DEEP LEARNING BASED AUTOMATIC MODULATION CLASSIFICATION IN THE PRESENCE OF CARRIER PHASE OFFSET AND CARRIER FREQUENCY OFFSET

by

Ramazan Yılmaz

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2018

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University

2022

ACKNOWLEDGEMENTS

First of all, I want to thank my supervisor Assoc. Prof. Ali Emre Pusane. The last two years have not been the best times, but whenever I was wandering off-track, he put me back in and motivated me again and again. All I know is I would not be able to finish this thesis without him and I am very grateful for what he has done for me.

I also want to thank my mother, father, and sister for their endless support throughout this journey. Covid-19 times have not been the best times for them either, but I am very proud that they stand so strong.

I also want to thank my friends. They help me to find joy and happiness in these difficult times, even when we could not see each other because of the pandemic.

ABSTRACT

DEEP LEARNING BASED AUTOMATIC MODULATION CLASSIFICATION IN THE PRESENCE OF CARRIER PHASE OFFSET AND CARRIER FREQUENCY OFFSET

Automatic Modulation Classification (AMC) has emerged after the efforts of making the modulation classification process autonomous. Since then, various methods, algorithms, and tools have been used in the AMC field, such as likelihood-based methods, the goodness of fit tests, feature-based methods, machine learning-based methods, and deep learning-based methods. With the help of these methods, the modulation classification operation can be performed automatically without any human input. In this thesis, we survey these methods in detail and propose our methods to contribute to the AMC field. First, we proposed a blind feature-based algorithm that uses K-nearest neighbor (KNN) to perform classification. When the number of symbols in each signal decreases, the classification process may encounter an error floor. The main goal of the proposed feature-based algorithm is to combat this error floor. Then, we proposed a novel polar coordinate approach in deep learning to classify the signals that are affected by carrier phase offset (CPO). The polar coordinate approach converts the rotational effect of CPO into the translational effect, which makes the classification easier. Finally, we propose a 2-staged deep learning-based classification algorithm under the presence of carrier frequency offset (CFO). In the first stage, the algorithm estimates the CFO amount and in the second stage, it classifies the CFO-affected signals. Finally, we conclude the thesis by discussing the future works and possible improvements.

ÖZET

TAŞIYICI FAZ KAYMASI VE TAŞIYICI FREKANS KAYMASI ALTINDA DERİN ÖĞRENME TEMELLİ OTOMATİK MODÜLASYON SINIFLANDIRMA

Otomatik modülasyon sınıflandırma (OMS), modülasyon sınıflandırma işlemlerini otonom bir hale getirme çabalarının sonucunda ortaya çıkmış bir alandır. OMS'nin ortaya çıkmasından bu yana birçok çok metot ve algoritma, OMS alanında kullanılmıştır. Bunları; olabilirlik temelli sınıflandırma, sınama temelli sınıflandırma, öznitelik temelli sınıflandırma, derin öğrenme temelli sınıflandırma ve makine öğrenmesi temelli sınıflandırma olarak ayırabiliriz. Bu metotlar yardımıyla modülasyon sınıflandırma işlemleri tamamen otomatik bir şekilde yapılabilir. Bu tezde, bu metotları yakından inceleyeceğiz ve OMS alanına katkıda bulunacak kendi algoritmalarımızı tanıtacağız. Önerdiğimiz algoritmalardan ilki K-en yakın komşu (KEYK) algoritması kullanan bir öznitelik temelli sınıflandırma. Her bir sinyaldeki sembol sayısı düşünüldüğünde, sınıflandırma yaparken bir hata tabanı ile karşılaşma ihtimali yüksektir. Bizim önerdiğimiz algoritmanın asıl amacı da bu durumla başa çıkmaktır. Önerdiğimiz bir diğer algoritma ise polar koordinat temelli bir derin öğrenme algoritması. Bu algoritma, taşıyıcı faz kayması (TFAK) etkisinde olan sinyalleri sınıflandırmayı hedeflemektedir. Polar koordinatlar ise dönme etkisini, doğrusal hareket etkisine çevirerek sınıflandırmayı kolaylaştırmaktadır. Önerdiğimiz son algoritma ise taşıyıcı frekans kayması (TFRK) yaşayan sinyalleri 2 aşamalı ve derin öğrenme temelli sınıflandırma algoritması. İlk aşamada; algoritma, TFRK kestirmesi yaparak frekans kayması etkisini azaltmayı amaçlar. İkinci aşamada ise bu sinyalleri sınıflandırmayı amaçlar. Son olarak, ileride yapılabilecek işleri ve gelişmeleri tartışarak tezi noktalıyoruz.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . ix
LIST OF TABLES . . . xi
LIST OF SYMBOLS . . . xii
LIST OF ACRONYMS/ABBREVIATIONS . . . xiv
1. INTRODUCTION . . . 1
1.1. Automatic Modulation Classification . . . 1
1.2. AMC Applications . . . 2
1.2.1. Military Applications . . . 2
1.2.2. Civil Applications . . . 3
1.3. Literature Review . . . 3
1.4. Thesis Contribution . . . 7
1.5. Thesis Organization . . . 8
2. AMC METHODS . . . 9
2.1. Signal Model . . . 9
2.2. Likelihood-Based Methods . . . 10
2.2.1. Maximum Likelihood-Based Classifier . . . 11
2.2.2. Average Likelihood Ratio Test . . . 12
2.2.3. Generalized Likelihood Ratio Test . . . 13
2.2.4. Hybrid Likelihood Ratio Test . . . 14
2.3. Feature-Based Methods . . . 15
2.3.1. Spectral-Based Features . . . 15
2.3.2. Wavelet Transform-Based Features . . . 19
2.3.3. Higher Order Statistics-Based Features . . . 20
2.4. Machine Learning-Based Methods . . . 23
2.4.1. K-Nearest Neighbor . . . 23

