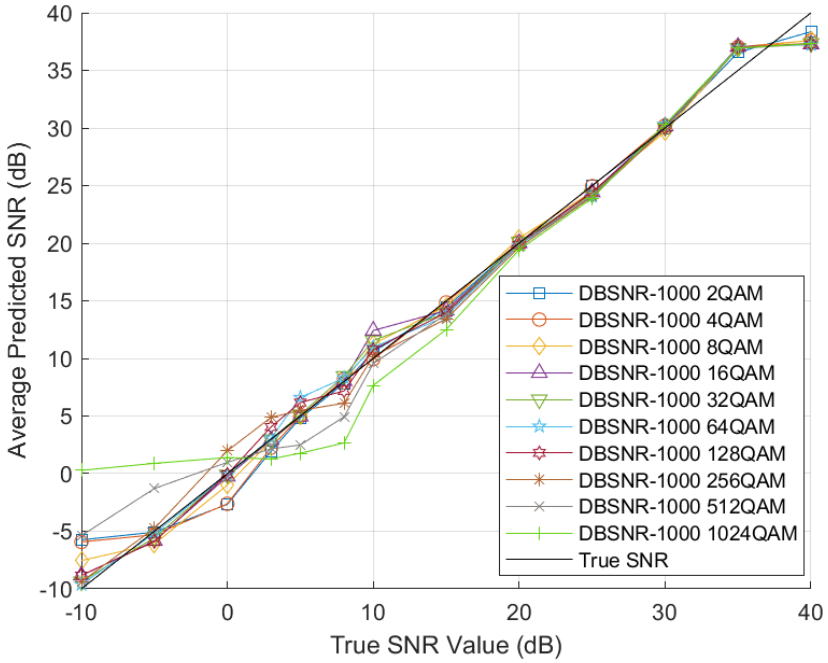


Figure 8.11: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-1000 on QAM Signals of Orders 2 to 1024*

Figure 8.10 displays the obtained MSE against SNR for QAM signals of order 2 to 1024 in the SNR range -10dB to 40dB. The MSE values follow a similar trend to what was observed for DBSNR-5000 with higher accuracy being obtained above an SNR of 20dB and comparatively lower accuracy below this SNR value. Despite the similar trend, the obtained MSE values are in general larger than seen with the larger dataset, with the values generally being between 5dB and 10dB in this region. The high MSE of the 3 highest order signals below an SNR of 20dB is once again observed, with 1024QAM showing a spike in MSE to 41.20 at 8dB SNR and a larger spike to 109.5 at -10dB. Disregarding the sub 20dB inaccuracies exhibited by the 3 highest order modulation schemes, the MSE is consistent for the majority of modulation schemes, implying consistent SNR estimation accuracy across the investigated SNR range.

Figure 8.11 displays the average predicted SNR against the true SNR for each modulation scheme. The first notable difference to the results seen for DBSNR-5000 is that at the SNRs 35dB and 40dB there is a significant deviation from the line denoting ideal performance. All modulation schemes deviate from the ideal line by a maximum of 1.92dB at 35dB and 2.65dB at 40dB. Between 30dB and 20dB SNR there is a region of accurate estimation. Below 20dB SNR the deviation becomes more pronounced, particularly with 512QAM and 1024QAM. Despite the significant deviation from the ideal by these two signals, in all other cases the estimated SNR remains accurate to within 3dB, other than at -10dB. At -10dB the two lowest order modulation schemes exhibit nearly 5dB of estimation inaccuracy, the largest inaccuracy observed on any signal outside of the two highest orders in the tested SNR range.

DBSNR-1000 has thus been shown to provide inferior performance to when a dataset size of 5000 was employed for estimation, yet despite the reduction in accuracy in all cases the average estimated SNR remains below 3dB when discounting the performance of the system on the two highest order signals. In the majority of cases the estimation accuracy is accurate

to within 1dB. Due to the high error achieved by the system on 512QAM and 1024QAM it is not recommended for usage on these signal formats, but on all QAM signals of an order 256 and below the estimation accuracy is low and consistent enough to provide a measure of signal SNR which may allow for informed decision making in a CR scenario.

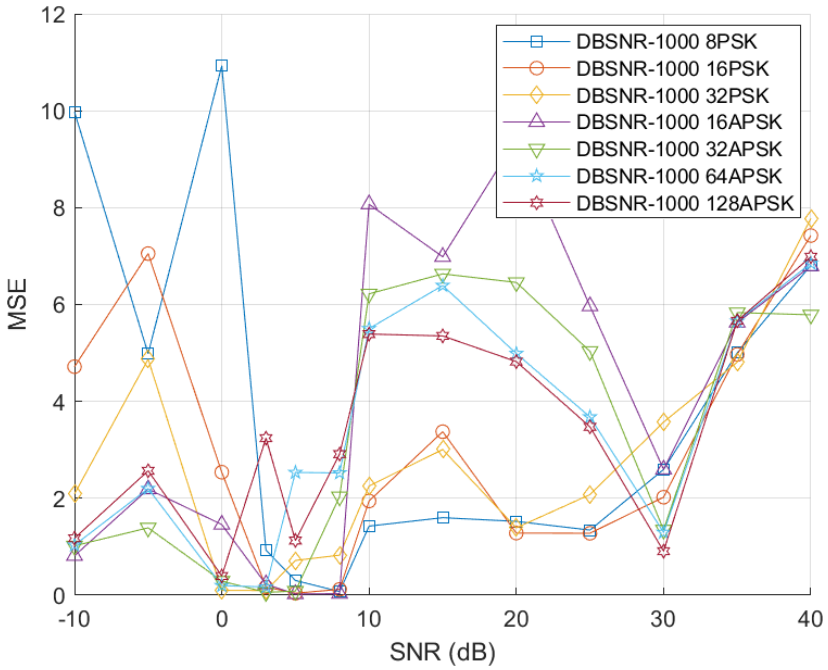


Figure 8.12: *MSE Against SNR (dB) Characteristics for DBSNR-1000 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

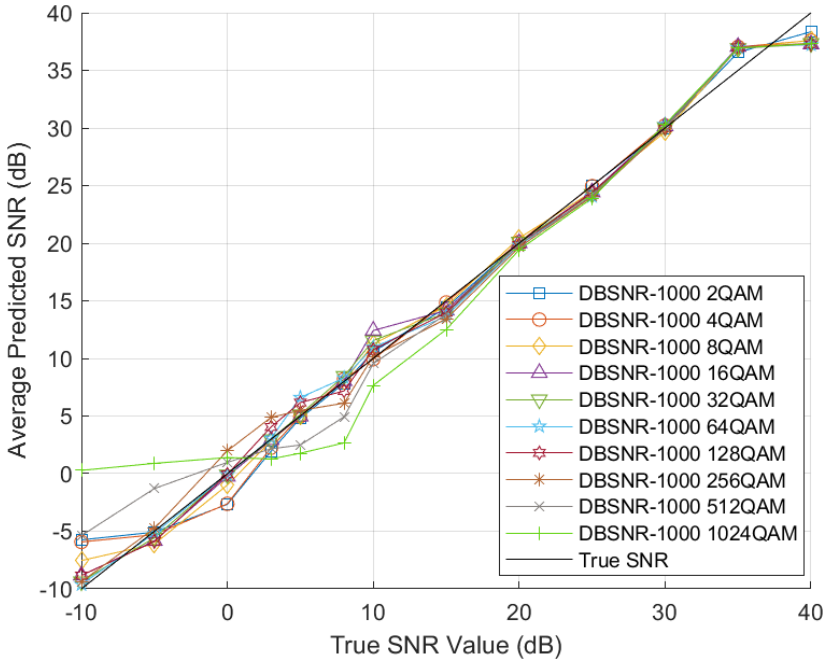


Figure 8.13: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-1000 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

Figure 8.12 displays the achieved MSE against SNR for DBSNR-1000 on various PSK and APSK modulation schemes in the SNR range -10dB to 40dB. Similarly to what was observed with DBSNR-5000, the system in general achieves lower MSE values on these signals than with QAM signals. The maximum obtained MSE was 10.9 which was achieved on 8PSK at 0dB SNR. Discounting the high MSE obtained on all signals at 35dB and 40dB SNR, the system seems to be superior at estimating the SNR of PSK signals in the 10dB to 30dB SNR region, and APSK signals in the -10dB to 10dB SNR region. The obtained MSE is therefore inconsistent but remains relatively low in all cases, suggesting that strong estimation accuracy is obtained across the majority of SNRs.

Figure 8.13 displays the DBMC-1000 average predicted SNR against the true SNR for each modulation scheme in the SNR range -10dB to 40dB. Similarly to what was observed with QAM signals there is a large error on all signals at 35dB and 40dB, at both SNRs a predicted SNR of approximately 37.4dB is obtained, suggesting that there is significant overlap between all feature clusters representing these SNRs. Other than this high SNR inaccuracy the estimated SNR tracks the true SNR with strong precision. The largest inaccuracy in the -5dB to 30dB range comes from 16APSK at 10dB where an average estimated SNR of 12.30dB is observed, an error of 2.30dB. The PSK signals display a large error of a maximum of 3.18dB at -10dB for 8PSK, but otherwise the estimated SNR of the APSK signals remains below 1dB at this SNR value.

The results shown in Figures 8.12 and 8.13 demonstrate that between -10dB and 30dB SNR a strong estimate of the SNR may be achieved by the DBSNR-1000 system, a maximum error of 2.3dB was obtained in a single case showing that this system can be used for SNR estimation purposes in a CR system to provide accurate information to enable informed decision making.

8.3.3 DBSNR-500 Accuracy

DBSNR-500 is the second largest of the proposed implementable SNR estimation systems. The results presented here were obtained with an argument and magnitude ε value of 0.5 and 0 respectively. The maximum order of modulation scheme tested was set to 128 which determined by Equation 6.1 in Section 6.2.1.

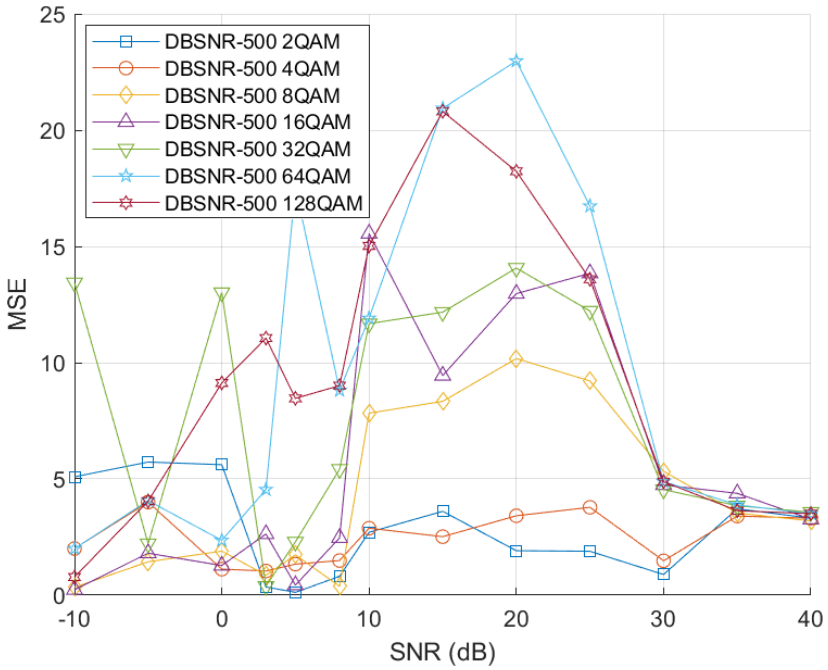


Figure 8.14: *MSE Against SNR (dB) Characteristics for DBSNR-500 on QAM Signals of Orders 2 to 128*

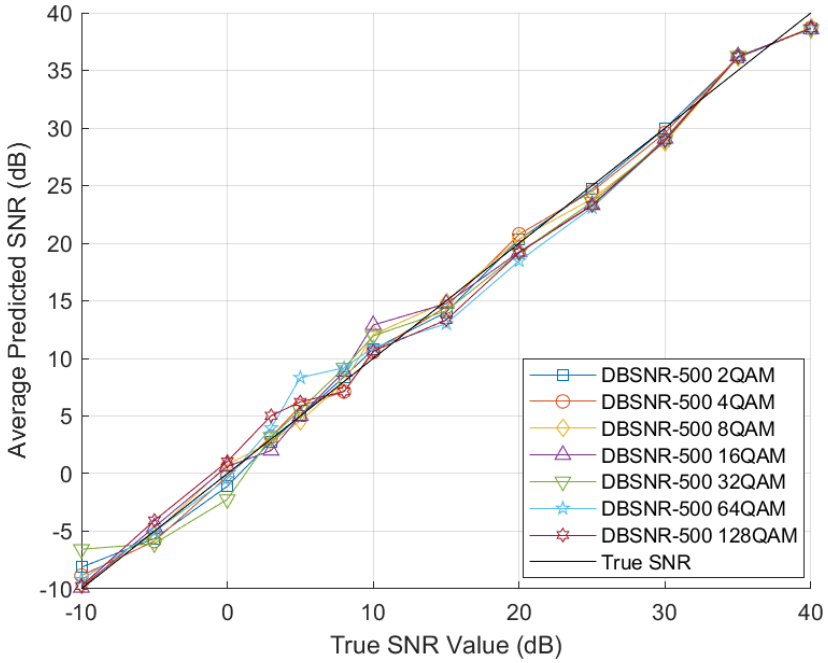


Figure 8.15: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-500 on QAM Signals of Orders 2 to 128*

Figure 8.14 displays the MSE against SNR values achieved by DBSNR-500 on QAM signals of orders 2 to 128 at SNR values -10dB to 40dB. Continuing the trend observed between DBSNR-5000 and DBSNR-1000, the obtained MSE values are consistently higher with the reduced dataset size. While 2QAM and 4QAM remain consistently below an MSE value of 5, the MSE values obtained for the remaining set of modulation schemes is primarily larger than 5 and often greater than 10. Despite the larger MSE values there are no cases of the MSE values being larger than 25, suggesting that across the full SNR range reasonably consistent estimation accuracy will be achieved. The trend of higher order modulation scheme resulting in larger MSE values is again exhibited.

Figure 8.15 displays the average predicted SNR against the true SNR on the same set of QAM signals in the SNR range -10dB to 40dB. At an SNR of 35dB and 40dB a similar inaccuracy as was observed with DBSNR-1000 can be seen, although to a lesser extent. At these SNRs a maximum error of 1.23dB and 1.39dB is achieved. As suggested by the discrepancy in the MSE values between Figure 8.10 and Figure 8.14, the average estimated SNR deviates from the true SNR to the largest extent thus far seen. This phenomenon is made most clear between 20dB and 30dB which is a region which the DBSNR systems generally estimate the SNR to a high degree of precision, in this case the estimated SNR of the majority of modulation schemes is consistently below the expected value. Despite this inaccuracy the maximum error within this range is still only 2dB. Below 20dB SNR there is a region of more significant and less consistent error, although despite this error the maximum estimation inaccuracy still remains within 3.3dB.

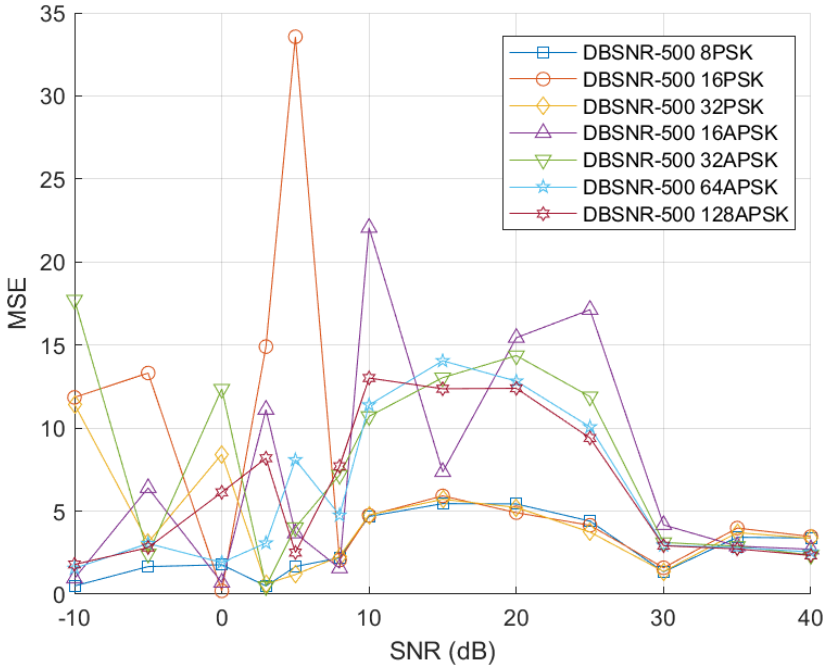


Figure 8.16: *MSE Against SNR (dB) Characteristics for DBSNR-500 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

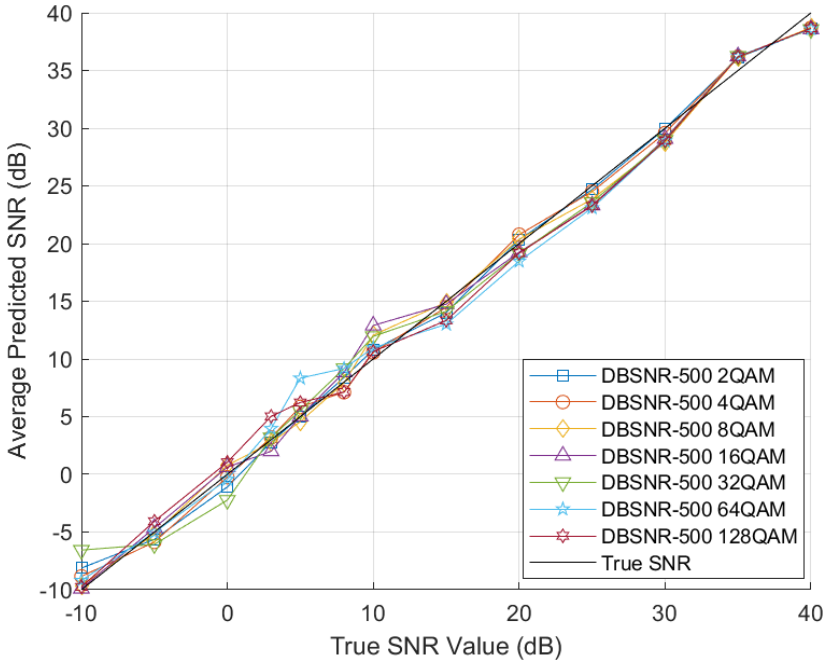


Figure 8.17: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-500 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

Figure 8.16 displays the MSE against SNR values obtained by DBSNR-500 on PSK signals

of orders 8 to 32 and APSK signals of orders 16 to 128. As has been observed for both DBSNR-5000 and DBSNR-1000, the achieved MSE on PSK signals remains consistently low above 10dB SNR before spiking below this SNR. The inverse is true for the MSE of the APSK signals. Once more the average obtained MSE increases as the dataset size is reduced, the scale of the values in this case remains consistent with what was achieved by the system on QAM signals if the 5dB value for 16PSK is discounted. The displayed MSE values suggest a similar amount of error to what was observed in figure 8.15, with consistent misestimation of approximately 1-2dB but no instances of significantly large errors, other than perhaps at 5dB for 16PSK.