2.4.2. Support Vector Machine 26
2.5. Feature Reduction Algorithms 28
2.5.1. Logistic Regression 28
2.5.2. Artificial Neural Networks 29
2.5.3. Genetic Programming 31
2.6. Convolutional Neural Network-Based Classification 34
2.6.1. Convolutional Layers 35
2.6.2. Fully Connected Dense Layers 37
2.6.3. Pooling Layers 37
2.6.4. Batch Normalization 38
2.6.5. Dropout 39
2.6.6. Conclusion 39
3. FEATURE-BASED AMC AND PERFORMANCE EVALUATION 41
3.1. Cumulant-Based Classification 41
3.1.1. Results 43
3.1.2. Root of the Problems 44
3.2. How to Improve the Performance 45
3.2.1. Narrower Region of Interest 45
3.2.2. Separating High SNR Signals From Low SNR Signals 48
3.3. General Proposed Algorithm 49
3.4. Results 50
4. AMC IN THE PRESENCE OF CARRIER PHASE OFFSET 52
4.1. CPO Presence 52
4.1.1. Signal Model 52
4.1.2. Training Dataset 53
4.1.3. Training Dataset Generating Procedure 53
4.1.4. CNN Architecture 54
4.1.5. Results 56
4.2. Polar Coordinates and CPO 57
4.2.1. Methodology 58
4.2.2. Results 59

4.3. Global Average Pooling 60
4.3.1. Results 61
5. AMC UNDER THE PRESENCE OF CARRIER FREQUENCY OFFSET . 63
5.1. Signal Model 63
5.2. CFO Estimation 64
5.2.1. Training Dataset 64
5.2.2. CNN Architecture 65
5.2.3. Results 66
5.3. Classification 69
5.3.1. Training Dataset and CNN Architecture 69
5.3.2. Results 71
6. CONCLUSION 73
REFERENCES 75

LIST OF FIGURES

Figure 2.1. Comparison of the CPO effect. 10

Figure 2.2. Comparison of the CFO effect. 11

Figure 2.3. 2-D KNN visualization. 25

Figure 2.4. 2-D SVM visualization. 28

Figure 2.5. 3-Layer MLP Example. 30

Figure 2.6. Parent branches before cross-over operation. 32

Figure 2.7. Child branches after cross-over operation. 32

Figure 2.8. Mutation Operation. 33

Figure 2.9. 2-D convolution visualization. 35

Figure 2.10. AlexNet Architecture. 40

Figure 3.1. Constellation diagrams of the selected modulation types. 42

Figure 3.2. 64-QAM and QPSK symbol histograms. Number of total symbol is 2000. 45

Figure 3.3. Visualization of the narrower region. 46

Figure 3.4. Separability after CoV. 48

Figure 3.5. Block diagram of the proposed feature-based classifier. 50

Figure 4.1. Samples from the cartesian dataset. 55

Figure 4.2. CNN architecture for the classification of CPO affected signals with a max-pooling layer. 56

Figure 4.3. Samples from the Cartesian dataset. 59

Figure 4.4. CNN architecture for the classification of CPO affected signals with a global average pooling layer. 61

Figure 5.1. Samples from the differential dataset. 65

Figure 5.2. CNN architecture for the CFO estimation. 66

Figure 5.3. CFO Estimation comparison of BPSK in terms of SNR. The CFO is uniformly distributed between 2 and -2. 67