Figure 8.17 displays the average predicted SNR against the true SNR on the same set of PSK and APSK signals in the SNR range -10dB to 40dB. This is perhaps the first instance of the performance of the system on PSK and APSK signals being approximately equivalent to what was obtained with QAM signals. The error of the system when estimating the SNR of all signals is above and below the true SNR at 40dB and 35dB respectively. A similar trend of consistently estimating the SNR to be below the true value is observed between 30dB and 20dB SNR, although a maximum error of only 1.96dB is achieved. Below 20dB there is a region of inconsistent error, the largest errors thus far seen in any comparisons are obtained with 16APSK exhibiting an error of 4.0dB at 10dB SNR and 16PSK having its SNR estimated with an error of 3.2dB at 3dB.

With the large errors, employing a system of this dataset size is perhaps approaching the limit of the minimum system size which can be appropriately utilised as part of a CR enabled system. While an error of 4dB is indeed significant, the estimation accuracy is in general less severe, although the inconsistency particularly below 15dB SNR does not give confidence that this system would provide information accurate enough to support dynamic modulation scheme adjustments.

8.3.4 DBSNR-250 Accuracy

The DBSNR-250 system was the smallest system for which SNR estimation performance will be provided. The provided results were obtained with an argument and magnitude ε value of 1.5 and 0.00390625 respectively.

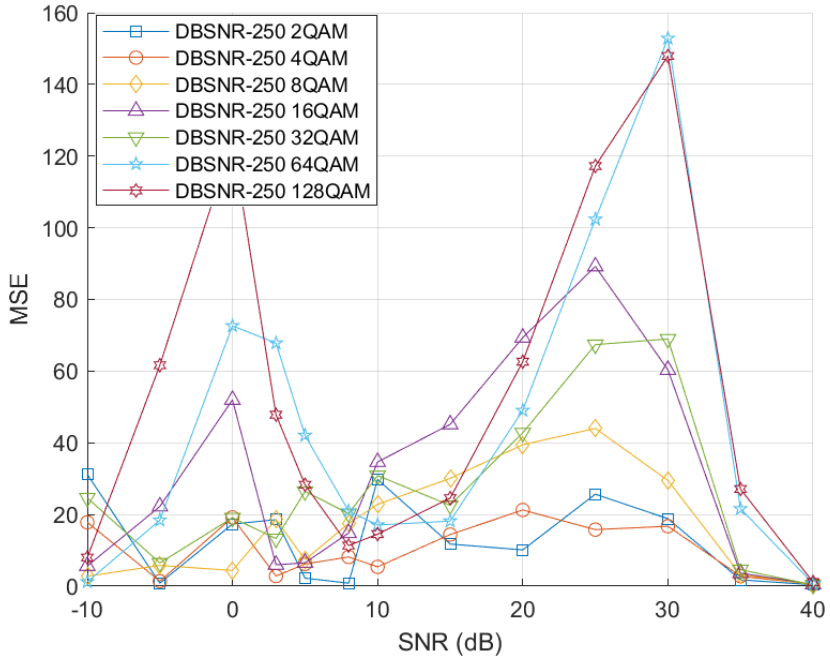


Figure 8.18: *MSE Against SNR (dB) Characteristics for DBSNR-250 on QAM Signals of Orders 2 to 128*

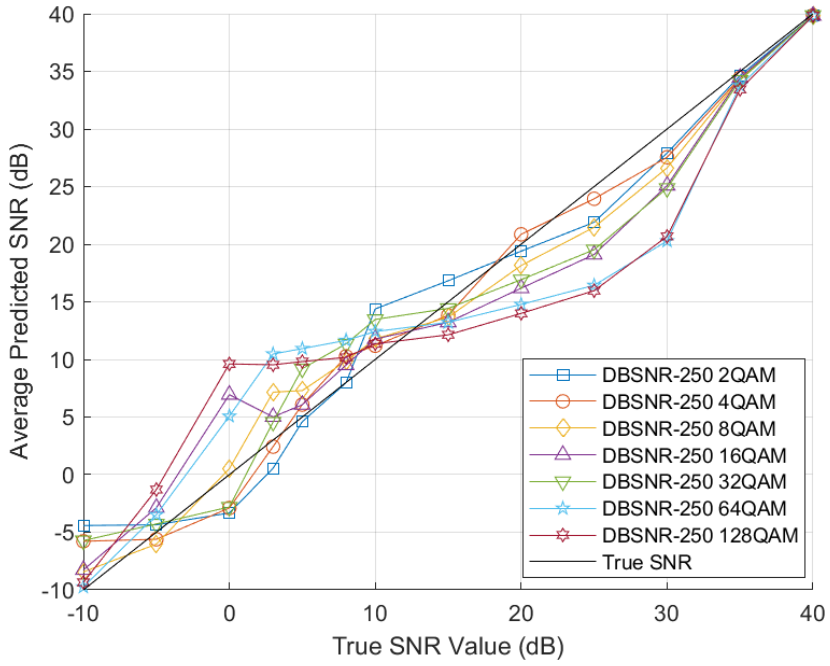


Figure 8.19: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-250 on QAM Signals of Orders 2 to 128*

Figure 8.18 displays the MSE against SNR values achieved by DBSNR-250 on QAM signals of orders 2 to 128 at SNR values -10dB to 40dB. The trend of increasing average MSE as the

dataset size is reduced once again continues but, in this case, the MSE value rise to such a degree that the MSE is greater than 20 in the majority of cases. Furthermore, in many cases the MSE rises to values greater than 60 and in a few cases to over 100. These values suggest that the SNR estimation performance of this size system will be poor.

Figure 8.19 displays the average predicted SNR against the true SNR on QAM signals of orders 2 to 128 in the SNR range -10dB to 40dB. The only region of strong accuracy is at 35dB and 40dB SNR, contrary to what has been observed with previous DBSNR configurations. The estimated SNR generally strays far from the true SNR, in many cases the estimation error is as large as 10dB, this degree of inaccuracy is particularly common with the higher order signals.

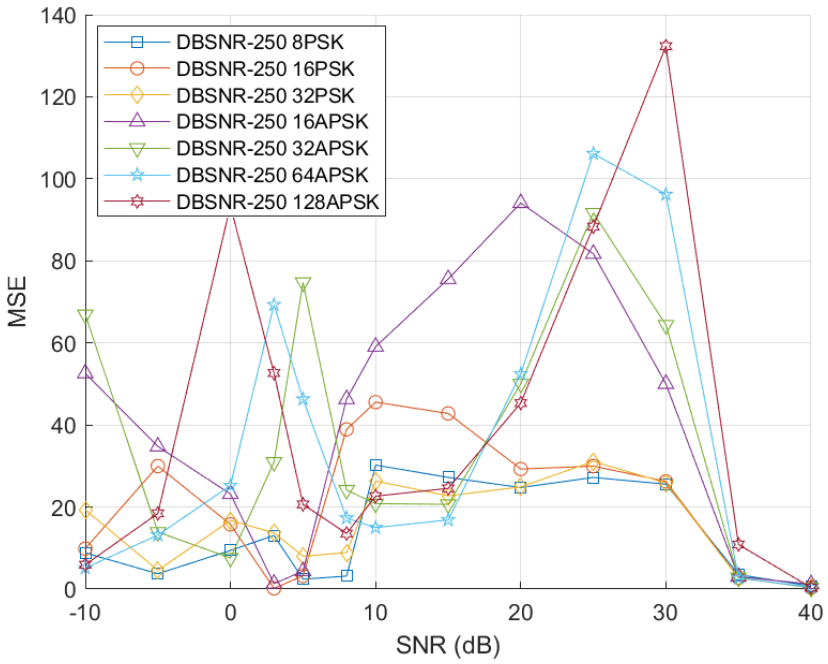


Figure 8.20: *MSE Against SNR (dB) Characteristics for DBSNR-250 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

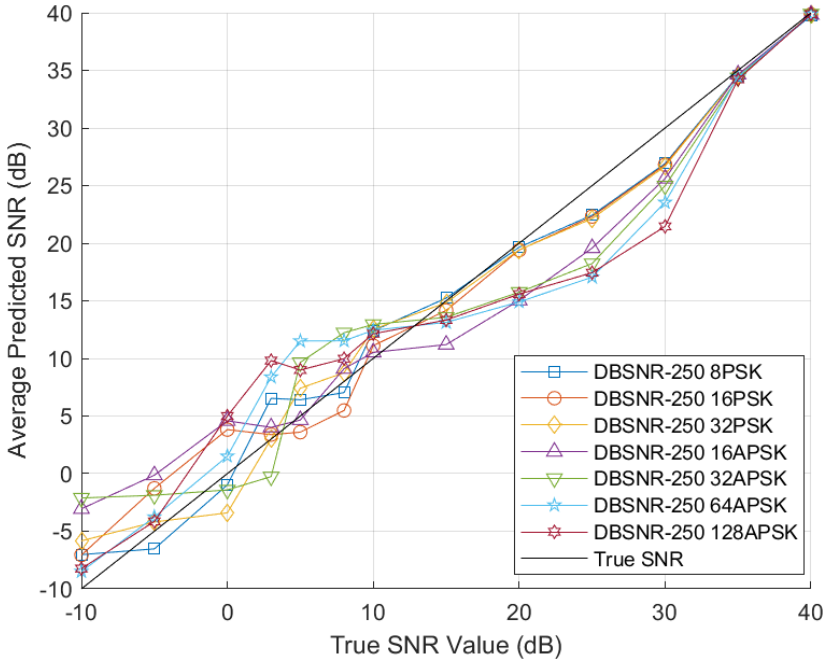


Figure 8.21: *Average Estimated SNR (dB) Against True SNR (dB) for DBSNR-250 on PSK Signals of Orders 8 to 32 and APSK Signals of Orders 16 to 128*

Figure 8.20 displays the MSE against SNR values obtained by DBSNR-250 on PSK signals of orders 8 to 32 and APSK signals of orders 16 to 128. Similarly to the obtained MSE against SNR on QAM signals the obtained MSE values are generally greater than 20 and in many cases spike to values greater than 80. The MSE values are inconsistent and do not seem to follow any particular trend.

Figure 8.21 displays the average predicted SNR against the true SNR on the same set of PSK and APSK signals in the SNR range -10dB to 40dB. Similar to what was observed on QAM signals with this system configuration, the estimated SNR largely fails to track the true SNR and errors greater than 5dB are commonly observed.

Figure 8.19 and figure 8.21 clearly show that this system configuration commonly results in an estimation error which is too large to be relied upon to make informed decisions. While the system can provide an approximate estimation of the SNR, errors which are frequently between 5dB and 10dB will assuredly lead to unreliable information if deployed in a CR system and thus frequent redundant requests for a modulation scheme adaptation. Due to the poor results exhibited by this system size and the general trend of smaller dataset sizes resulting in poor estimation performance, it is unnecessary to discuss the performance of DBSNR-50.

8.3.5 DBSNR Results Comparison and Discussion

This chapter has provided the MSE against SNR statistics as well as the estimated SNR against true SNR performance for 4 different DBSNR configurations. Within the provided data several trends have been identified.

The most important trend which has been observed is that estimation accuracy increases

as the dataset size increases. While DBSNR-5000 and DBSNR-1000 exhibited an error as large as 4dB in some cases on the 512QAM and 1024QAM modulation schemes, the maximum error achieved on the rest of the QAM signals in the dataset between SNR values of -5dB and 30dB was found to be consistently lower than 1dB for DBSNR-5000 and 2dB for DBSNR-1000, with the majority of examples exhibiting much lower error. DBSNR-500 was found to approach the limit of acceptable performance with consistent errors greater than 2dB a maximum error of 4dB being observed. DBSNR-250 was demonstrated to be an unreliable estimator with consistent errors greater than 5dB being obtained.

Similarly to what was observed in the modulation classification results discussed in Chapter 6, the two largest configurations exhibited a stronger performance on PSK and APSK signals than was achieved on QAM signals. The performance differential between the two datasets for DBSNR-500 and DBSNR-250 was found to be minimal.

Among modulation schemes of the same format (QAM, PSK, APSK), it has been found that in general the system achieves a lower SNR estimation error on signals of a lower order. While this is not always the case, it has been observed that low order signals are less likely to exhibit high spikes in MSE. Said spikes in MSE, as will be seen in the following section, are atypical for SNR estimation systems where smooth trends are generally observed. The spikes are due to feature cluster overlap, in these cases the MLP is not provided with enough information to make an informed decision about the signal SNR, a more detailed discussion about this phenomenon can be found in Section 8.2.1.

While the various DBSNR configurations have been compared among each other and it has been found that the systems featuring the larger dataset result in lower error, to truly evaluate the effectiveness of the proposed method comparisons must be drawn with the state-of-the-art methods from the literature. The next section provides this analysis.

8.4 NDA SNR Estimation Results Comparison

This section provides comparisons and analysis of the DBSNR estimation performance with the works discussed in the literature review in Chapter 2. Focus will be placed upon comparisons with methods which have been demonstrated to be effective on a wide range of modulation schemes as this is the principal requirement of an SNR estimation mechanism which forms part of a CR. Despite this, to obtain a perspective of the overall effectiveness of the proposed system, comparisons will also be drawn to NDA estimation techniques which have been found to perform well on a limited set of modulation schemes.

8.4.1 Low-Order NDA SNR Estimation Results Comparisons

The first of the provided comparisons will be drawn with the NDA SNR estimation systems which provide data only when QPSK data is employed as the input. This includes each of the DL-based estimators as well as the M_2M_4 estimator results taken from comparison provided by Zheng et al. [24].

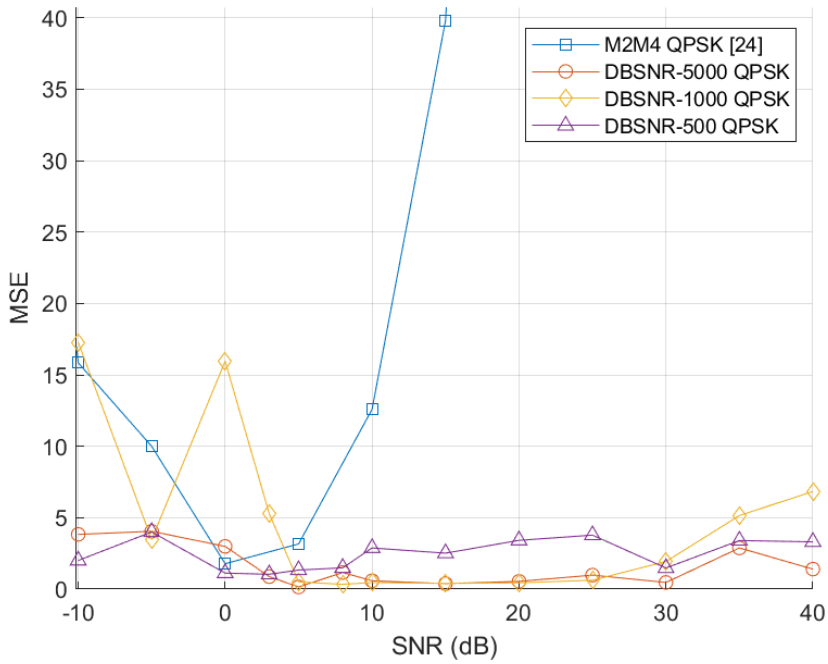


Figure 8.22: *MSE Against SNR (dB) of the M_2M_4 Estimator and DBSNR-5000, 1000, and 500 on QPSK Data. M_2M_4 Datapoints Beyond 15dB Lie Outside of Y-axis scale. These Datapoints were Omitted to Better Display DBSNR Accuracy Trends.*

Figure 8.22 displays the MSE against SNR curves for DBSNR configurations 5000, 1000, and 500, alongside the M_2M_4 performance on QPSK data taken from [24]. There are only 4 examples where a DBSNR system exhibits a higher MSE than M_2M_4, these occur at -10dB, 0dB, and 3dB SNR for DBSNR-1000 as well as 0dB for DBSNR-5000. Otherwise, each of the DBSNR systems achieve a lower MSE than the algorithmic estimator. Crucially, DBSNR in general maintains an MSE consistently below 5 above an SNR of 10dB, demonstrating strong performance in the SNR range at which algorithmic estimators have been found to exhibit asymptotic behaviour.

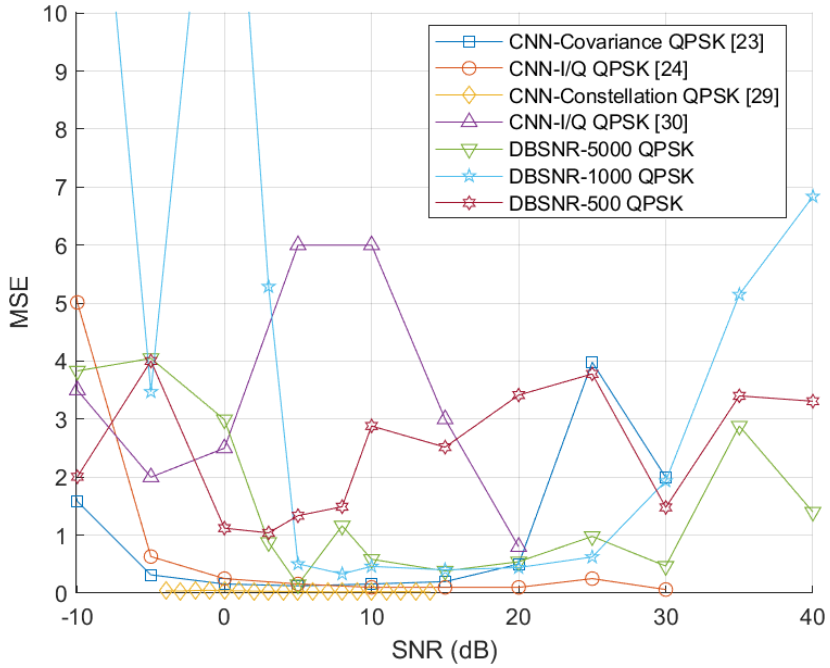


Figure 8.23: *MSE Against SNR (dB) of the CNN Estimators and DBSNR-5000, 1000, and 500 on QPSK Data. Some CNN-Covariance [23] Datapoints Lie Outside of Y-axis Scale, Datapoints Were Omitted to Better Display the Performance Trends of DBSNR and other CNNs.*

Figure 8.23 shows the MSE against SNR curves for the 3 largest DBSNR configurations and the 4 CNN-based SNR estimators from the literature, QPSK was used as the input data in all cases. Comparisons with the raw I/Q accepting CNN by Yang et al. [30] show that all 3 DBSNR configurations are competitive in terms of MSE, DBSNR-5000 achieves a lower MSE at all SNRs greater than 0dB and DBSNR-1000 at SNRs greater than 3dB, DBSNR-500 achieves the same feat in the SNR range 0dB to 15dB. The constellation diagram accepting CNN [29] only provides MSE statistics in the range -5dB to 15dB, within this range both DBSNR-5000 and DBSNR-1000 exhibit a slightly greater MSE, but the difference is small. DBSNR-500 consistently achieves an MSE greater than 1 in this range, which is a more significant performance differential, but consistent accuracy is still demonstrated. Comparisons with the I/Q [24] and covariance matrix [23] accepting CNNs are similar, the 2 largest DBSNR systems achieve comparable MSE values above an SNR of 5dB, the obtained low MSE values are even maintained to 30dB SNR whereas [23] sees an MSE increase at 25dB. The largest disparities in performance can be seen below 5dB SNR for DBSNR-1000, here the MSE values peak as high as 17.3 and 16, however in all other cases the trends in performance and achieved MSE values are similar.