Figure 5.4. CFO Estimation comparison of QPSK in terms of SNR. The CFO is uniformly distributed between 2 and -2. 67

Figure 5.5. CFO Estimation comparison of 8-PSK in terms of SNR. The CFO is uniformly distributed between 2 and -2. 68

Figure 5.6. CFO Estimation of QPSK and 8-PSK when the input size is smaller. The CFO is uniformly distributed between 10 and -10. 68

Figure 5.7. Samples from the new polar dataset. 70

Figure 5.8. CNN architecture for the classification of CFO affected signals. . . 71

LIST OF TABLES

Table 2.1. Cumulant values of some popular modulation types. 23

Table 3.1. Confusion matrices of the KNN method. 43

Table 3.2. Confusion matrices of the narrower region method. 47

Table 3.3. Confusion matrices of the proposed method. 51

Table 4.1. Confusion matrices of Cartesian network with max-pooling layer. . 57

Table 4.2. Confusion matrices of the polar network with max-pooling layer. . 59

Table 4.3. Confusion matrices of the polar network with a global average pooling layer. 62

Table 5.1. Confusion matrices of the polar network with CFO affected test data. 72

LIST OF SYMBOLS

A	Channel gain
A_n	n^{th} symbol's amplitude
A_T	Threshold value for one sample point
C_m	m^{th} order cumulant
E	Error
$F(A)$	Signal A's feature set
$f[n]$	Instantaneous frequency vector
f_c	Carrier frequency
f_o	Carrier frequency offset
f_s	Sampling frequency
$g[n]$	AWGN component
$h[n]$	Residual channel effects
$L(\mathbf{r})$	Likelihood function
m_i	i^{th} modulaton type
N	Number of symbols in a signal
N_C	Number of samples that exceeds the threshold
S	Energy of the signal
T	Period
T_s	Sampling period
$\mathbf{W}$	Weight matrix
$\mathbf{w}$	Weight vector
w_c	Carrier frequency
w_n	n^{th} symbol's frequency
$\mathbf{w}_o$	Weight offset vector
w_o	Scalar weight offset
w_{ij}	Weight of the connection between two neurons
$x[n]$	Symbol vector
X_C	DTFT of $A[n]$

θ_0	Carrier phase offset
κ	Learning rate
μ	Expected value
μ_A	Mean of all points of $A[n]$
μ_i	i^{th} modulaton symbol
σ	Standard deviation
σ^2	Variance
Φ	Unknown parameters
ϕ_{NL}	Nonlinear component of instantaneous phase
ψ	Wavelet function
Ψ	Characteristic function

LIST OF ACRONYMS/ABBREVIATIONS

AD	Anderson-Darling
ALRT	Average Likelihood Ratio Test
AMC	Automatic Modulation Classification
ANN	Artificial Neural Networks
ASK	Amplitude Shift Keying
AWGN	Additive White Gaussian Noise
BPSK	Binary Phase Shift Keying
CFO	Carrier Frequency Offset
CNN	Convolutional Neural Network
CoV	Coefficient of Variance
CPO	Carrier Phase Offset
CSI	Channel State Information
CvM	Cramer-Von Mises
CWT	Continuous Wavelet Transform
DFT	Discrete Fourier Transform
DTFT	Discrete-Time Fourier Transform
FNN	Feed-Forward Neural Network
FSK	Frequency Shift Keying
GLRT	Generalized Likelihood Ratio Test
GP	Genetic Programming
HLRT	Hybrid Likelihood Ratio Test
KNN	K-Nearest Neighbor
KS	Kolmogorov-Smirnov
ML	Maximum Likelihood
MLP	Multi-Layer Perceptron
PDF	Probability Density Function
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation

QPSK	Quadrature Phase Shift Keying
RNN	Recurrent Neural Network
SNR	Signal to Noise Ratio
SVM	Support Vector Machine

1. INTRODUCTION

Since the day wireless communication entered our lives, modulations and modulation techniques have been topics of interest for communication engineers. Modulations make long-distance communication possible due to their ability to work in the high-frequency range. However, modulating a signal essentially means encoding a signal in a certain way. Therefore, the receiver needs to know the received signal's modulation technique to decode it, or demodulate it in this case, accordingly.

In one scenario, both the receiver and transmitter may agree on the set of standards, including modulation techniques, beforehand to enable cooperative communication. As a result, the receiver would know the modulation types which are necessary for the demodulation process. For some other cases, the receiver may want to classify a received signal due to various reasons, such as spectrum monitoring or intelligence gathering. Since the communication is not cooperative, the receiver would not know the modulation; therefore, some kind of modulation classification would be required to determine the modulation type.

In this chapter, we will discuss automatic modulation classification (AMC), its applications, and literature review of AMC.

1.1. Automatic Modulation Classification

At first, the modulation classification process was operated manually by an engineer by looking at the characteristics of the signal, such as its bandwidths, moments, phases, etc. However, due to the human factor, this process was slow and inefficient. The necessity of improving the efficiency resulted in a new method that is called automatic modulation classification.

AMC determines the received signal's modulation type, which is necessary to demodulate the given signal to recover the whole message as accurate as it can. The whole AMC operation is performed automatically by an AMC algorithm without any human input during the process.

1.2. AMC Applications

1.2.1. Military Applications

Similar to most of the technological developments in history; first, AMC was developed to be utilized in military applications. The main motivation for military scenarios is to interfere in the communication line. We can divide military applications into three main applications: gathering intelligence, attacking the communication line, and protecting the data. AMC can be employed to interfere in as a third party and demodulate the signal to recover the intelligence in the enemy's communication line. Gathering intelligence during wars proved to be important on many occasions throughout history.

AMC can also be used to attack the communication line along with jammers. Jammers are used to generate high power signals to disrupt the real signal in the transmission line and widely used in the world. However, the jammer needs to propagate a signal that has the same characteristic as the signal in the transmission line. Therefore, the jammer also needs to know the modulation type of the signal in order to imitate it. Otherwise, the receiver of the communication would be able to demodulate and decode the system easily with the help of a sequence of filters.

Lastly, an AMC system may be used to offer protection. If the receiver and transmitter manage to classify the modulation of jammers, they can adaptively change the modulation type to avoid jammer signals.

1.2.2. Civil Applications

Civil applications of AMC, on the other hand, focus on the performance. Spectrum monitoring, for instance, is one of the civil areas where AMC is used. In spectrum monitoring, relevant agencies control all the signals in the transmission line and their frequency bands in order to prevent any interference from outside sources. If any interference or bandwidth overlap occurs, relevant agencies could identify the unknown source by demodulating the signal. Therefore, AMC is required to find the modulation type of the unknown signal.

In some other scenarios, the transmitter may use a dynamic modulation scheme, where the system changes the modulation type according to the changing channel conditions. For instance; the transmitter may change the modulation type to more robust ones, such as BPSK or QPSK, when the channel becomes noisy. As the noise level decreases, the transmitter may prioritize the higher bit rate and use modulation types, such as 64-QAM or 256-QAM. To adapt to these changes, the receiver could perform AMC, If this information is not known at the receiver end.

As a result, these applications show that AMC is being utilized in many systems and developing and implementing a good AMC algorithm only enhances the performance of these systems further.

1.3. Literature Review

The entire AMC literature can be divided into two subcategories: blind and non-blind AMC. In non-blind AMC, the receiver is assumed to know the channel state information (CSI) or in other words, the receiver knows the characteristics of the communication channel. In blind AMC, on the other hand, the receiver does not know anything about the channel or the signal. Therefore, blind AMC is a more complicated problem.

The first and the most popular non-blind AMC method is likelihood-based classification. The first step to the main algorithm of the likelihood-based classifier is finding a likelihood function for each modulation type and signal sample. The second step is comparing the outputs of the different modulation likelihood functions to determine the modulation type. Depending on the known CSI, There are different likelihood-based classifiers in the literature.

When the CSI and all the other parameters, except the modulation type, are known perfectly, maximum likelihood (ML) classifiers are used. First, in [1], AMC is performed by using ML classifiers. In the following years, ML classifiers were also used in [2–4] with various ML classifier methods.

ML classifiers need perfect knowledge of CSI and it is not viable if one or more parameters are unknown. To overcome this problem, average likelihood ratio test (ALRT) is developed. ALRT is first used in the AMC area in [5]. Instead of writing down the known parameters, ALRT takes the integral of the unknown parameters over all possible values. Later, ALRT is also utilized in [6] to perform AMC. Since ALRT makes the problem significantly more complex because of the integrals, generalized likelihood ratio test (GLRT) is developed. GLRT employed first in [7] for AMC. Instead of taking integrals, GLRT treats the unknown parameters as deterministic unknown parameters and maximizes the likelihood function over all possible values.

Both ALRT and GLRT have their own problems. ALRT makes the problem more complex and its alternative, GLRT, performs a biased classification and it is not possible to use it for certain modulation types. Therefore, in [7], hybrid likelihood ratio test (HLRT) is also proposed for the AMC problem. HLRT is a hybrid form of ALRT and GRLT. Some parameters are treated as probabilistic parameters as if the classifier is ALRT and some other parameters are treated as deterministic parameters as if the classifier is GLRT.