Figure 8.23 has shown that each DBSNR configuration matches or outperforms the accuracy of M_2M_4 algorithm in its optimal SNR range and they do not suffer from the asymptotic behaviour which characterises the M_2M_4 MSE against SNR performance. There are two exceptions to these findings which are the MSE achieved by DBSNR-5000 and DBSNR-1000 at 0dB SNR where a greater MSE is exhibited. Strangely, the weakest of the proposed systems DBSNR-500 performs the strongest at this SNR, these results are likely to be due to

hyperparameter tuning rather than indicitive of an inability of DBSNR-5000 and DBSNR-1000 to perform SNR estimation at this SNR. Due to these these findings it cannot be said that DBSNR systems are wholly superior estimators on QPSK signals in terms of MSE, but they do in general exhibit a lower MSE within the optimum SNR range of M_2M_4 and crucially do not display asymptotic behaviour. Comparisons in Figure 8.23 have shown that the minimum obtained MSE by the 2 largest DBSNR configurations was slightly inferior to that of the strongest CNN SNR estimators, all 3 DBSNR systems generally surpassed the performance in comparison to the weakest CNN [30]. Thus, it can be concluded that DBSNR technology can achieve near equivalence to the accuracy obtained by CNN estimators at all SNRs greater than 5dB regardless of the CNN input, and is also demonstrated across a wider set of modulation schemes.

8.4.2 Medium-Order NDA SNR Estimation Results Comparison

The next comparisons to be drawn are to the methods which provide NDA SNR estimation results when medium-order modulation schemes are employed as the input dataset. The modulation schemes utilised in this comparison are QAM signals of orders 16, 32, and 64, as well as APSK orders 16 and 32. The works included in this comparison are the moment-based estimator by Álvarez-Díaz et al. [28], the I/Q accepting CNN by Yang et al. [30], the EDF-based estimator by Wang et al. [86], and the envelope estimator proposed by Zuo et al. [26].

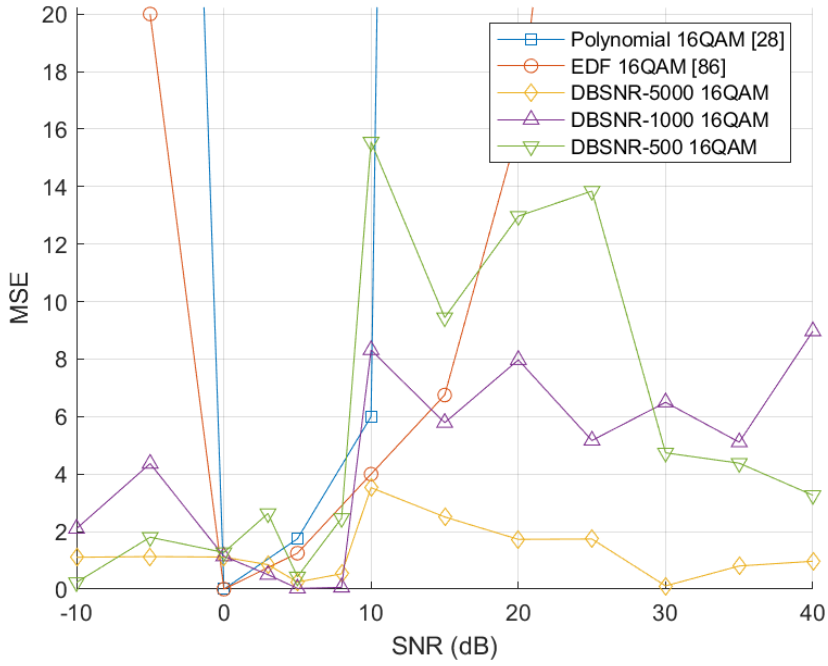


Figure 8.24: *MSE Against SNR (dB) of the Polynomial [28] and EDF [86] Estimators as well as DBSNR-5000, 1000, and 500 on 16QAM Data. Some Polynomial [28] and EDF [86] Datapoints Lie outside of the Y-axis Scale. Datapoints were Omitted to Better Display the Trends in DBSNR Performance.*

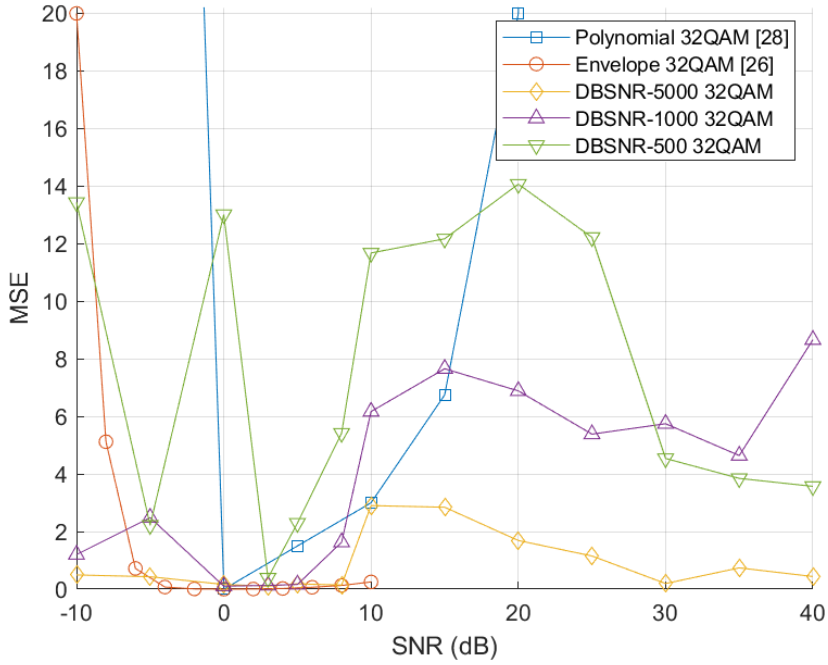


Figure 8.25: *MSE Against SNR (dB) of the Polynomial [28] and Envelope [26] Estimators as well as DBSNR-5000, 1000, and 500 on 32QAM Data. Some Polynomial [28] Datapoints Lie Outside of the Y-axis Scale, These Datapoints were Omitted to Better Display the Trends in DBSNR and Envelope Performance.*

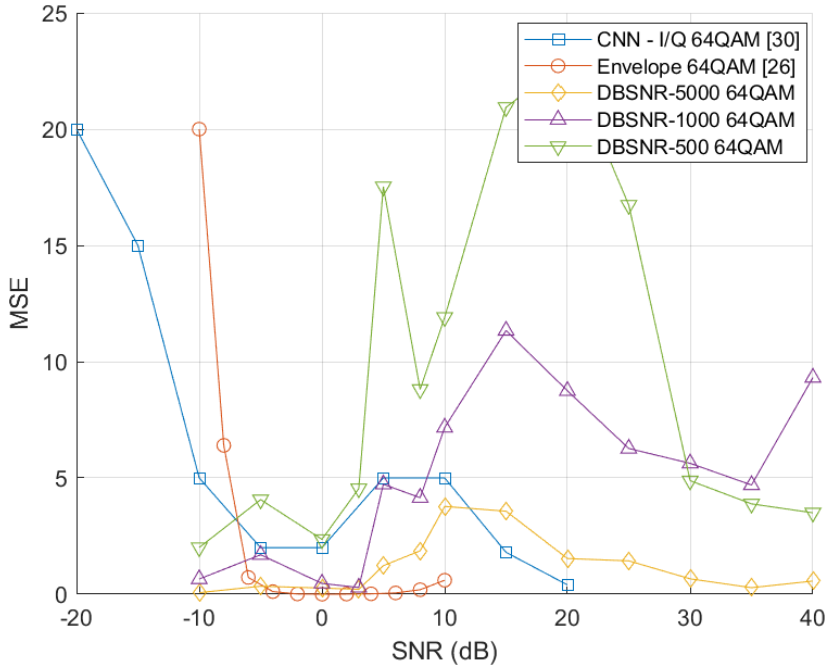


Figure 8.26: *MSE Against SNR (dB) of the I/Q Accepting CNN [30] and Envelope [26] Estimators as well as DBSNR-5000, 1000, and 500 on 64QAM Data*

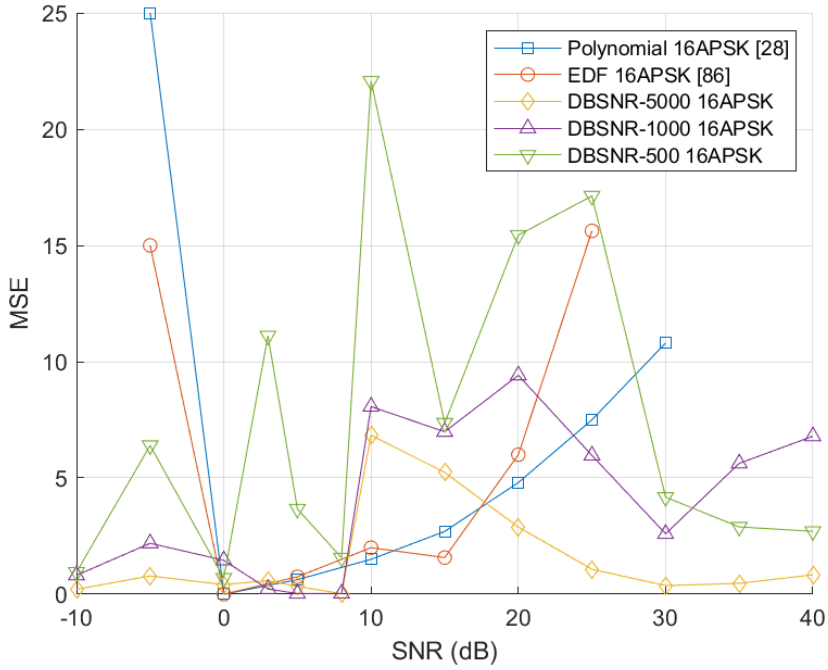


Figure 8.27: *MSE Against SNR (dB) of the Polynomial [28] and EDF [86] Estimators as well as DBSNR-5000, 1000, and 500 on 16APSK Data*

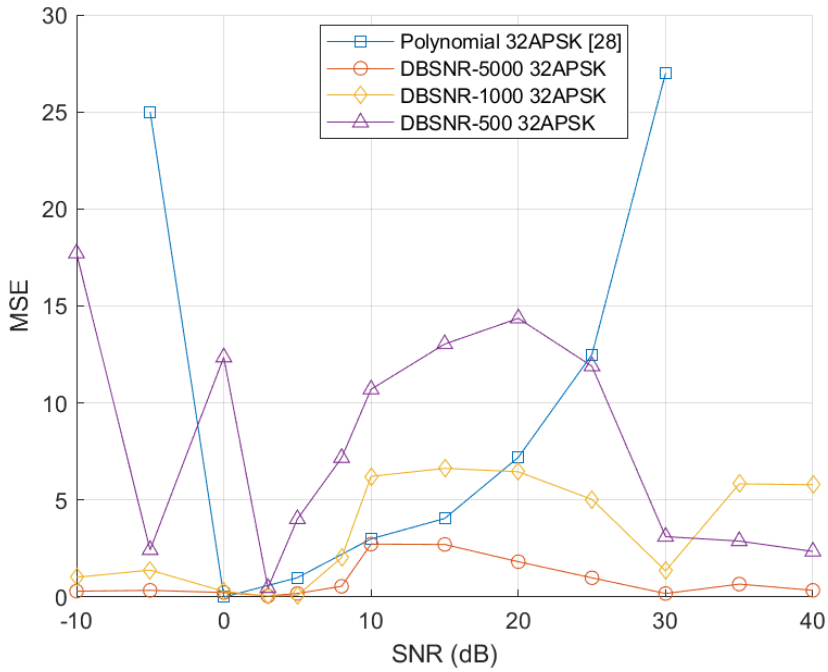


Figure 8.28: *MSE Against SNR (dB) of the Polynomial [28] Estimator as well as DBSNR-5000, 1000, and 500 on 32APSK Data*

Figures 8.24, 8.25, 8.26, 8.27, 8.28 each compare DBSNR with the stated works from the

literature in terms of the MSE against SNR for various QAM and APSK modulation schemes. In the literature review it was found that algorithmic estimators achieve a low MSE in a particular SNR range, which was typically 0dB to 10dB. The provided compararisons again demonstrate this to be the case, the system proposed by Zuo et al. [26] perhaps being the only exception although it appears that asymptotic effects would be demonstrated if data for SNRs greater than 10dB was provided. In every example shown here it can be seen that DBSNR-5000 and DBSNR-1000 achieve an MSE in the 0dB to 10dB SNR range which is either lower, comparable, or slightly greater than that of the algorithmic methods. Demonstrating that even within the SNR range at which the systems from the literature are optimal, DBSNR is capable of matching their performance. Furthermore, outside of the 0dB to 10dB SNR range the systems from the literature generally see large increases in MSE due to asymptotic estimation behaviour. While the 2 largest DBSNR systems may exhibit slight increases in MSE outside of this SNR range, the obtained MSE remains consistent and typically lower than that of the algorithmic and CNN-based systems. The 2 largest DBSNR configurations may therefore be evaluated to be superior SNR estimators across the full -10dB to 40dB SNR range as not only is the MSE approximately matched in 0dB to 10dB region, but asymptotic effects are not demonstrated.

The DBSNR-500 system in some cases performs as well as DBSNR-5000 and DBSNR-1000 but performance is unreliable and often large MSE peaks can be observed, therefore despite often matching the performance of the systems from the literature at certain SNRs and not displaying asymptotic effects, the variability of the MSE makes it a weaker SNR estimator.

8.4.3 Multi-Order NDA SNR Estimation Results Comparison

The final comparisons to draw are with the estimation techniques which demonstrate the capacity to operate on an array of modulation schemes. These systems were found to be the only which rigorously demonstrated performance across multiple orders of QAM, as such they are the systems with which comparison is most important as this functionality is a necessary requirement for dynamic modulation scheme adjustments. The first comparisons will be drawn with the polynomial fitting technique proposed by Xu et al. [27]. The results are provided in terms of SD, defined by the authors as shown in Equation 8.3.

$$\mathrm{SD} = \mathrm{RMSE} = \sqrt{E[(\hat{\mathrm{SNR}} - \mathrm{SNR})^2]} \tag{8.3}$$

Which is equivalent to the square root of the expected value of the MSE, the expected value is otherwise known as the mean, the SD is therefore equivalent to the root of the MSE or Root Mean Square Error (RMSE).