Instead of using likelihood tests over analytical expressions, another method to perform AMC is performing tests over empirical distributions of the signals, assuming the signal is long enough.

One of those tests is Kolmogorov-Smirnov (KS) test, which is first proposed in [8] and explained in detail in [9]. KS test simply compares two distributions and scores how well they fit. In [10], this test is adapted to the problem of AMC. Later, in [11], the KS test for AMC is improved further. Computational efficiency alongside the accuracy is also important. Therefore, in [12], the complexity is reduced by lowering the number of comparison points and the performance is improved by choosing comparison points in key areas.

Cramer-Von Mises (CvM) test is another test that is used in the AMC problem. CvM test is first introduced in [13]. Similar to the KS test, CvM test also compares and scores the fit between two distributions. This test is implemented into the AMC problem in [14].

Anderson-Darling (AD) test, which is proposed in [15], counters the shortcomings of KS and CvM tests. Similar to the aforementioned tests, AD test also compares two distributions, but unlike others, the AD test gives more weight to the tails of the distribution. Therefore the AD test is more sensitive on the tails of the distributions. Since the AD test is the weighted version of the KS test, it can also be used to perform AMC.

Blind AMC techniques mostly involve machine learning tools. To use machine learning, some features are needed to be extracted from a signal. Fortunately, communication signals have lots of meaningful features that can be used in AMC. We can categorize these features under three sub-categories. First one is spectral-based features that are proposed in [16–18]. These are the features that are related to the signal's frequency, amplitude, and phase. The second one is Wavelet-transform-based features. These features are obtained through continuous wavelet transform as it is

proposed in [19] and later adopted in [20] as well. However, these features are not very successful in AMC when the modulation types are m-QAM and m-PSK. Finally, the last and the most popular features in AMC are high-order statistics-based features that are proposed first in [21] and later adopted and improved in [22, 23]. These features consist of the combination of high-order moments of the signal. One such special combination is called cumulants. Cumulants are proposed first in [24] and utilized in several ways in the following years, such as [25–27], since they make classification easier. After establishing the feature set, machine learning tools are very simple to use.

One of the popular machine learning tools for AMC is K-nearest neighbor (KNN). KNN is a supervised learning method, which means that the number of classes is determined beforehand and the classification is performed accordingly. Since the modulation set is selected before the AMC, KNN is suitable to be used in AMC. It was first utilized in the AMC area in [28]. In KNN, each feature is used as a dimension in the n-dimensional space, where n is the number of features in the feature set. Then, each signal sample is placed on this space and the signal sample is classified to the closest modulation point.

Support vector machines (SVM) can also be used as a classifier in the AMC problem. Although SVM is used as a multi-class classifier, it is generally used as a two-class classifier. Therefore, it is not utilized in the AMC area as much, but there are still studies that employ SVM as the modulation classifier, such as [29]. Essentially, SVM cuts the space half by a hyperplane and each side of the plane belongs to their respective class.

KNN and SVM are feature-hungry methods, as the number of features increases, the accuracy would also increase. However, that would increase the computational budget as well. Therefore, reducing the number of features without losing accuracy has been a topic of interest. As a result, several feature combination algorithms have been proposed in the AMC field.

Logistic regression is proposed in [30]. Logistic regression is used to map an n-dimensional feature set into a lower-dimensional one. The same can be performed in artificial neural networks (ANN) and its use in the AMC area is proposed in [31]. In the following years, it is employed in other studies, such as [32] as well. Genetic Programming can also be used for feature combination. Genetic programming as a feature combination algorithm is proposed in [33] and feature combination for AMC feature set is proposed in [34]. Later, the system is further improved in [26].

Finally, in recent years, deep learning algorithms started to be used in the AMC area. Due to the success of the traditional algorithms, such as ML classifiers or feature-based algorithms, the integration between deep learning algorithms and AMC happened later than the norm in the industry. After the deep learning tools started to be taken advantage of in the AMC area, many studies emerged in the literature, such as [35–38]. The common thing between these studies is that all of them have employed convolutional neural networks (CNN) as a deep learning tool. CNN is preferred in the AMC area because of its ability to recognize the patterns that modulated signals are known to have.