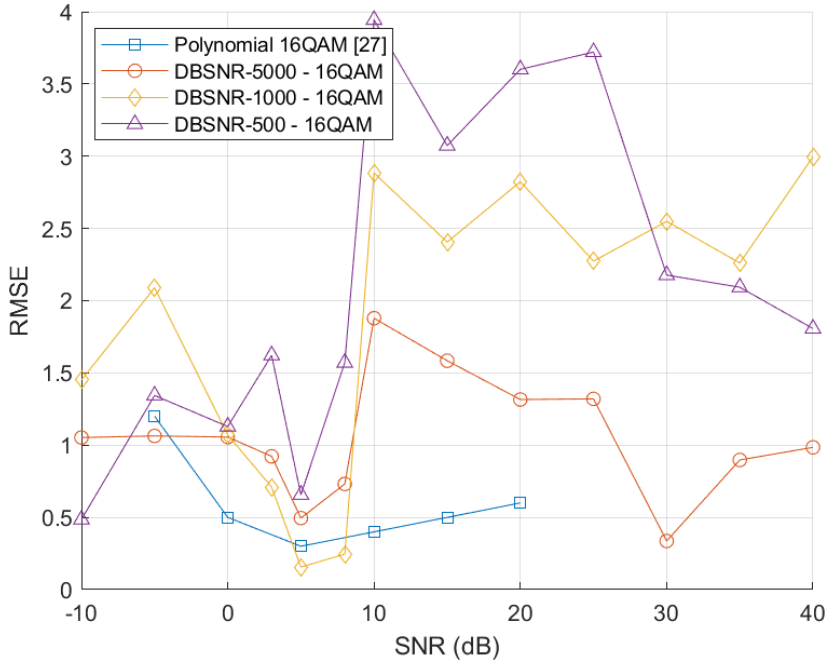


Figure 8.29: *MSE Against SNR (dB) of the Polynomial [27] Estimator as well as DBSNR-5000, 1000, and 500 on 16QAM Data*

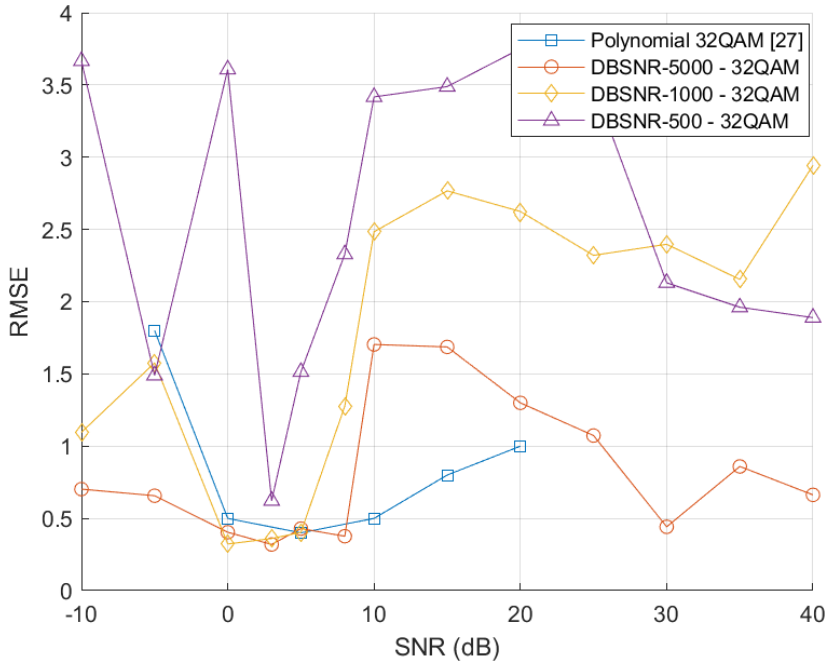


Figure 8.30: *MSE Against SNR (dB) of the Polynomial [27] Estimator as well as DBSNR-5000, 1000, and 500 on 32QAM Data*

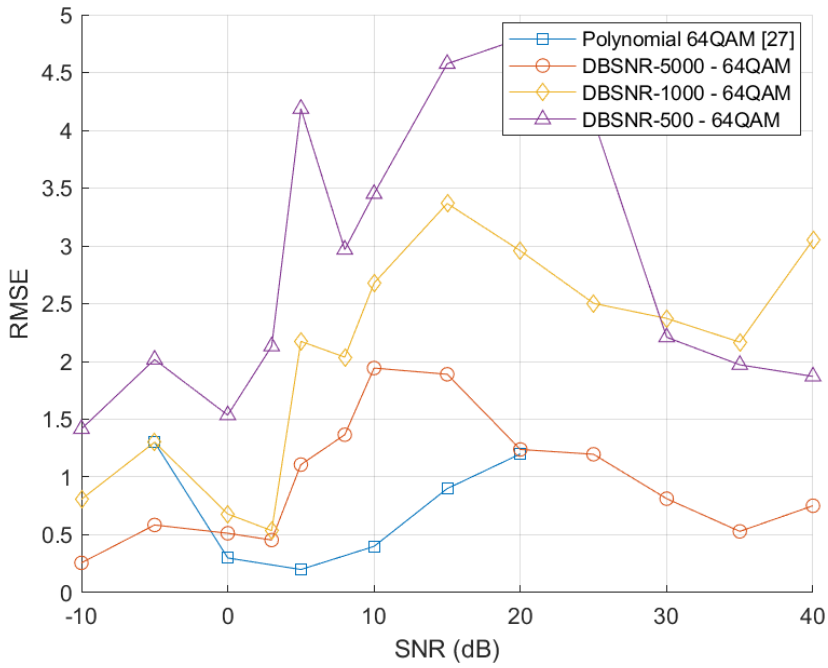


Figure 8.31: *MSE Against SNR (dB) of the Polynomial [27] Estimator as well as DBSNR-5000, 1000, and 500 on 64QAM Data*

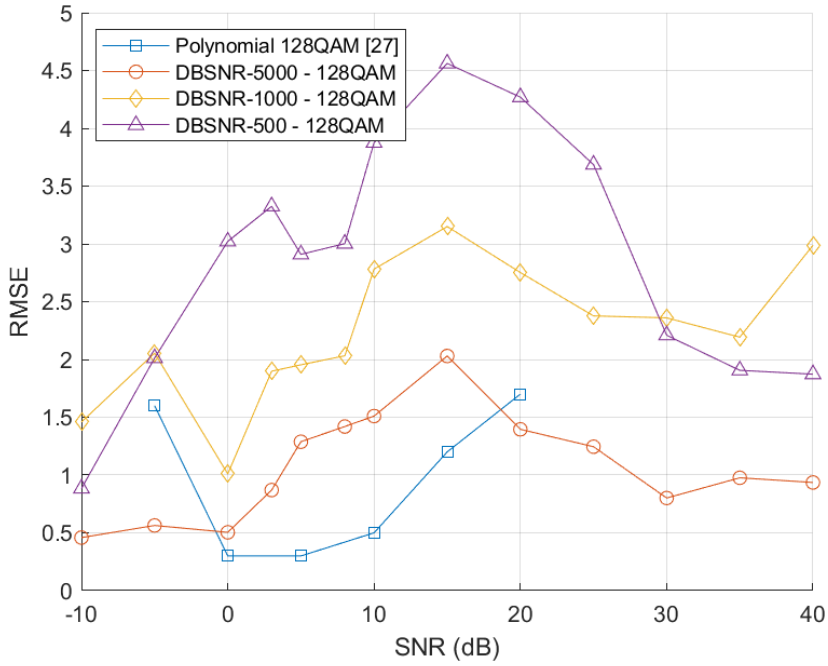


Figure 8.32: *MSE Against SNR (dB) of the Polynomial [27] Estimator as well as DBSNR-5000, 1000, and 500 on 128QAM Data*

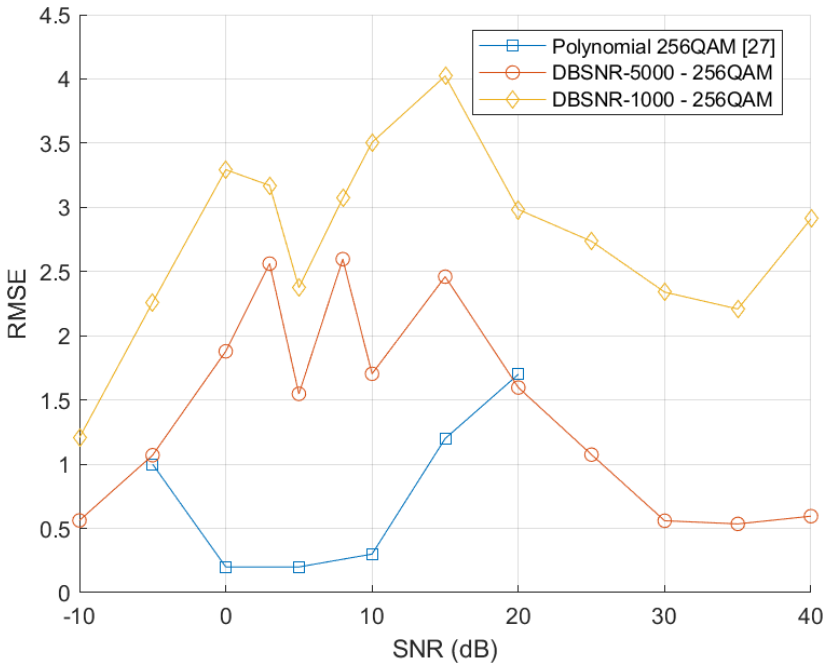


Figure 8.33: *MSE Against SNR (dB) of the Polynomial [27] Estimator as well as DBSNR-5000 and 1000 on 256QAM Data*

Figures 8.29, 8.30, 8.31, 8.32, 8.33 each compare DBSNR with the method proposed by Xu et al. [27]. The first trend in the comparisons to identify is that on 16QAM and 32QAM DBSNR-5000 and DBSNR-1000 is shown to be capable of matching or outperforming the RMSE obtained by the polynomial fitting technique in the 0dB to 10dB SNR range in which it performs strongest. As the modulation order increases the polynomial fitting technique tends to achieve a lower RMSE than that of DBSNR, with the difference increasing with modulation order, culminating with a large performance differential on 256QAM.

There is no data provided for the polynomial fitting method outside of the SNR range -5dB to 20dB, although if the performance trend seen at either extreme of this range were to continue then the RMSE would continue to increase, as has been seen to be the case with all other algorithmic estimators. Should this be the case then it would be expected that at least DBSNR-5000 would outperform the system from the literature at SNRs greater than 20dB, and perhaps DBSNR-1000 and DBSNR-500 as well. In some cases, the DBSNR systems are already seen to outperform the polynomial fit method at SNRs below 0dB, the performance difference would therefore be expected to increase further at -10dB.

The analysis provided here holds particularly for DBSNR-5000 and DBSNR-1000, there is only a single example where DBSNR-500 obtains a lower RMSE than the polynomial fitting method, and the difference in performance between the 2 systems in generally large. Therefore, it cannot be said that this configuration is as strong an estimator than the polynomial fitting method in any case, apart from perhaps at either extreme of the SNR range, but without data to support this it is unfair to assume.

DBSNR-5000 and DBSNR-1000 can therefore be judged to be SNR estimators with slightly inferior performance in the -5dB to 20dB SNR range on 16QAM and 32QAM, with the performance differential growing on higher orders of QAM in this SNR range. Outside

this SNR range it is thought that DBSNR may be superior but there is a lack of data to confirm this assumption. DBSNR-500 is inferior to polynomial fitting.

The 2D DBSCAN NDA SNR estimation mechanism proposed by Zhao et al. [69] was found to utilise the ratio of core points to total clustered points as the input to a polynomial fitting algorithm. It was discussed in Chapter 4 how the DBSCAN system proposed in this thesis does not make a distinction between core and non-core points, it was therefore required to develop an alternative mechanism of achieving NDA SNR estimation. The following comparison therefore not only evaluates the performance differential between two NDA SNR estimators, but also ultimately determines the effectiveness of the proposed enhancements to the DBSCAN algorithm for this purpose.

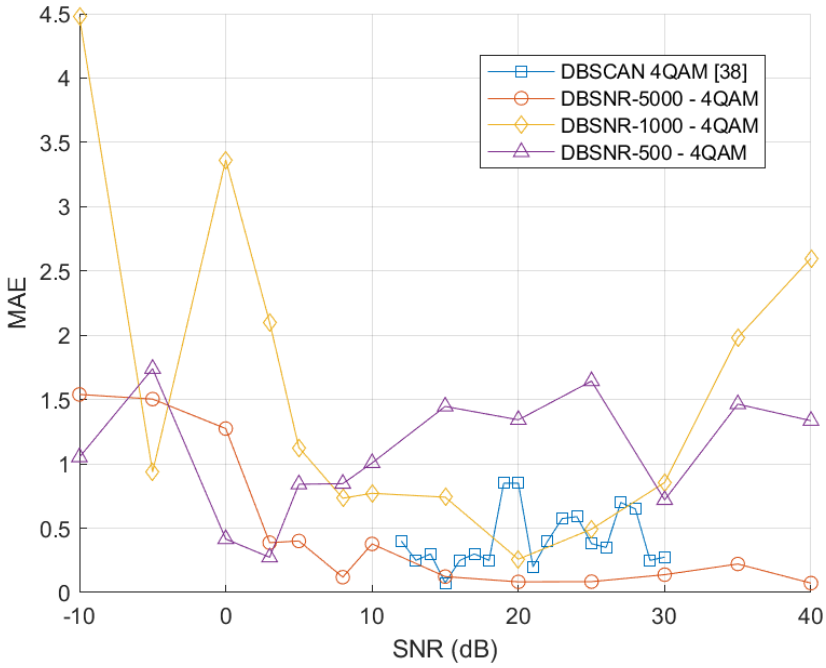


Figure 8.34: *MAE Against SNR (dB) of the 2D DBSCAN [38] Estimator as well as DBSNR-5000, 1000, and 500 on 4QAM Data*

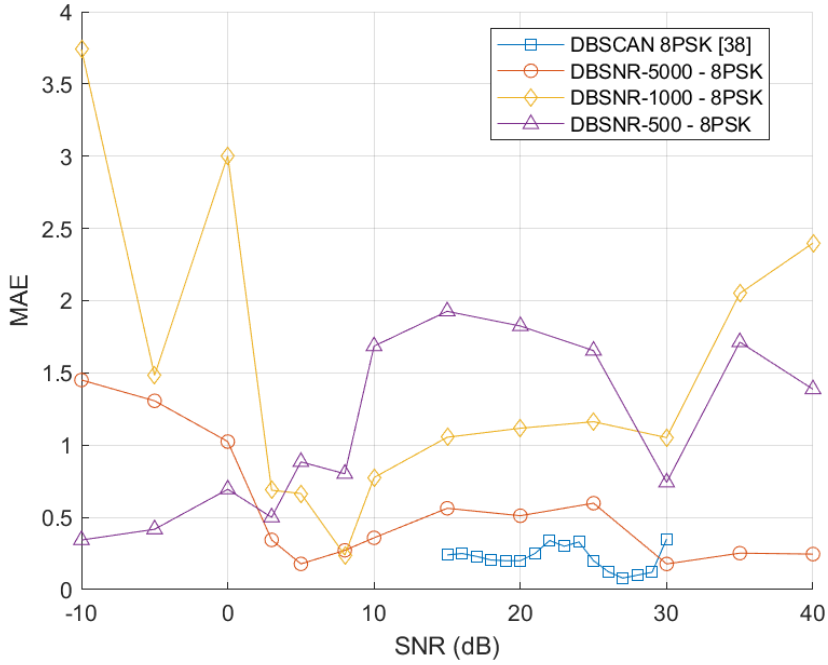


Figure 8.35: *MAE Against SNR (dB) of the 2D DBSCAN [38] Estimator as well as DBSNR-5000, 1000, and 500 on 8PSK Data*

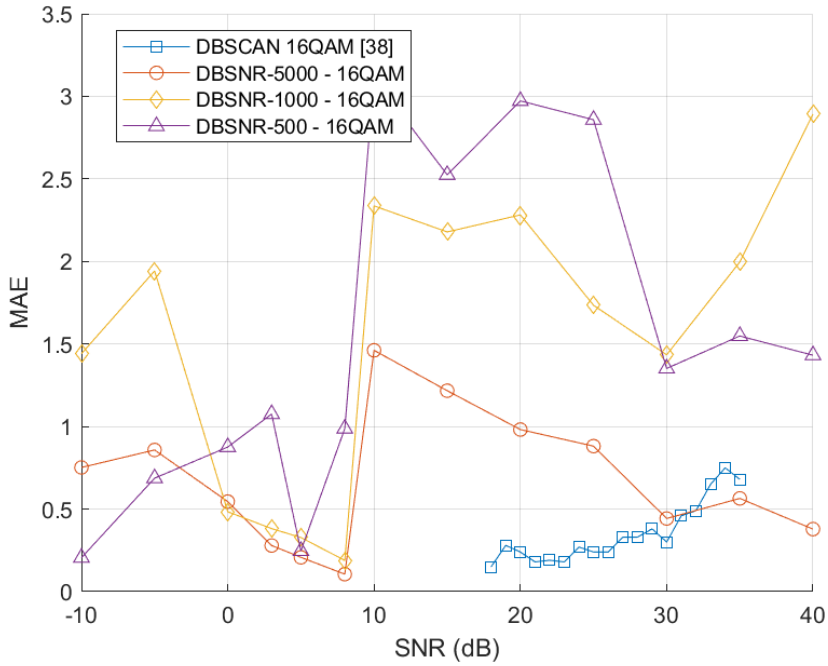


Figure 8.36: *MAE Against SNR (dB) of the 2D DBSCAN [38] Estimator as well as DBSNR-5000, 1000, and 500 on 16QAM Data*

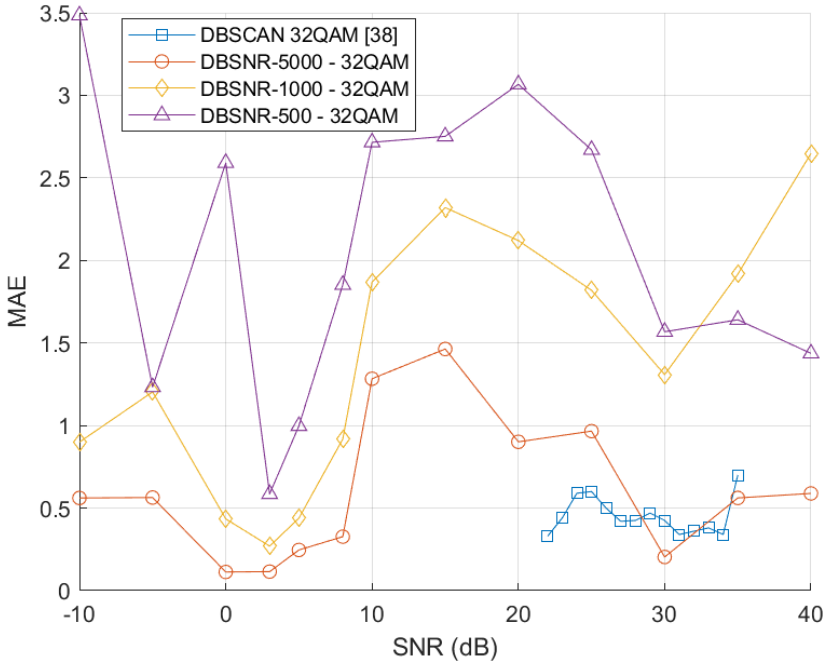


Figure 8.37: *MAE Against SNR (dB) of the 2D DBSCAN [38] Estimator as well as DBSNR-5000, 1000, and 500 on 32QAM Data*

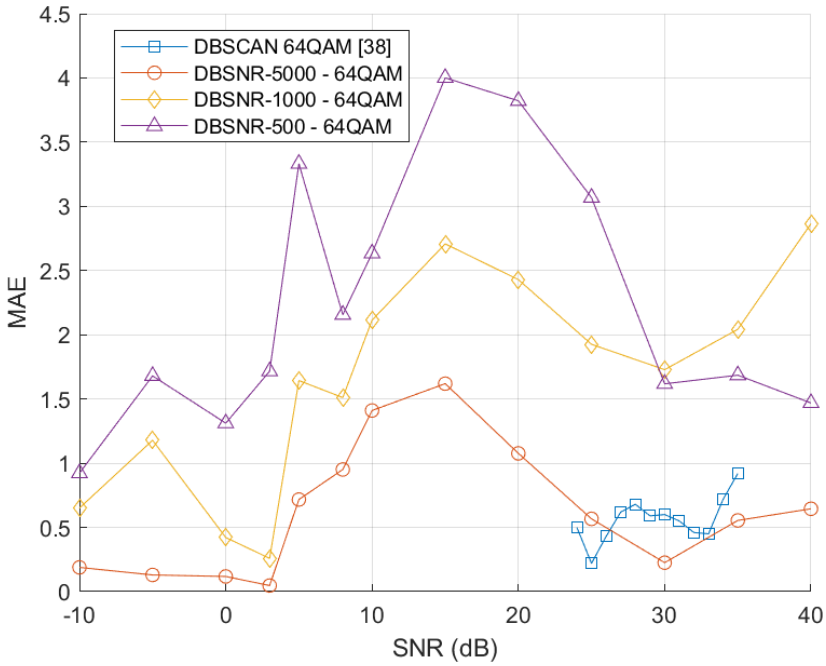


Figure 8.38: *MSE Against SNR (dB) of the 2D DBSCAN [38] Estimator as well as DBSNR-5000, 1000, and 500 on 64QAM Data*

Figures 8.34, 8.35, 8.36, 8.37, and 8.38 compare DBSNR with the 2D DBSCAN SNR esti-

mation method proposed by Zhao et al. [69] when various orders of QAM are utilised as the input data. Figure 8.34 shows that across all but one of the SNRs where there are results provided for the 2D DBSCAN system DBSNR-5000 achieves a lower MAE. DBSNR-1000 also obtains a comparable MAE, whereas DBSNR-500 cannot reach a MAE as low as the 2D DBSCAN method.