1.4. Thesis Contribution

In this thesis, we will propose three new methods that contribute to the AMC field:

- The first method is a feature-based classification method that uses the KNN algorithm. Feature-based KNN classification has been implemented in the literature before but we add two new stages to the classification process that combat the error floor problem. These new stages are called narrower region analysis and coefficient of variance analysis.
- The second method that we propose in this paper is a deep learning-based AMC approach under the presence of carrier phase offset. In this method, we use a novel polar coordinate approach to combat carrier phase offset. The polar coor-

dinate approach increases the efficiency of the network by reducing the amount of training data.

- The last method that we propose in this thesis is deep learning-based AMC under the presence of carrier frequency offset. In this method, we estimate the amount of frequency offset. Then, we use the polar coordinate approach that we proposed beforehand to classify the received signals.

1.5. Thesis Organization

In Chapter 2, we will discuss the signal model that will be used in the rest of the thesis and analysis of the methods that are used in AMC. In Chapter 3, we will discuss the thesis' contribution to the machine learning methods and the results of the proposed methods. In Chapters 4 and 5, we will discuss the thesis' contribution to the deep learning methods under the presence of carrier phase offset and carrier frequency offset respectively. After the deep learning algorithms are defined, the results will be shown in the end. Finally, we will conclude the thesis by summarizing the works in the thesis and discussing the future works and plans.

2. AMC METHODS

In this section, the signal model of the modulation signal will be defined and the AMC methods in the literature will be explained in detail.

2.1. Signal Model

A communication signal operates in the bandpass channel and most of the negative effects on the signal stem from here, excluding the other negative effects that are caused by the imperfections in the electronic systems. However, this thesis focuses on the bandpass channel effects. Therefore, only the bandpass channel effects are modeled on a signal. Additionally, for the sake of simplicity, baseband representation of the signal is used in this thesis to model channel effects, since these effects are also observed in the baseband region in the same way.

Baseband representation of a transmitted signal, including all the channel effects, is

$$r[n] = Ae^{j(2\pi f_o nT+\theta_o)} \sum_{l=\infty}^{l} x[l]h[nT - lT + \epsilon_T T] + g[n]. \tag{2.1}$$

Here, $x[l]$ is a signal vector, A is the channel gain, and h function is the residual channel effects. $g[n]$ is the additive white Gaussian noise (AWGN) component and AWGN affects every symbol independently. Since $g[n]$ is a complex number, the representation of $g[n]$ is

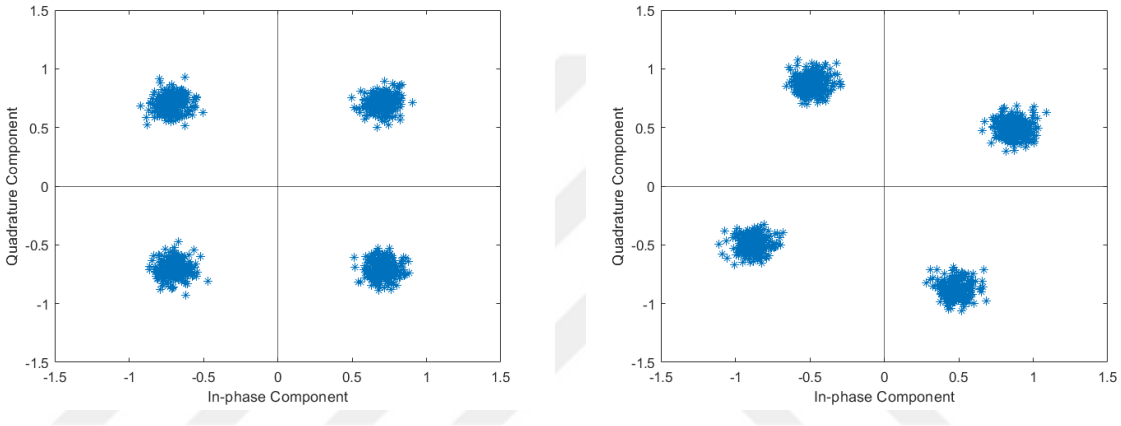
$$g[n] = R[n] + jI[n], \tag{2.2}$$

$$f_{I[n]}(x) = f_{R[n]}(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}}, \tag{2.3}$$

where $f_{I[n]}(x)$ and $f_{R[n]}(x)$ are the probability density functions (PDF) of $R[n]$ and

$I[n]$, respectively.

In Equation (2.1), θ_0 is the carrier phase offset (CPO). CPO causes all symbols to rotate around the origin by the CPO value. The effect of the CPO can be seen in Figure 2.1. Since all symbols rotate by the same amount, the characteristics of the signal are preserved.



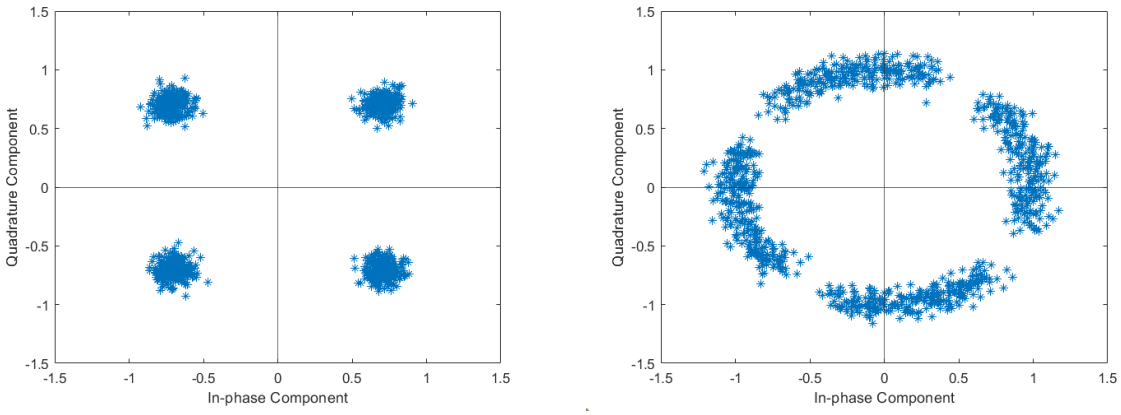
a) QPSK without CPO **b)** QPSK with CPO

Figure 2.1. Comparison of the CPO effect.

In Equation (2.1), f_0 is the carrier frequency offset (CFO). Under the effect of CFO, each symbol rotates around the origin by the CFO value with respect to the prior symbol. Effect of CFO can be seen in Figure 2.2. Since all symbols rotate by a different amount, the characteristics of the signal may not be preserved if the CFO is high enough.

2.2. Likelihood-Based Methods

Likelihood-based methods have been immensely popular in the AMC area; many researchers employed likelihood-based methods to solve the AMC problem in their works. Likelihood-based classifiers have 2 main steps to perform classification:



a) QPSK without CFO **b)** QPSK with CFO

Figure 2.2. Comparison of the CFO effect.

(i) By using the modulation schemes and signal samples, derive likelihood functions for each modulation type to represent them.

(ii) Compare likelihood functions with each other to perform classification.

2.2.1. Maximum Likelihood-Based Classifier

The most well-known classifiers among the likelihood-based classifiers are maximum likelihood (ML) classifiers. ML classifiers are used in the AMC literature, such as [1–4]. ML classifiers assume perfect channel knowledge. Therefore, every parameter is known to the receiver. ML classifier function can be interpreted as a PDF as it is proposed in [39]. The probability of a given symbol, $r_n = r_{I,n} + jr_{Q,n}$, to belong to a given modulation type, m_i, is

$$p_{\vec{r}}(r_{l,n}, r_{Q,n}|m = m_i) = \frac{1}{M(i)} \sum_{k=1}^{M(i)} \frac{1}{\sigma\sqrt{2\pi}} e^{\left(\frac{-(r_{I,n}-\mu_{I,k})^2-(r_{Q,n}-\mu_{Q,k})^2}{2\sigma^2}\right)}, \tag{2.4}$$

where $\mu_n = \mu_{I,n} + j\mu_{Q,n}$ are the modulation symbols of a modulation type and σ^2 is the variance of the AWGN channel that the received signal passed through. This PDF only gives a solution for one symbol. Since all the considered symbols are statistically independent, this calculation can be performed for all symbols, and, for a general

solution, all individual solutions are multiplied. As a result, the final solution becomes

$$p_{\overrightarrow{r}}(\mathbf{r}_I, \mathbf{r}_Q | m = m_i) = \prod_{n=1}^{N} p_{\overrightarrow{r}}(r_{l,n}, r_{Q,n} | m = m_i), \tag{2.5}$$

where N is the number of symbols in the symbol vector.

Finally, This process is repeated for all the modulation types in the modulation set and the classification is concluded according to the classification rule, which is

$$class(k) = arg\max_i \; p_{\overrightarrow{r}}(\mathbf{r}_I, \mathbf{r}_Q | m = m_i). \tag{2.6}$$

These classifiers are calculated considering the channel is AWGN. Additionally, these calculations still apply if the channel has fading or non-Gaussian noise.