In Figures 8.35, 8.36, 8.37, and 8.38 it is only DBSNR-5000 which achieves a MAE values which are comparable to the 2D DBSCAN in the SNR ranges for which results are provided. It is only in Figure 8.38 that DBSNR-5000 outperforms 2D DBSCAN in the majority of cases, otherwise 2D DBSCAN tends to achieve a lower MAE across the reported SNR ranges.

In Chapter 3 it was hypothesised that the authors only provided results at particular SNRs as performance outside of the provided ranges was inferior. Performance of the same and similar systems on the task of modulation classification suggested that the DBSCAN algorithm suffered from poor performance at low SNRs, therefore it is reasonable to assume that this would be the case for SNR estimation. If it is the case that the 2D DBSCAN algorithm suffers from reduced accuracy outside the ranges for which data is provided, then perhaps DBSNR would outperform it in this regard as consistently low MAE is demonstrated across the full SNR range. If this is not the case, and similar accuracy is maintained from -10dB to 40dB SNR then DBSNR-5000 and DBSNR-1000 in general matches the obtained MAE at SNRs below 10dB in 8.35, 8.36, 8.37, and 8.38. DBSNR-500 reaches an MAE comparable to 2D DBSCAN in 8.35 and 8.36. Disregarding any assumptions about performance outside of the SNR range for which data is provided, it can be said with certainty that DBSNR is demonstrated to be capable of SNR estimation within a larger SNR range than 2D DBSCAN.

The comparisons and analysis provided here have demonstrated that the modifications made to the DBSCAN algorithm in this thesis have enabled the creation of an SNR estimation system capable of achieving an MAE which is comparable to the SNR estimation algorithm which utilises the traditional DBSCAN algorithm, as long as a dataset size of 5000 is employed. However, it is only on 4QAM and 64QAM where the proposed system consistently achieves a superior MAE. However, the proposed systems have been demonstrated to provide accurate NDA SNR estimation across a wider range of SNRs than was demonstrated with the traditional algorithm.

8.5 NDA SNR Estimation Conclusion

This chapter proposed a method of how the modified DBSCAN algorithm can be utilised as an NDA SNR estimation system. The mechanism of using the number of argument and magnitude clusters found by the algorithm as a proxy for the constellation diagram density was outlined. It was shown how as the SNR reduced so too did the number of clusters found by the DBSCAN algorithm; the relative positioning of feature clusters could then be used to train a regression model to predict the SNR based upon an input feature pair.

The performance of the method when a dataset size of 250, 500, 1000, and 5000 was utilised was demonstrated in the SNR range -10dB to 40dB on QAM signals of orders 2 to 1024, PSK signals of orders 8 to 32, and APSK signals of orders 16 to 128. Results were provided in terms of MSE against SNR and average estimated SNR against true SNR. It was found that NDA SNR estimation accuracy increased with larger dataset sizes and in general

reduced with increasing modulation order. DBSNR-5000, 1000, 500, and 250, were found to be capable of SNR estimation of QAM orders up to and including 128 with a maximum MSE value of 5.3, 18.7, 22.9, and 152.8 respectively, the maximum average estimation error was also found to be 1.3dB, 4.3dB, 3.3dB, and 9.6dB. Only DBSNR-5000 and 1000 could estimate the SNR of QAM orders of 256 and greater, in general performance on these modulation schemes was found to be inferior to the lower-order QAM signals, with even DBSNR-5000 obtaining MSE values up to 27.6 at low SNRs, performance on these signals at high SNRs remained strong as MSE values below 2 were achieved at 20dB SNR and higher.

The 2 largest of the proposed systems were found to exhibit stronger performance on APSK and PSK signals than with QAM as the average MSE remained lower and more consistent. Despite this, similar peak MSE values were obtained as DBSNR-5000, 1000, 500, and 250 achieved a peak of 6.8, 10.9, 33.6, and 132.3 respectively, the largest observed error was found to be 2.9dB, 2.4dB, 4dB, and 8.5dB.

Comparisons between DBSNR-5000, 1000, 500 and the strongest estimators which were found in the literature were drawn by using the MSE, RMSE, and MAE statistics. It was decided not to include DBSNR-250 in these comparisons due to weak estimation performance. It was found that in comparison to algorithmic estimators, DBSNR-5000 and DBSNR-1000 could match the achieved performance in the 0dB to 10dB range in which the majority of the systems from the literature were optimal, furthermore, it was also found that all DBSNR systems in general maintained lower MSE, RMSE, and MAE across a wider SNR range than all algorithmic estimators [22, 26–28, 86]. Comparisons with DL-based estimators revealed that CNNs which accept constellation diagrams [29], covariance matrices [23], and a highly processed I/Q input [24] could obtain a slightly lower MSE than all DBSNR configurations and maintain this level of performance across a similar SNR range, although this class of estimator was not demonstrated to be capable of estimating the SNR of modulation schemes other than QPSK. Comparisons with the best performing algorithmic estimator which possessed the capability of estimating the SNR of an array of QAM orders [27], found that DBSNR-5000 and DBSNR-1000 could again match the RMSE of this system in the SNR range in which it was optimal in some cases but the system from the literature in generaly outperformed DBSNR in this range. Outside this SNR range is was hypothesised that DBSNR would provide superior accuracy but in the absence of provided results no conclusive statement could be made.

Finally, the estimation performance of DBSNR was compared with a 2D DBSCAN NDA SNR estimator [69]. It was found that DBSNR-5000 could match the MAE achieved by this system within the range of SNRs for which results were provided in two cases, otherwise the prior DBSCAN implementation [69] tended to exhibit a lower MAE. However, DBSNR-5000 was demonstrated to be capable of maintaining a strong MAE outside this SNR range. It was thought that results were not provided outside this SNR range due to diminished performance, if this is the case then the modifications to the DBSCAN algorithm proposed in this thesis have resulted in a superior method of achieving NDA SNR estimation than previously achieved by utilising the DBSCAN algorithm, however, a dataset size of 5000 is a requirement as DBSNR-1000 and 500 did not achieve a MAE as low as the 2D DBSCAN estimator.

To the best of the author's knowledge, the DBSNR mechanism has been demonstrated to be capable of accurately estimating the SNR of a wider range of modulation schemes than

any previous work. The system can also provide equivalent estimation performance to many algorithmic estimators in the SNR range in which they are optimal whilst also displaying no asymptotic effects. It achieves slightly weaker accuracy than certain CNNs on QPSK data. Crucially the system offers SNR estimation capabilities with only modifications to stored variables within the AMC implementation. Therefore, when implementing the DBSCAN modulation classifier, designers can also achieve NDA SNR estimation capabilities which are competitive with the state-of-the-art NDA SNR estimators in many cases. The DBSNR mechanism therefore offers a highly efficient means of achieving SNR estimation within a CR which includes the DBMC modulation classifier, but the performance shown in this chapter also demonstrates that this method can also be utilised for NDA SNR estimation as a standalone system as it achieves highly competitive performance across a large range of SNRs and modulation schemes. The caveat to this conclusion is that for optimum performance a dataset size of 5000 is a requirement, although the implemented DBSNR-1000 also achieved accuracy competitive with the state-of-the-art in the majority of comparisons.

Chapter 9

Discussion, Conclusions, and Future Work

All the work which has been completed throughout the course of this PhD programme has now been presented. This chapter will analyse the performed work and obtained results, combining the achieved hardware implementation with the NDA SNR estimation and AMC performance. The key contributions towards the field of AMC and NDA SNR estimation which this thesis has made are synthesised. Finally, recommendations for possible future avenues of development for the proposed systems are outlined.

9.1 Discussion

This thesis has addressed the critical challenge of implementing efficient AMC and NDA SNR estimation for resource constrained CR systems. The optimisations to DBSCAN have yielded a system that balances accuracy with power efficiency and hardware utilisation. This discussion synthesises the key findings across all aspects of this research and evaluates their significance within the broader context of 5G and future wireless communications systems.

9.1.1 Algorithmic Enhancements to DBSCAN

The decomposition of the DBSCAN algorithm from a 2D to 1D implementation represents a fundamental contribution to both clustering techniques and signal processing. The computational complexity reduction from $O(n^2)$ to $O(n)$ directly addresses a primary limitation that has historically prevented the deployment of clustering-based classification in resource-constrained applications. This improvement surpasses purely computational complexity improvements as the resulting algorithm maintains, and in some cases enhances, classification accuracy.

The transformation from 2D constellation data into separate 1D magnitude and argument components introduced three critical capabilities. First, it enabled differentiation between modulation schemes of equivalent order, which had been impossible with previous DBSCAN implementations [38]. This represents a significant advancement for cognitive radio systems that must distinguish between schemes like 16QAM and 16PSK. Second, this 1D decomposition introduced partial robustness to carrier frequency offset, providing a level of immunity

to common channel impairments without requiring additional compensatory mechanisms. Third, the NDA SNR estimation accuracy was found to be equivalent to prior works which utilised DBSCAN in the SNR ranges in which they were found to be optimal as well as demonstrating consistent accuracy across a wider SNR range.

When compared with other feature extraction methodologies, the modified DBSCAN algorithm achieved superior ability to classify a broader range of modulation schemes as a well as higher orders. The elimination of the computationally intensive *rangeQuery* function and integration with an efficient sorting mechanism produced an algorithm which is suitable for real-time operation. A capability which was found to be rare in the literature review.

9.1.2 Key Hardware Implementation Results

The hardware implementation results demonstrate that the algorithmic optimisations translate effectively to resource savings. The proposed design was implemented in such a way that a continuous datastream could be processed in real-time.

The novel insertion sort implementation was found to be both a strength and a limitation of the system. While it effectively sorts incoming data with zero effective latency by operating during required waiting periods, its resource utilisation scales linearly with the dataset size n. This resulted in the sorting unit becoming the largest module in the system by a significant margin in larger configurations, accounting for 90.5% and 92.7% of the DBMC-1000 FF and LUT utilisation respectively. It was also found that the best performing DBMC-5000 system could not be implemented in hardware due to the size of this module and the system clock frequency had to be reduced to 142.86MHz to enable it to pass timing checks. Despite these limitations, the real-time sorting capability enabled a continuous processing pipeline that distinguishes the proposed system from batch-processing alternatives.

In contrast, the modified DBSCAN algorithm's hardware implementation is very efficient, requiring only 54 LUTs and 60 FFs. 2 insertion sort and DBSCAN modules were found to require a slice utilisation reduction of 64.1% in comparison to an efficient DBSCAN implementation from the literature [114]. Thus demonstrating that the algorithmic optimisations result in a highly efficient implementation even if the size of the dataset is limited.

Comparative analysis revealed that while the largest configuration (DBMC-1000) required similar FF and LUT resources to the state-of-the-art CNN implementations [18, 39], it achieved a 71.7% reduction in power consumption and a 0.248μs reduction in terms of latency. This power efficiency is particularly significant for resource-constrained mobile and edge devices where battery life is a limiting factor. The low latency demonstrates that the system would be capable of reacting to the dynamic fast-changing changing channel conditions inherent with high frequency transmission in crowded areas.

9.1.3 Contributions Towards DBSCAN Hyperparameter Optimisation

The development of automated hyperparameter selection methods addresses a limitation of the practical application of DBSCAN. The traditional elbow point method for finding strong ε values introduces subjectivity when there is not a distinctive change in k-distance graph gradient. The proposed RMS and $d-1$ methods demonstrated not only automatation but also also performance improvements, with AMC accuracy improving by 9.8% at some SNRs.

The relationship between the dataset size, *minPts*, and ε was analysed in depth, revealing their inderdependencies and effects on feature space scale as well as feature cluster arrangement and spacing. The findings that larger dataset sizes create expanded feature space with greater cluster separation provides and that *minPts* has the inverse effect provides valuable insights for future work on this system to build upon.

The adaptation to fixed-point representation for hardware values required careful quantisation strategies. The discovery that optimal ε values could be found by approximating high precision floating-point values with quantised values and removing an amount equal to 1-bit of precision ensured consistent performance between software and hardware implementations.

9.1.4 Performance Evaluation

The classification accuracy results demonstrate that the proposed system achieves state-of-the-art performance under specific conditions. DBMC-1000 obtained 100% accuracy on low-order modulation schemes at SNRs as low as 30dB, matching the performance of the most accurate low-order classifiers [20, 47]. When utilising the complementary dataset (4QAM, 16PSK, 64APSK, and 256QAM), DBMC-1000 maintained 100% accuracy down to 10dB SNR, matching the accuracy of the strongest DL-based system [57]. However, it was found that on broader datasets of up to 17 different signals that the proposed system was unable to match the robustness to SNR impairments which was demonstrated by DL models, generally only achieving over 90% accuracy above an SNR of 30dB. Similarly, while performance on a dataset representative of currently employed 5G modulation schemes was improved over the largest dataset, 100% accuracy was achieved only above 25dB.

Statistical analysis of the NDA SNR estimation capabilities revealed consistently low estimation error across a wider range of modulation schemes than had ever been demonstrated by prior works. The DBSNR-1000 system achieved a maximum estimation error of 5dB on signals up to a modulation order of 512 which only occurred at -10dB, generally the estimation error was below 1dB. This performance compares favourably with existing algorithms found in the literature which were generally found to only perform well within the 0dB to 10dB SNR range and did not demonstrate applicability to as wide a range of modulation schemes as was demonstrated by the proposed system. The inverse was found to be the case when comparing with DL systems, they were found to exhibit marginally superior estimation accuracy yet were not shown to capable of estimating the SNR of any modulation scheme bar QPSK and in one case 64QAM.

Comparison between DBMC/DBSNR configurations showed that both systems benefitted from increased dataset sizes, with a size 5000 dataset achieving the best performance in the fields of both AMC and NDA SNR estimation. The trade-off between implementation size and accuracy is therefore optimised around the DBMC/DBSNR-1000 configuration, balancing strong performance with reasonable hardware requirements.

When evaluated as an integrated system, the dual AMC and SNR estimation functionality represents a signficant advancement in efficient cognitive radio implementation. The ability to perform both functions with essentially the same hardware structure, requiring only changes to stored parameters, offers unprecedented resource efficiency for adaptive systems. This dual functionality addresses both key requirements for cognitive radio systems, offering the ability to identify the modulation scheme and estimating the channel quality with minimal additional hardware overhead.

This research demonstrates that through algorithm redesign and hardware optimisation, clustering-based approaches can achieve significantly less power and comparable hardware resources to state-of-the-art DL systems. The resulting system represents a viable path towards implementing CR functionality in resource constrained devices where power efficiency is paramount, with the caveat that specific modulation schemes must be employed to achieve state-of-the-art modulation classification accuracy.

9.2 Conclusion

This thesis has addressed the critical challenge of implementing efficient modulation classification and NDA SNR estimation systems suitable for mobile and edge devices in 5G networks. The research was motivated by the need to mitigate variable path loss issues inherent in high-frequency communications while maintaining hardware efficiency for practical deployment.

The primary contribution of this work is the development of a modified DBSCAN algorithm that matches the classification accuracy achieved by prior works while significantly reducing computational complexity. By transforming 2D constellation data into 1D magnitude and argument components, the algorithm achieves three key improvements: (1) the introduction of the ability to differentiate between same-order modulation schemes, (2) a reduction in the worst-case computational complexity from $O(n^2)$ to $O(n)$, and (3) the elimination of extraneous functionality that consumes hardware resources.

The hardware implementation introduces a novel approach to the insertion sort algorithm that effectively performs sorting operations during required waiting periods, maintaining the overall $O(n)$ computational complexity while enabling a fully pipelined datapath. When combined with the streamlined DBSCAN implementation the system achieves a 65% reduction in slice utilisation compared to the most efficient DBSCAN implementation found in the literature. When compared with state-of-the-art AMC implementations a reduction in terms of FFs and LUTs was not achieved by the best-performing proposed system, only approximately equivalent utilisation was achieved. However, the proposed system requires the least DSP utilisation of any system which utilises this element and requires no RAM. Significant improvements in terms of power consumption and latency were also obtained.

The process of optimising the proposed system was outlined in depth. Two similar novel methods of finding strong ε hyperparameter values were proposed. Utilising these methods was shown to result in a maximum classification accuracy increase of 9.8% over traditional optimisation methods. Methods for optimising ε for the 10-bit datapath of the proposed system were also provided.

Performance evaluations demonstrated that the proposed system achieves comparable AMC accuracy to state-of-the-art DL approaches when working with complementary modulation scheme sets. Performance was found to be inferior to DL systems on equivalent datasets. It was also found that optimum performance was obtained with a feature-extraction dataset size of 5000 which was too large to implement in hardware with the proposed system structure. Comparisons with NDA SNR estimation systems showed that the proposed method achieved a more consistent accuracy across a wider SNR range and set of modulation schemes than has been demonstrated in the literature. However, optimum performance was also only obtained with the unimplementable 5000-point dataset size. The dual functionality

of SNR estimation and modulation classification further enhances the efficiency of the proposed system, although the obtained results show that the system is a viable candidate for either function individually.

In summary, this research has provided a number of significant and novel contributions to the field of hardware-implemented AMC and NDA SNR estimation technologies. The proposed system has improved upon the state-of-the-art CNN modulation classifiers in terms of power consumption and latency despite similar FF and LUT utilisation. Furthermore, on a dataset of signals similar to what is currently used in 5G systems it has demonstrated superior accuracy across a wide SNR range. The SNR estimation functionality was demonstrated to be superior to all systems in the literature in terms of the applicability to a broader set of modulation schemes as well as the performance across a wider SNR range. Furthermore, a practical hardware implementation for an SNR estimation system was proposed when the majority of research in this field focuses primarily on software implementation. The dual AMC and SNR functionality with a single implementation provides large gains in efficiency as there could be no system found which performs both functions with a single hardware implementation. The efficient hardware implementation combined with competitive AMC accuracy and competetive NDA SNR estimation performance demonstrates that this system has the potential to see deployment as part of a CR in future generations of wireless systems. Outside of the fields of AMC and NDA SNR estimation, there are a number of contributions made to clustering methods and efficient sorting in hardware. The 1D decomposition of the DBSCAN algorithm demonstrated how a complex algorithm could be made more efficient should the employed dataset be conducive to such a decomposition. The devised methods to finding optimal ε values provide an automatic means of determining hyperparameter values which increase accuracy over traditional methods. The proposed insertion sort implementation introduces a new method of sorting a streaming input with a worst-case computational complexity of $O(n)$ and results in a n sized dataset being sorted in essentially 0 clock cycles.

A number of limitations were identified with the proposed system. Firstly, the AMC accuracy on a large dataset was found to be inferior to other implementations, particularly DL approaches. While 100% accuracy was often obtained at high SNRs, accuracy below 20dB was found to be poor. SNR estimation accuracy was demonstrated to be inconsistent with large spikes in MSE at particular SNR values. The proposed hardware structure did not achieve a FF and LUT utilisation lower than the DL systems which were aimed to be improved upon. The high utilisation was primarily due to the proposed sorting algorithm implementation which also limited the system clock rate to 142.86MHz. The most impactful consequence of the hardware structure was the inability to implement a dataset size of 5000, this inhibited the strongest AMC and SNR estimation performance from being achieved by the hardware implementation. Performance was only demonstrated on signals with AWGN signal impairments, further research must be conducted to determine the effectiveness with other forms of interference.

9.3 Future Work

The DBSCAN-based systems have shown competitive accuracy and hardware implementation statistics whilst achieving real-time operating capabilities. Despite the breadth of work completed and quality of the results gained there remains various avenues of research which

are yet to be explored in this area of research, but there is unfortunately no time remaining to take the research conducted in this thesis any further. In this section various areas of research which may lead to improvements to each topic discussed in this thesis will be outlined in this section in the hopes that the reader may choose to pursue them for their own research.

9.3.1 Problems with the Sorting Algorithm Implementation

When presenting and discussing of the DBMC and DBSNR results the accuracy of each algorithm with a dataset size of 5000 was included, both results showed that a dataset size of 5000 lead to a higher level of accuracy than the maximum dataset size which was implemented on an FPGA in this thesis. It is unfortunate that the hardware implementations discussed in this work did not reach the full potential which the DBSCAN feature extraction algorithm is capable of, it was explained that scaling the hardware designs to be capable of accommodating a dataset size of 5000 would lead to a utilisation which was too large for the target FPGA to be capable of accommodating. The choice to design the system in the manner presented in this work was to minimise latency and enable the ability to operate in real-time rather than on batches of data, the system was heavily optimised for this functionality and strong performance in terms of latency was achieved but it was this prioritisation of pipelining that led to poor hardware utilisation scaling when the dataset size was increased. This is solely due to the sorting algorithm developed to sort the data stream as it entered the hardware. The sorting algorithm was implemented in such a way that all datums within a batch were required to be stored in individual registers connected to a comparator, this meant that an n sized dataset required n registers and comparators. Ideally, storing large amounts of data should be performed using the available BRAM on the FPGA but the sorting algorithm required access to all data in the dataset simultaneously, something that using BRAM prohibits due to the read and write operations introducing a bottleneck. Creating a sorting algorithm on an FPGA capable of sorting up to 5000 datapoints without being prohibitively large must require the storage of a section of the total dataset in RAM whilst the remaining data is sorted in a smaller sorting unit designed to operate on batches of data. Once the smaller batch of data had been sorted it will be placed back into BRAM and sorting of another batch of data can begin, this process will iterate until the complete dataset has been sorted. This iterative approach eliminates any possibility of a pipelined data path and therefore the system will be incapable of operating on a real-time data stream but there would be significant savings in terms of hardware utilisation, possibly savings in terms of power consumption, and most importantly the ability to operate on datasets of sizes up to 5000 which will allow for DBSCAN to achieve its maximum performance levels.

If such a sorting algorithm was implemented alongside the DBSCAN feature extraction system, there is also potential to investigate the effects of increasing the value of the *minPts* hyperparameter. In Chapter 5 it was shown how increasing the *minPts* hyperparameter resulted in denser feature clusters with increased separation. It has also been found in prior works which utilise DBSCAN that increased values of *minPts* provide increased robustness to noise in the clustering process [24]. Combining the increased noise robustness with the increased separation between feature clusters may enable gains to be made in terms of the system's accuracy when classifying low SNR data. Ultimately, the value of *minPts* was limited to 2 in the proposed work due to the size of the sorting unit, but this avenue of research may prove fruitful if it was to be investigated.

Another consequence of the sorting algorithm implementation was the need to reduce the clock speed. Many methods were tried to enable this module to pass timing checks at a clock rate of 200MHz or greater, however none succeeded. It was found that with an 10-bit datapath and incredibly forceful constraints that timing checks could be passed but when implemented and tested on the FPGA itself there were frequent setup/hold violations and metastability problems. Ultimately to enable the pipelined system structure it was conceded that a reduced clock rate was required.

The final and most impactful consequence of the proposed sorting algorithm was the extremely large FF and LUT utilisation, particularly with the 2 larger implemented systems. The DBMC-1000 system being equivalent size to the highly optimised CNNs was a direct result of the sorting algorithm scaling poorly with dataset sizes.

Should an alternative method of performing sorting be devised it is possible to achieve the maximum level of DBMC performance demonstrated with the DBMC-5000 results, achieve a vastly smaller utilisation size, and operate at a clock frequency of 200MHz, however to the best of the author's knowledge this may have to come at the cost of the pipelined datapath.

9.3.2 DBSCAN Parallelisation

While the sorting algorithm was designed to minimise latency, a major drawback of the DBMC and DBSNR design is the DBSCAN operation itself. The operation is performed by sequentially iterating through the array of sorted values, taking the difference between values d and $d-1$ and comparing the difference to ε. This is a slow method which is hardware efficient, but a much faster method could involve splitting the sorted array into two and performing DBSCAN on the two arrays in parallel. This would double the hardware required for this operation but halve the latency, and as has been shown in Section 5.5, operation of the DBSCAN module comprised the majority of the total latency of the entire system meaning that halving the latency of this modules would nearly halve the total latency. Doubling the number of DBSCAN modules does not contribute significantly to the total implementation size of the system as the DBSCAN module is small in comparison to other modules, all that is required is two 10-bit registers to hold data, two 10-bit counters to store the point and cluster counts, an 10-bit full adder to perform subtraction, and two LUTs to store $minPts$ and ε values. There would also be some upgrades to the control system required to manage the parallel operations as well as changes to the way data is loaded out from the sorting unit, but these would contribute little to the total implementation size, it would just be difficult to implement in a bug free manner. For example, there would need to be some circuitry to manage the likely case that a cluster lies at the point which the dataset is split into two, in which case the system should recognise this and handle it appropriately.

This parallelisation of DBSCAN could be taken to further degrees, by splitting the dataset into 4 and applying each to a separate DBSCAN module the latency could be reduced by four times. The same goes for a split into 8 or 16 and so on. There is no reason why this process could not be taken to the nth degree and take the difference between all points of data in the array simultaneously. An implementation which can do this could be structured as shown in Figure 9.1:

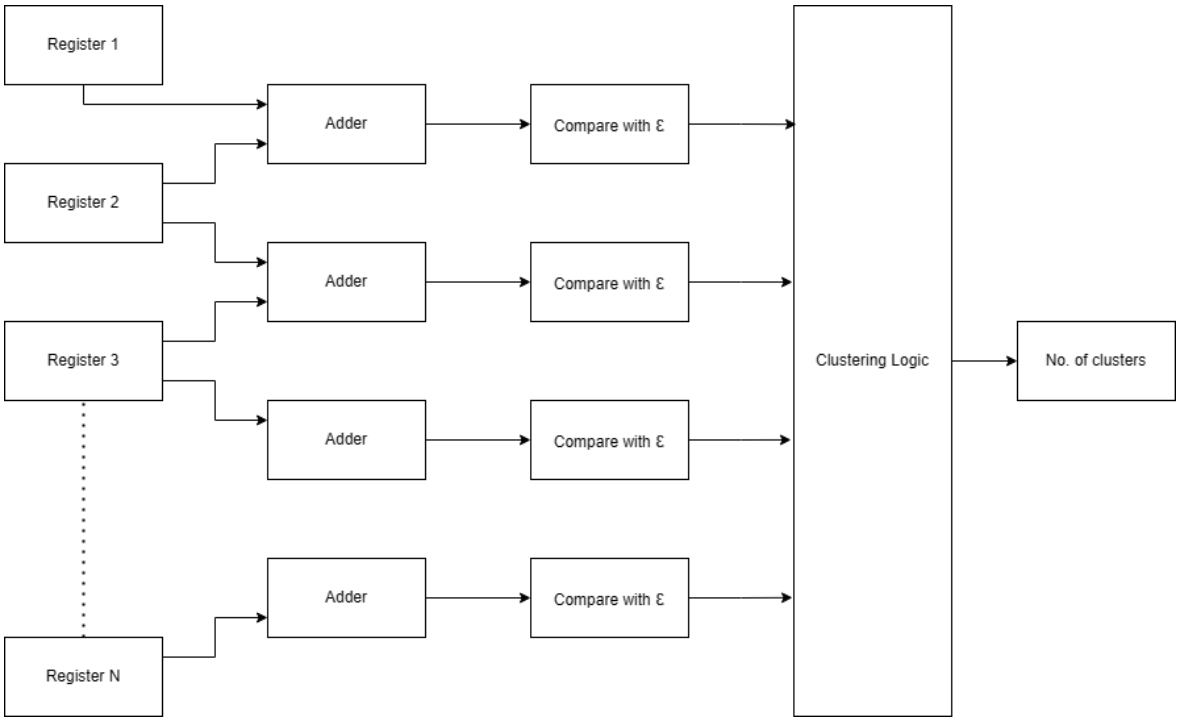


Figure 9.1: *Theoretical Highly Parallelised Modified DBSCAN FPGA Implementation Block Diagram*

An adder and comparator attached between each register holding datums which comprise the entire dataset can compute a 1D DBSCAN result with combinatorial logic. It may therefore be possible to compute DBSCAN within a clock cycle if the clock period is longer than the propagation delay. It would also be required to include additional logic to handle the output from the comparators and as well as a method of ensuring the *minPts* functionality is performed. This architecture would be very costly in terms of hardware size, with $n - 1$ adders and comparators required, therefore this may be unviable to implement with a dataset size of $n = 1000$, but for lightweight DBMC implementations such as DBMC-250 this may be a viable implementation if near-instantaneous classification is required at the cost of implementation size and power consumption. The 1D DBSCAN algorithm can therefore be implemented in a range of configurations, hardware sizes and power consumption can be increased in favour of reduced latency by scaling the number of parallel DBSCAN processes, an engineer designing a system using this technology could analyse the requirements of their use-case and target platform and decide the optimum ratio between size and latency.

In the case of the work in this thesis, as the sorting module had already been optimised for minimal latency it would have been advantageous to at least split the DBSCAN algorithm into two parallel parts. Implementing DBSCAN in this way would cut the latency of each DBMC system in half, allowing for every system configuration presented in this thesis to outperform every competitor in the literature in terms of latency. The cost of the additional hardware and more complex control logic is miniscule in comparison to the rest of the system and therefore would not have increased the total hardware cost significantly.

9.3.3 Additional Channel Impairments

The final drawback of the research proposed in this thesis is the lack of inclusion of channel impairments such as Rayleigh fading or other types of non-Gaussian interference. The majority of authors in the literature do not investigate the effects of such impairments on their systems and therefore AWGN channel impairments were the primary focus in order to establish baseline performance and to aid comparison. Here a brief discussion of how the system may theoretically be impacted by various impairments as well as potential mitigation techniques is provided.

The proposed AMC mechanism utilises amplitude and phase data to extract features from the constellation diagram. As Rayleigh fading causes both amplitude and phase distortion the likely implications would perhaps be more severe than Gaussian Noise, particularly at lower SNRs. However, it was demonstrated how the system has partial robustness to phase distortion so there is potentially the chance that the proposed system would perform well.

It was discussed in Section 4.3 how utilising the polar form provides partial immunity to CFO as the magnitude relationships are preserved. Should the magnitude of the CFO be constant there will be no effect upon the performance of the system as the value of the arguments is irrelevant to the clustering result, it is only when the CFO magnitude varies that performance would be diminished, such as offset induced by the Doppler effect. This is due to all constellations of a similar magnitude merging into rings around the origin, in this case the argument feature would become useless for classification. To combat this reduction in performance some form of CFO estimation and compensation would be required. As a side note, in a similar manner to how feature cluster positioning was found to be a strong method of estimating SNR, there may be a way to utilise the proposed system to estimate CFO in a similar manner.

Performance deterioration under phase noise will be affected similarly to CFO. Minimal phase noise may result in little to no accuracy loss, large phase noise values will cause a merging of argument clusters and therefore reduce accuracy. Modulation order also plays a role here, higher order modulation schemes will be more sensitive to phase noise as their constellations are more closely spaced. Smaller argument ε values may go some way towards mitigating accuracy losses but cannot counteract complete constellation overlap.

Robustness to timing synchronisation errors was demonstrated by works which utilised the RadioML datasets [16,41]. It was found that the proposed system's accuracy significantly deteriorated with such an impairment. The reason for this was that timing synchronisation errors create "ghost" constellation points at the transitions between expected clusters, these ghost points not only can increase the number of clusters found but can also provide bridging points between expected clusters thereby joining them together. Bridging may be mitigated by reducing ε values as fortunately DBSCAN is a clustering algorithm designed to be effective on noisy data. However, maximum system accuracy may only be obtained via symbol timing recovery.

These are some of the most common real-world channel impairments and how they may effect the performance of the proposed systems in terms of both AMC and SNR estimation accuracy. In the majority of cases it is likely that the introduction of these impairments would result in performance reductions, however, there is also potential for the system to be trained to recognise the presence of such impairments. For instance, the system could be created with varying ε values, and depending upon the values used differing numbers of

clusters will be obtained. These differing numbers could be related to impairment severities and therefore provide a means of channel estimation. So while the proposed system would theoretically be sensitive to such impairments, such sensitivity could prove valuable in the creation of an all-in-on CR system.

9.3.4 DBSNR Hardware

The ability of the DBSCAN based feature extraction system to estimate SNR was thoroughly demonstrated via software simulations. The first obvious task which would be advantageous to perform would be to implement the potential hardware implementation which was described in Section 8.1.

The SNR is one metric which describes the effects which the channel may have on a communications signal, it would be ideal for the proposed hardware to have the ability to perform an entire suite of channel estimation operations, the metrics which it would be advantageous to obtain may for example include a measure of the phase noise and doppler effect-imposed frequency shifts. Theoretically it may be possible to estimate the magnitude of the doppler effects as the effects result in CFO [10], which in turn results in an apparent rotation of the constellation diagram. It was stated that the proposed system has a degree of CFO immunity, but this is not the case if the imposed rotation is severe enough to be captured within a set of samples. In these cases, the number of argument clusters found would severely decrease while the number of magnitude clusters would remain the same, by measuring the decrease in argument clusters and comparing with the expected value a measure of the CFO and therefore frequency offset imposed by doppler effects could be obtained. Phase noise imposes a static rotation of the constellation, the mechanism of the DBSCAN feature extractor is blind to these effects as it extracts a measure of how many different arguments and magnitudes there are, rather than the values of these constellation diagram features. A method of determining the constellation positioning would be required to be added to the system to obtain this value which could then perhaps be utilised as a comparison with the expected positioning. If the system could be configured to provide measures of these impairments, then further steps would be taken to realise a DBSCAN based core which can perform a greater number of functions which are required for CR.

9.3.5 Addressing the Circular Dependency of the Proposed System

There is a circular dependency of the proposed DBMC and DBSNR systems. While it is the case that both AMC and SNR estimation can be performed with a single implementation, to perform either function requires accurate knowledge of the result of the other. For instance, to perform a modulation classification operation requires knowledge of the signal's SNR, to perform SNR estimation requires knowledge of the signal's modulation scheme. This is a critical flaw of the proposed system but can be mitigated in a number of ways.

Firstly, knowledge of one value allows the other to be obtained. If a signal is input and the modulation scheme is known them the SNR may then be obtained. The system could then alternate between SNR estimation and AMC functionality so it always has an up to date knowledge of the current transmission characteristics. A rapid change in the SNR and modulation scheme simultaneously would however cause all knowledge of signal characteristics to be lost. In this case the system could either request a short transmission of

the current modulation scheme so it can once again begin operation, or the transmitter could periodically transmit this information. This is not ideal as the system is designed to eliminate such requirements, although it is the case that the required frequency of such transmissions would be lower.

The second and perhaps most promising option is to develop a means of determining what the most likely modulation scheme and SNR combination is. The system could iterate between performing alternative AMC and SNR estimation operations and find the combination which is most likely. It is also possible to train a larger MLP on all modulation scheme data at all SNRs. In this case the classes would be for instance 4QAM at 25dB, 8PSK at 20dB, or 16APSK at 30dB. Over the course of several classification operations the most likely candidate could be identified. This implementation was not developed in this thesis due to the requirements for an extremely complex classification model which would naturally increase the implementation size of an already comparatively large implementation. The performance of such a system was investigated briefly in software, particularly on large datasets, it was found that there was significant feature cluster overlap which lead to poor performance. Nonetheless, this would be a mechanism of not necessarily always finding the exact modulation scheme/SNR combination but would provide a set of possible combinations. Figures 9.2 and 9.3 show the resulting feature space which a classification model would have to learn from. Figure 9.2 shows the feature space which includes all modulation schemes at all SNRs investigated in this work. There are certainly clusters with strong separation which would be classified correctly in 100% of cases, but on the right hand side of the plot there is a huge number of clusters which overlap, suggesting very poor performance. Figure 9.3 includes only 4PSK, 16PSK, 64APSK, and 256QAM, in this case there is still overlap at the right of the plot but it is less severe and in many cases clear clusters can be identified. When utilising only 4 modulation schemes, as with 5G, [1] the system could potentially not only solve the circular dependency but also provide the ability to perform SNR estimation and modulation classification within a single operation. The addition of a third feature, perhaps a high-order cumulant which were found to be highly discriminatory, may provide a means of distinguishing the clusters which overlap.

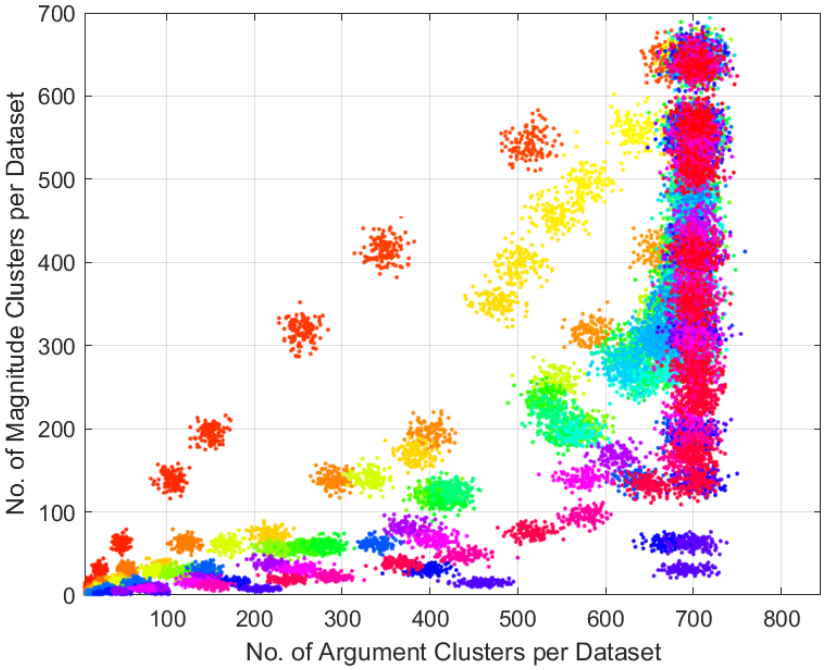


Figure 9.2: *Feature Space of All Modulation Schemes Investigated in this Thesis at SNRs from -10dB to 40dB*

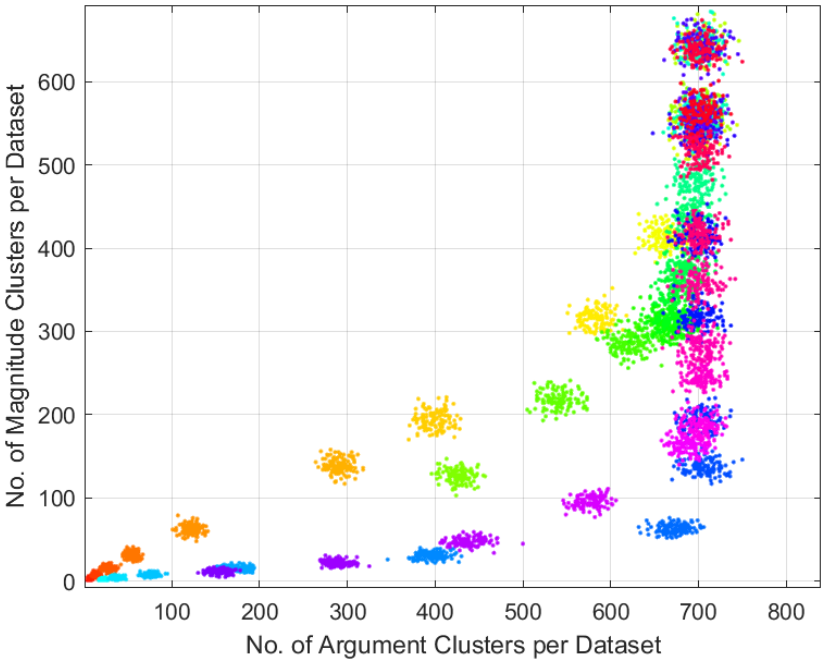


Figure 9.3: *Feature Space of 4PSK, 16PSK, 64APSK, and 256QAM at SNRs from -10dB to 40dB*

Bibliography

[1] "5G NR Overall description;." [Online]. Available: https://www.etsi.org/deliver/etsi$_ts$/138300$_1$38399/138300/15.03.01$_6$0/$ts_1$38300v150301$p.pdf$

[2] ETSI, "5G; NR; User Equipment (UE) radio transmission and reception; Part 1: Range 1 Standalone," Jan. 2023. [Online]. Available: https://www.etsi.org/deliver/etsi$_ts$/138100$_1$38199/13810101/17.08.00$_6$0/$ts_1$3810101v170800$p.pdf$

[3] M. Tesanovic and M. Nekovee, "mmWave-Based Mobile Access for 5G: Key Challenges and Projected Standards and Regulatory Roadmap," in *2015 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2015, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/7417194/?arnumber=7417194

[4] I. Rodriguez, H. C. Nguyen, T. B. Sorensen, J. Elling, J. A. Holm, P. Mogensen, and B. Vejlgaard, "Analysis of 38 GHz mmWave Propagation Characteristics of Urban Scenarios," in *Proceedings of European Wireless 2015; 21th European Wireless Conference*, May 2015, pp. 1–8. [Online]. Available: https://ieeexplore.ieee.org/document/7147721/?arnumber=7147721

[5] E. Harinda, S. Hosseinzadeh, H. Larijani, and R. M. Gibson, "Comparative Performance Analysis of Empirical Propagation Models for LoRaWAN 868MHz in an Urban Scenario," in *2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*. Limerick, Ireland: IEEE, Apr. 2019, pp. 154–159. [Online]. Available: https://ieeexplore.ieee.org/document/8767245/

[6] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y.-J. A. Zhang, "The Roadmap to 6G: AI Empowered Wireless Networks," *IEEE Communications Magazine*, vol. 57, no. 8, pp. 84–90, Aug. 2019, conference Name: IEEE Communications Magazine. [Online]. Available: https://ieeexplore.ieee.org/document/8808168/?arnumber=8808168

[7] Y. Wang, J. Wang, W. Zhang, Y. Zhan, S. Guo, Q. Zheng, and X. Wang, "A survey on deploying mobile deep learning applications: A systemic and technical perspective," *Digital Communications and Networks*, vol. 8, no. 1, pp. 1–17, Feb. 2022. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2352864821000298

[8] Y. Linn, "New structures for modulation classification and SNR estimation with applications to Cognitive Radio and Software Defined Radio," in *2014 IEEE 27th Canadian Conference on Electrical and Computer Engineering (CCECE)*, May 2014, pp. 1–8, iSSN: 0840-7789. [Online]. Available: https://ieeexplore.ieee.org/document/6900915/?arnumber=6900915

[9] S. Haykin, "Cognitive radio: brain-empowered wireless communications," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 2, pp. 201–220, Feb. 2005. [Online]. Available: http://ieeexplore.ieee.org/document/1391031/

[10] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge University Press, 2005.

[11] Z. Zhu and A. K. Nandi, "Artificial Intelligence Enabled Radio Signal Intelligence," in *Artificial Intelligence and Cybersecurity*, T. Sipola, T. Kokkonen, and M. Karjalainen, Eds. Cham: Springer International Publishing, 2023, pp. 247–278. [Online]. Available: https://link.springer.com/10.1007/978-3-031-15030-2$_1$1

[12] "What is 5G? – 5G Observatory." [Online]. Available: https://5gobservatory.eu/about/what-is-5g/

[13] P. Aswathylakshmi and R. K. Ganti, "Pilotless Uplink for Massive MIMO Systems," in *GLOBECOM 2023 - 2023 IEEE Global Communications Conference*, Dec. 2023, pp. 4205–4210, arXiv:2305.12431 [cs]. [Online]. Available: http://arxiv.org/abs/2305.12431

[14] N. Jindal and A. Lozano, "Optimum Pilot Overhead in Wireless Communication: A Unified Treatment of Continuous and Block-Fading Channels," in *2010 European Wireless Conference (EW)*, 2010, pp. 725–732, arXiv:0903.1379 [cs]. [Online]. Available: http://arxiv.org/abs/0903.1379

[15] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep Learning Models for Wireless Signal Classification With Distributed Low-Cost Spectrum Sensors," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 3, pp. 433–445, Sep. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8357902/

[16] T. J. O'Shea, T. Roy, and T. C. Clancy, "Over-the-Air Deep Learning Based Radio Signal Classification," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, Feb. 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8267032/

[17] S. Tridgell, D. Boland, P. H. Leong, R. Kastner, A. Khodamoradi, and Siddhartha, "Real-time Automatic Modulation Classification using RFSoC," in *2020 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*. New Orleans, LA, USA: IEEE, May 2020, pp. 82–89. [Online]. Available: https://ieeexplore.ieee.org/document/9150443/

[18] J. Woo, K. Jung, and S. Mukhopadhyay, "Efficient Hardware Design of DNN for RF Signal Modulation Recognition Employing Ternary Weights," *IEEE Access*, vol. 12, pp. 80 165–80 175, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10547036/

[19] A. Kumar, M. S. Chaudhari, and S. Majhi, "Automatic Modulation Classification for OFDM Systems Using Bi-Stream and Attention-Based CNN-LSTM Model," *IEEE Communications Letters*, vol. 28, no. 3, pp. 552–556, Mar. 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10380547/

[20] C. Cardoso, A. R. Castro, and A. Klautau, "An Efficient FPGA IP Core for Automatic Modulation Classification," *IEEE Embedded Systems Letters*, vol. 5, no. 3, pp. 42–45, Sep. 2013. [Online]. Available: https://ieeexplore.ieee.org/document/6568874/

[21] S. Zhang, Z. Yue, Y. Liu, J. Song, P. Liu, and X. Zhao, "Design for a Communication Signal Modulation Pattern Recognition System Based on AP-SoC," in *2023 8th International Conference on Computer and Communication Systems (ICCCS)*. Guangzhou, China: IEEE, Apr. 2023, pp. 97–102. [Online]. Available: https://ieeexplore.ieee.org/document/10150727/

[22] D. Pauluzzi and N. Beaulieu, "A comparison of SNR estimation techniques in the AWGN channel," in *IEEE Pacific Rim Conference on Communications, Computers, and Signal Processing. Proceedings*. Victoria, BC, Canada: IEEE, 1995, pp. 36–39. [Online]. Available: http://ieeexplore.ieee.org/document/519404/

[23] S. Chen, S. Zheng, Z. Yang, T. Chen, Z. Zhao, and X. Yang, "Deep Learning-Based SNR Estimation with Covariance Input," in *2023 IEEE 23rd International Conference on Communication Technology (ICCT)*. Wuxi, China: IEEE, Oct. 2023, pp. 181–187. [Online]. Available: https://ieeexplore.ieee.org/document/10419442/

[24] S. Zheng, S. Chen, T. Chen, Z. Yang, Z. Zhao, and X. Yang, "Deep Learning-Based SNR Estimation," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 4778–4796, 2024. [Online]. Available: https://ieeexplore.ieee.org/document/10620237/

[25] N. Beaulieu, A. Toms, and D. Pauluzzi, "Comparison of four SNR estimators for QPSK modulations," *IEEE Communications Letters*, vol. 4, no. 2, pp. 43–45, Feb. 2000. [Online]. Available: http://ieeexplore.ieee.org/document/824751/

[26] Zhenyong Zuo and Kai Liu, "An envelop-based SNR estimation algorithm for QAM signals," in *2009 4th IEEE Conference on Industrial Electronics and Applications*. Xian, China: IEEE, May 2009, pp. 3868–3871. [Online]. Available: http://ieeexplore.ieee.org/document/5138931/

[27] Hua Xu, Zupeng Li, and Hui Zheng, "A non-data-aided SNR estimation algorithm for QAM signals," in *2004 International Conference on Communications, Circuits and Systems (IEEE Cat. No.04EX914)*. Chengdu, China: IEEE, 2004, pp. 999–1003. [Online]. Available: http://ieeexplore.ieee.org/document/1346347/

[28] M. Alvarez-Diaz, R. Lopez-Valcarce, and C. Mosquera, "SNR Estimation for Multilevel Constellations Using Higher-Order Moments," *IEEE Transactions on Signal Processing*, vol. 58, no. 3, pp. 1515–1526, Mar. 2010, conference Name: IEEE Transactions on Signal Processing. [Online]. Available: https://ieeexplore.ieee.org/document/5313956/?arnumber=5313956

[29] X. Xie, S. Peng, and X. Yang, "Deep Learning-Based Signal-To-Noise Ratio Estimation Using Constellation Diagrams," *Mobile Information Systems*, vol. 2020, pp. 1–9, Nov. 2020. [Online]. Available: https://www.hindawi.com/journals/misy/2020/8840340/

[30] K. Yang, Z. Huang, X. Wang, and F. Wang, "An SNR Estimation Technique Based on Deep Learning," *Electronics*, vol. 8, no. 10, p. 1139, Oct. 2019. [Online]. Available: https://www.mdpi.com/2079-9292/8/10/1139

[31] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, "Google's Neural

Machine Translation System: Bridging the Gap between Human and Machine Translation," Oct. 2016, arXiv:1609.08144 [cs]. [Online]. Available: http://arxiv.org/abs/1609.08144

[32] "How Google Tensor Helps Pixel Phones Do More." [Online]. Available: https://store.google.com/intl/en$_u k/ideas/articles/google-tensor-pixel-smartphone/$

[33] OpenAI, J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, R. Avila, I. Babuschkin, S. Balaji, V. Balcom, P. Baltescu, H. Bao, M. Bavarian, J. Belgum, I. Bello, J. Berdine, G. Bernadett-Shapiro, C. Berner, L. Bogdonoff, O. Boiko, M. Boyd, A.-L. Brakman, G. Brockman, T. Brooks, M. Brundage, K. Button, T. Cai, R. Campbell, A. Cann, B. Carey, C. Carlson, R. Carmichael, B. Chan, C. Chang, F. Chantzis, D. Chen, S. Chen, R. Chen, J. Chen, M. Chen, B. Chess, C. Cho, C. Chu, H. W. Chung, D. Cummings, J. Currier, Y. Dai, C. Decareaux, T. Degry, N. Deutsch, D. Deville, A. Dhar, D. Dohan, S. Dowling, S. Dunning, A. Ecoffet, A. Eleti, T. Eloundou, D. Farhi, L. Fedus, N. Felix, S. P. Fishman, J. Forte, I. Fulford, L. Gao, E. Georges, C. Gibson, V. Goel, T. Gogineni, G. Goh, R. Gontijo-Lopes, J. Gordon, M. Grafstein, S. Gray, R. Greene, J. Gross, S. S. Gu, Y. Guo, C. Hallacy, J. Han, J. Harris, Y. He, M. Heaton, J. Heidecke, C. Hesse, A. Hickey, W. Hickey, P. Hoeschele, B. Houghton, K. Hsu, S. Hu, X. Hu, J. Huizinga, S. Jain, S. Jain, J. Jang, A. Jiang, R. Jiang, H. Jin, D. Jin, S. Jomoto, B. Jonn, H. Jun, T. Kaftan, Kaiser, A. Kamali, I. Kanitscheider, N. S. Keskar, T. Khan, L. Kilpatrick, J. W. Kim, C. Kim, Y. Kim, J. H. Kirchner, J. Kiros, M. Knight, D. Kokotajlo, Kondraciuk, A. Kondrich, A. Konstantinidis, K. Kosic, G. Krueger, V. Kuo, M. Lampe, I. Lan, T. Lee, J. Leike, J. Leung, D. Levy, C. M. Li, R. Lim, M. Lin, S. Lin, M. Litwin, T. Lopez, R. Lowe, P. Lue, A. Makanju, K. Malfacini, S. Manning, T. Markov, Y. Markovski, B. Martin, K. Mayer, A. Mayne, B. McGrew, S. M. McKinney, C. McLeavey, P. McMillan, J. McNeil, D. Medina, A. Mehta, J. Menick, L. Metz, A. Mishchenko, P. Mishkin, V. Monaco, E. Morikawa, D. Mossing, T. Mu, M. Murati, O. Murk, D. Mély, A. Nair, R. Nakano, R. Nayak, A. Neelakantan, R. Ngo, H. Noh, L. Ouyang, C. O'Keefe, J. Pachocki, A. Paino, J. Palermo, A. Pantuliano, G. Parascandolo, J. Parish, E. Parparita, A. Passos, M. Pavlov, A. Peng, A. Perelman, F. d. A. B. Peres, M. Petrov, H. P. d. O. Pinto, Michael, Pokorny, M. Pokrass, V. H. Pong, T. Powell, A. Power, B. Power, E. Proehl, R. Puri, A. Radford, J. Rae, A. Ramesh, C. Raymond, F. Real, K. Rimbach, C. Ross, B. Rotsted, H. Roussez, N. Ryder, M. Saltarelli, T. Sanders, S. Santurkar, G. Sastry, H. Schmidt, D. Schnurr, J. Schulman, D. Selsam, K. Sheppard, T. Sherbakov, J. Shieh, S. Shoker, P. Shyam, S. Sidor, E. Sigler, M. Simens, J. Sitkin, K. Slama, I. Sohl, B. Sokolowsky, Y. Song, N. Staudacher, F. P. Such, N. Summers, I. Sutskever, J. Tang, N. Tezak, M. B. Thompson, P. Tillet, A. Tootoonchian, E. Tseng, P. Tuggle, N. Turley, J. Tworek, J. F. C. Uribe, A. Vallone, A. Vijayvergiya, C. Voss, C. Wainwright, J. J. Wang, A. Wang, B. Wang, J. Ward, J. Wei, C. J. Weinmann, A. Welihinda, P. Welinder, J. Weng, L. Weng, M. Wiethoff, D. Willner, C. Winter, S. Wolrich, H. Wong, L. Workman, S. Wu, J. Wu, M. Wu, K. Xiao, T. Xu, S. Yoo, K. Yu, Q. Yuan, W. Zaremba, R. Zellers, C. Zhang, M. Zhang, S. Zhao, T. Zheng, J. Zhuang, W. Zhuk, and B. Zoph, "GPT-4 Technical Report," Mar. 2024, arXiv:2303.08774 [cs]. [Online]. Available: http://arxiv.org/abs/2303.08774

[34] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," in *Proceedings of the Second International*

Conference on Knowledge Discovery and Data Mining (KDD-96). AAAI Press, 1996, pp. 226–231. [Online]. Available: https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

[35] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN," *ACM Transactions on Database Systems*, vol. 42, no. 3, pp. 1–21, Sep. 2017. [Online]. Available: https://dl.acm.org/doi/10.1145/3068335

[36] J. P. Mouton, M. Ferreira, and A. S. Helberg, "A comparison of clustering algorithms for automatic modulation classification," *Expert Systems with Applications*, vol. 151, p. 113317, Aug. 2020. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0957417420301421

[37] H. Zhang, P. Liu, Y. Guo, L. Zhang, and D. Huang, "Blind modulation format identification using the DBSCAN algorithm for continuous-variable quantum key distribution," *Journal of the Optical Society of America B*, vol. 36, no. 3, p. B51, Mar. 2019. [Online]. Available: https://opg.optica.org/abstract.cfm?URI=josab-36-3-B51

[38] Z. Zhao, A. Yang, P. Guo, and Q. Tan, "A Density Clustering Algorithm for Simultaneous Modulation Format Identification and OSNR Estimation," *Applied Sciences*, vol. 10, no. 3, p. 1095, Feb. 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/3/1095

[39] S. Kumar, R. Mahapatra, and A. Singh, "Automatic Modulation Recognition: An FPGA Implementation," *IEEE Communications Letters*, vol. 26, no. 9, pp. 2062–2066, Sep. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9804849/

[40] J. G. Proakis and M. Salehi, *Digital Communications*, 5th ed. McGraw Hill, 2007.

[41] T. O'Shea, "RadioML.2016.10A," 2016. [Online]. Available: https://www.deepsig.ai/datasets/

[42] W. Gardner and C. Spooner, "Cyclic spectral analysis for signal detection and modulation recognition," in *MILCOM 88, 21st Century Military Communications - What's Possible?'. Conference record. Military Communications Conference*, Oct. 1988, pp. 419–424 vol.2. [Online]. Available: https://ieeexplore.ieee.org/document/13425/?arnumber=13425

[43] T. V. R. O. Câmara, A. D. L. Lima, B. M. M. Lima, A. I. R. Fontes, A. D. M. Martins, and L. F. Q. Silveira, "Automatic Modulation Classification Architectures Based on Cyclostationary Features in Impulsive Environments," *IEEE Access*, vol. 7, pp. 138 512–138 527, 2019, conference Name: IEEE Access. [Online]. Available: https://ieeexplore.ieee.org/document/8846691/?arnumber=8846691

[44] K. Assaleh, K. Farrell, and R. Mammone, "A new method of modulation classification for digitally modulated signals," in *MILCOM 92 Conference Record*. San Diego, CA, USA: IEEE, 1992, pp. 712–716. [Online]. Available: http://ieeexplore.ieee.org/document/244137/

[45] A. Nandi and E. Azzouz, "Algorithms for automatic modulation recognition of communication signals," *IEEE Transactions on Communications*, vol. 46, no. 4, pp. 431–436, Apr. 1998. [Online]. Available: http://ieeexplore.ieee.org/document/664294/

[46] D. Saharia, M. R. Boruah, N. K. Pathak, and N. Sarma, "An Ensemble based Modulation Recognition using Feature Extraction," in *2021 International Conference on Intelligent Technologies (CONIT)*. Hubli, India: IEEE, Jun. 2021, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/9498547/

[47] A. Swami and B. Sadler, "Hierarchical digital modulation classification using cumulants," *IEEE Transactions on Communications*, vol. 48, no. 3, pp. 416–429, Mar. 2000. [Online]. Available: http://ieeexplore.ieee.org/document/837045/

[48] S. Zhou, T. Li, and Y. Li, "Recursive Feature Elimination Based Feature Selection in Modulation Classification for MIMO Systems," *Chinese Journal of Electronics*, vol. 32, no. 4, pp. 785–792, Jul. 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10168807/

[49] M. L. D. Wong, S. K. Ting, and A. K. Nandi, "Naïve Bayes classification of adaptive broadband wireless modulation schemes with higher order cumulants," in *2008 2nd International Conference on Signal Processing and Communication Systems*. Gold Coast, Australia: IEEE, Dec. 2008, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/document/4813755/

[50] A. Alarabi and O. A. S. Alkishriwo, "Modulation Classification Based on Statistical Features and Artificial Neural Network," in *2021 IEEE 1st International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering MI-STA*. Tripoli, Libya: IEEE, May 2021, pp. 748–751. [Online]. Available: https://ieeexplore.ieee.org/document/9464363/

[51] L. Wang and Y. Li, "Constellation based signal modulation recognition for MQAM," in *2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN)*. Guangzhou: IEEE, May 2017, pp. 826–829. [Online]. Available: http://ieeexplore.ieee.org/document/8230227/

[52] C. M. Bishop, *Pattern Recognition and Machine Learning*, ser. Information Science and Statistics. New York: Springer, 2006.

[53] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., ser. Springer Series in Statistics. New York: Springer, 2009.

[54] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[55] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning with Applications in R*. New York: Springer, 2013.

[56] M. Kuhn and K. Johnson, *Applied Predictive Modeling*. New York: Springer, 2013.

[57] V.-S. Doan, T. Huynh-The, C.-H. Hua, Q.-V. Pham, and D.-S. Kim, "Learning Constellation Map with Deep CNN for Accurate Modulation Recognition," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*. Taipei, Taiwan: IEEE, Dec. 2020, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/9348129/

[58] G. Huang, Y. Li, Q. Zhu, and C. He, "Modulation Classification of MQAM Signals Based on Gradient Color Constellation and Deep Learning," in *2021 International Wireless Communications and Mobile Computing (IWCMC)*. Harbin City, China: IEEE, Jun. 2021, pp. 1309–1313. [Online]. Available: https://ieeexplore.ieee.org/document/9498864/

[59] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems*, vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html

[60] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015, arXiv:1512.03385 [cs]. [Online]. Available: http://arxiv.org/abs/1512.03385

[61] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," Mar. 2015, arXiv:1502.03167 [cs]. [Online]. Available: http://arxiv.org/abs/1502.03167

[62] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting."

[63] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," Aug. 2023, arXiv:1706.03762 [cs]. [Online]. Available: http://arxiv.org/abs/1706.03762

[64] Google AI, "Introducing Gemini: Google's Next-Generation AI Model," 2023. [Online]. Available: https://ai.googleblog.com/2023/12/introducing-gemini-models.html

[65] S. Hamidi-Rad and S. Jain, "MCformer: A Transformer Based Deep Neural Network for Automatic Modulation Classification," in *2021 IEEE Global Communications Conference (GLOBECOM)*. Madrid, Spain: IEEE, Dec. 2021, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/9685815/

[66] J. Alammar, "The Illustrated Transformer," 2018. [Online]. Available: http://jalammar.github.io/illustrated-transformer/

[67] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in *Proceedings of NAACL-HLT*, 2019.

[68] O. Kaziha and T. Bonny, "A Comparison of Quantized Convolutional and LSTM Recurrent Neural Network Models Using MNIST," in *2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA)*, Nov. 2019, pp. 1–5. [Online]. Available: https://ieeexplore.ieee.org/document/8959793/?arnumber=8959793

[69] Z. Zhang, H. Luo, C. Wang, C. Gan, and Y. Xiang, "Automatic Modulation Classification Using CNN-LSTM Based Dual-Stream Structure," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 11, pp. 13 521–13 531, Nov. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9220797/

[70] S. Kumar, A. Singh, and R. Mahapatra, "Hardware Implementation of Automatic Modulation Classification with Deep Learning," in *2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*. Goa, India: IEEE, Dec. 2019, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/9118057/

[71] Y. Kumar, M. Sheoran, G. Jajoo, and S. K. Yadav, "Automatic Modulation Classification Based on Constellation Density Using Deep Learning," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1275–1278, Jun. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9037117/

[72] Y. Wang, M. Liu, J. Yang, and G. Gui, "Data-Driven Deep Learning for Automatic Modulation Recognition in Cognitive Radios," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 4074–4077, Apr. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8645696/

[73] A. P. Hermawan, R. R. Ginanjar, D.-S. Kim, and J.-M. Lee, "CNN-Based Automatic Modulation Classification for Beyond 5G Communications," *IEEE Communications Letters*, vol. 24, no. 5, pp. 1038–1041, May 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8977561/

[74] Z. Ke and H. Vikalo, "Real-Time Radio Technology and Modulation Classification via an LSTM Auto-Encoder," *IEEE Transactions on Wireless Communications*, vol. 21, no. 1, pp. 370–382, Jan. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9487492/

[75] X. Liu, D. Yang, and A. E. Gamal, "Deep neural network architectures for modulation classification," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*. Pacific Grove, CA, USA: IEEE, Oct. 2017, pp. 915–919. [Online]. Available: http://ieeexplore.ieee.org/document/8335483/

[76] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. Alemi, "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning," in *Thirty-first AAAI Conference on Artificial Intelligence*, 2017, pp. 4278–4284.

[77] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International conference on learning representations*, 2015.

[78] A. Howard, M. Sandler, G. Chu, L.-C. Chen, B. Chen, M. Tan, W. Wang, Y. Zhu, R. Pang, V. Vasudevan, and others, "Searching for mobilenetv3," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1314–1324.

[79] S. Tridgell, D. Boland, P. H. Leong, and S. Siddhartha, "Real-Time Automatic Modulation Classification," in *2019 International Conference on Field-Programmable Technology (ICFPT)*. Tianjin, China: IEEE, Dec. 2019, pp. 299–302. [Online]. Available: https://ieeexplore.ieee.org/document/8977889/

[80] T. Kim, D. Ahn, D. Lee, and J.-J. Kim, "V-LSTM: An Efficient LSTM Accelerator Using Fixed Nonzero-Ratio Viterbi-Based Pruning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, no. 10, pp. 3327–3337, Oct. 2023, conference Name: IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. [Online]. Available: https://ieeexplore.ieee.org/document/10041123/?arnumber=10041123

[81] AMD Xilinx, "7 Series FPGAs Overview Data Sheet," 2020. [Online]. Available: https://docs.amd.com/v/u/en-US/ds180_7Series_Overview

[82] C. E. Gilchriest, "Signal-to-noise Monitoring," *Supporting Research and Advanced Development*, vol. Space Programs Summary Vol IV, Jun. 1964.

[83] T. Benedict and T. Soong, "The joint estimation of signal and noise from the sum envelope," *IEEE Transactions on Information Theory*, vol. 13, no. 3, pp. 447–454, Jul. 1967. [Online]. Available: http://ieeexplore.ieee.org/document/1054037/

[84] B. Shah and S. Hinedi, "The split symbol moments SNR estimator in narrow-band channels," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 26, no. 5, pp. 737–747, Sep. 1990. [Online]. Available: http://ieeexplore.ieee.org/document/102709/

[85] C. M. Thomas, "MAXIMUM LIKELIHOOD ESTIMATION OF SIGNAL-TO-NOISE RATIO," PhD, University of Southern California, 1967. [Online]. Available: https://www.proquest.com/docview/288063016/fulltextPDF/C93D085513634180PQ/ 1?accountid=13828&sourcetype=Dissertations%20&%20Theses

[86] W. Wang, Y. Shen, and Y. Wang, "Low-Complexity Non-Data-Aided SNR Estimation for Multilevel Constellations," *IEEE Communications Letters*, vol. 24, no. 1, pp. 113–116, Jan. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/8907856/

[87] N. V. Smirnov, "Table for estimating the goodness of fit of empirical distributions," *The Annals of Mathematical Statistics*, vol. 19, no. 2, pp. 279–281, 1948, publisher: Institute of Mathematical Statistics.

[88] J. E. Volder, "The CORDIC Trigonometric Computing Technique," *IRE Transactions on Electronic Computers*, vol. EC-8, no. 3, pp. 330–334, Sep. 1959, conference Name: IRE Transactions on Electronic Computers. [Online]. Available: https://ieeexplore.ieee.org/document/5222693/?arnumber=5222693

[89] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. MIT Press, 2009.

[90] S. Haykin and M. Moher, *Digital Communication Systems*. Wiley, 2009.

[91] T. S. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Prentice Hall, 2002.

[92] R. Andraka, "A survey of CORDIC algorithms for FPGA based computers," in *Proceedings of the 1998 ACM/SIGDA sixth international symposium on Field programmable gate arrays - FPGA '98*. Monterey, California, United States: ACM Press, 1998, pp. 191–200. [Online]. Available: http://portal.acm.org/citation.cfm?doid=275107.275139

[93] "CORDIC part two: rectangular to polar conversion." [Online]. Available: https://zipcpu.com/dsp/2017/09/01/topolar.html

[94] The MathWorks, Inc., "MATLAB R2021b," Natick, Massachusetts, United States, 2021. [Online]. Available: https://www.mathworks.com/products/matlab.html

[95] B. Van Essen, C. Macaraeg, M. Gokhale, and R. Prenger, "Accelerating a Random Forest Classifier: Multi-Core, GP-GPU, or FPGA?" in *2012 IEEE 20th International Symposium on Field-Programmable Custom Computing Machines*, 2012, pp. 232–239.

[96] C. Kardaris, C. Kachris, and D. Soudris, "A high-performance FPGA architecture for Acceleration of SVM Machine Learning Training," in *2022 Panhellenic Conference on Electronics & Telecommunications (PACET)*, 2022, pp. 1–6.

[97] D. Anguita, A. Ghio, S. Pischiutta, and S. Ridella, "A hardware-friendly support vector machine for embedded automotive applications," in *2007 International Joint Conference on Neural Networks*. IEEE, 2007, pp. 1360–1364.

[98] A. W. Savich, M. Moussa, and S. Areibi, "The impact of arithmetic representation on implementing MLP-BP on FPGAs: A study," in *2007 International Conference on Field-Programmable Technology*. IEEE, 2007, pp. 221–228.

[99] C. Latino, M. A. Moreno-Armendariz, and M. Hagan, "Realizing general MLP networks with minimal FPGA resources," in *2009 International Joint Conference on Neural Networks*, 2009, pp. 1722–1729.

[100] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," Dec. 2019, arXiv:1912.01703 [cs]. [Online]. Available: http://arxiv.org/abs/1912.01703

[101] PyTorch Documentation, "torch.nn.Linear — PyTorch documentation," 2025. [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.Linear.html

[102] K. Levenberg, "A Method for the Solution of Certain Non-Linear Problems in Least Squares," *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944, publisher: American Mathematical Society.

[103] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," Jan. 2017, arXiv:1412.6980 [cs]. [Online]. Available: http://arxiv.org/abs/1412.6980

[104] J. Nocedal and S. J. Wright, *Numerical optimization*, 2nd ed. New York: Springer, 2006.

[105] Xilinx Inc., *Vivado Design Suite User Guide*, 2021, version Number: 2021.2. [Online]. Available: https://www.xilinx.com/support/documentation/sw$_m$*anuals*/*xilinx*2021$_2$/*ug*901 − *vivado* − *synthesis.pdf*

[106] IEEE, "IEEE Standard for Verilog Hardware Description Language," Institute of Electrical and Electronics Engineers, New York, NY, USA, Tech. Rep. IEEE Std 1364-2005, 2006.

[107] Avnet, *ZedBoard Hardware User's Guide*. Avnet Inc., 2014, version Number: 2.2. [Online]. Available: https://digilent.com/reference/programmable-logic/zedboard/reference-manual

[108] XILINX, "Zynq-7000 SoC Data Sheet: Overview, DS190 (v1.11.1)," Jun. 2018. [Online]. Available: https://www.mouser.com/datasheet/2/903/ds190-Zynq-7000-Overview-1595492.pdf

[109] F. Kaltenberger, "5G New Radio in OpenAirInterface," in *Proceedings of the 5th OAI Workshop*. OpenAirInterface, Jun. 2018. [Online]. Available: https://www.openairinterface.org/docs/workshop/5_OAI_Workshop_20180620/KALTENBERGER_5G%20New%20Radio%20in%20OpenAirInterface.pdf

[110] F.-L. Luo and C. Zhang, *Digital Signal Processing for 5G: Algorithms and Implementations*, ser. Wiley - IEEE. Hoboken, NJ: Wiley-IEEE Press, 2021.

[111] R. Mueller, J. Teubner, and G. Alonso, "Sorting networks on FPGAs," *VLDB J.*, vol. 21, pp. 1–23, Feb. 2012.

[112] D. Koch and J. Torresen, "FPGASort: a high performance sorting architecture exploiting run-time reconfiguration on fpgas for large problem sorting," in *Proceedings of the 19th ACM/SIGDA International Symposium on Field Programmable Gate Arrays*, ser. FPGA '11. New York, NY, USA: Association for Computing Machinery, 2011, pp. 45–54, event-place: Monterey, CA, USA. [Online]. Available: https://doi.org/10.1145/1950413.1950427

[113] N. Scicluna and C.-S. Bouganis, "ARC 2014: A Multidimensional FPGA-Based Parallel DBSCAN Architecture," *ACM Transactions on Reconfigurable Technology and Systems*, vol. 9, no. 1, pp. 1–15, Nov. 2015. [Online]. Available: https://dl.acm.org/doi/10.1145/2724722

[114] S. Shi, Q. Yue, and Q. Wang, "FPGA based accelerator for parallel DBSCAN algorithm," 2014.

[115] Xilinx Inc., "PetaLinux Tools Documentation Reference Guide," Xilinx, User Guide UG1144, Oct. 2022, version Number: v2022.2. [Online]. Available: https://www.xilinx.com/support/documents/sw$_m anuals/xilinx2022_2/ug1144-petalinux-tools-reference-guide.pdf$

[116] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu, "Density-Based Clustering in Spatial Databases: The Algorithm GDBSCAN and Its Applications," *Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 169–194, 1998. [Online]. Available: http://link.springer.com/10.1023/A:1009745219419

[117] Rohde & Schwarz, *R&S\textsuperscript®SMB100A RF Signal Generator Specifications*. Munich, Germany: Rohde & Schwarz GmbH & Co. KG, 2020, version Number: 03.00. [Online]. Available: https://scdn.rohde-schwarz.com/ur/pws/dl_downloads/pdm/cl_brochures_and_datasheets/specifications/3609$_4$761_22/$SMB100A_specs_en_3609-4761-22_v0300.pdf$

[118] Keysight Technologies, *N9030B PXA Signal Analyzer Data Sheet*. Santa Rosa, CA: Keysight Technologies Inc., 2022, no. 5992-1317EN. [Online]. Available: https://www.keysight.com/us/en/product/N9030B/pxa-signal-analyzer-multi-touch-10-hz-to-50-ghz.html

[119] European Telecommunications Standards Institute, "Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; Part 2: DVB-S2 Extensions (DVB-S2X),"

ETSI, European Standard EN 302 307-2 V1.3.1, Feb. 2022. [Online]. Available: https://www.etsi.org/deliver/etsi$_e n$/302300$_3$02399/30230702/01.03.01$_6$0/$en_3$0230702v010301$p.pdf$

[120] P. Xue, A. Wagh, G. Ma, Y. Wang, Y. Yang, T. Liu, and C. Huang, "Integrating Deep Learning and Hydrodynamic Modeling to Improve the Great Lakes Forecast," *Remote Sensing*, vol. 14, p. 2640, May 2022.

[121] Phung and Rhee, "A High-Accuracy Model Average Ensemble of Convolutional Neural Networks for Classification of Cloud Image Patches on Small Datasets," *Applied Sciences*, vol. 9, p. 4500, Oct. 2019.

[122] S. Yan, "Understanding LSTM and its Diagrams," 2016, publisher: ML Review. [Online]. Available: https://blog.mlreview.com/understanding-lstm-and-its-diagrams-37e2f46f1714