2.2.2. Average Likelihood Ratio Test

ML classifiers assume complete knowledge of parameters and are not viable in many scenarios. As a result, new likelihood-based classifiers emerged. Average Likelihood Ratio Test (ALRT) is one of them. ALRT is first employed as a modulation classifier in [5]. If one or more parameters about the channel are unknown, then ML classifiers become unusable and need a modification. Instead of using the unknown parameter, taking the integral over all possible values can be used to create likelihood functions, as can be seen in [7]. Therefore, the likelihood function for the AMC problem that is derived before becomes

$$L_{ALRT}(\mathbf{r} | m = m_i) = \int_{\Phi} f(\mathbf{r} | \Phi, m = m_i) f(\Phi | m = m_i), \tag{2.7}$$

where f function is the likelihood function that is derived as an ML classifier and Φ represents the set of unknown parameters.

Using this function, the ratio test can be applied and the classification can be concluded. The ratio test is given as

$$class = \begin{cases} m_i & L_{ALRT}(\mathbf{r}|m = m_i) > L_{ALRT}(\mathbf{r}|m = m_j) \\ m_j & L_{ALRT}(\mathbf{r}|m = m_j) > L_{ALRT}(\mathbf{r}|m = m_i) \end{cases}. \tag{2.8}$$

2.2.3. Generalized Likelihood Ratio Test

ALRT also has its problems. First and foremost, it is a computationally complex problem and it is very hard to obtain an exact solution, as shown in [7]. Mostly it comes down to making approximations. Therefore, generalized likelihood ratio test is employed to solve the AMC problem in [7]. Instead of taking an integral over all possible values of the unknown parameters, treating these unknown parameters as unknown but deterministic parameters is suggested. Then, the likelihood function is maximized over these parameters. As a result, the likelihood function becomes

$$L_{GLRT}(\mathbf{r}|m = m_i) = \max_{\Phi}(f(\mathbf{r}|\Phi, m = m_i)) \tag{2.9}$$

where Φ represents the set of unknown parameters. The lack of an integral in the likelihood function results in reduced complexity. These unknown parameters can be CPO, channel gain, and AWGN channel noise variance. In [40], it is shown that the maximum likelihood estimation of CPO is easier to calculate before the other unknown parameters. Therefore, the form is given as

$$L_{GLRT}(\mathbf{r}|m = m_i) = \max_{\Phi}(\max_{\theta_o}(f(\mathbf{r}|\Phi, \theta_o, m = m_i))) \tag{2.10}$$

is more preferable.

Additionally, in [7], maximization over modulation scheme points is also proposed. In its final form, the likelihood function looks like

$$L_{GLRT}(\mathbf{r}|m=m_i) = \max_{\Phi}\left(\max_{\theta}\prod_{n=1}^{N}\max_{m_k}\frac{1}{M(i)}\frac{1}{\sigma\sqrt{2\pi}}e^{\left(\frac{-(r_{I,n}-\mu_{I,k})^2-(r_{Q,n}-\mu_{Q,k})^2}{2\sigma^2}\right)}\right), \tag{2.11}$$

which reduces the complexity of the problem further. Finally, the classification rule is given as

$$class = \begin{cases} m_i & L_{GLRT}(\mathbf{r}|m=m_i) > L_{GLRT}(\mathbf{r}|m=m_j) \\ m_j & L_{GLRT}(\mathbf{r}|m=m_j) > L_{GLRT}(\mathbf{r}|m=m_i) \end{cases}. \tag{2.12}$$

Even though ALRT solves the complexity problem, it introduces a new problem: GLRT creates a biased classifier. GLRT gives weight to some certain points and if two modulation types overlap on one of those points it could favor one of the modulation types to the other one. 4-QAM and 16-QAM, for instance, have this problem. These modulation types have very close modulation points and considering 4-QAM have denser points due to fewer modulation points, the classifier may tend to favor 4-QAM over 16-QAM.

2.2.4. Hybrid Likelihood Ratio Test

Since both ALRT and GLRT have their own problems, a new classifier, which is called hybrid likelihood ratio test (HLRT), emerges. HLRT is basically the hybrid of GLRT and ALRT. In other words, HLRT treats some unknown parameters as probabilistic parameters and treats the others as deterministic parameters. HLRT is employed to solve the AMC problem in [7]. In [7], the likelihood function is averaged over all the unknown parameters except CPO. Then, the resulting function is maximized over CPO to obtain the HLRT likelihood function. As a result, the likelihood

function looks like

$$L_{HLRT}(\mathbf{r}|m = m_i) = \max_{\theta} \left(L(\mathbf{r}|m = m_i, \theta)\right) \tag{2.13}$$

and the decision rule is

$$class = \begin{cases} m_i & L_{HLRT}(\mathbf{r}|m = m_i) > L_{HLRT}(\mathbf{r}|m = m_j) \\ m_j & L_{HLRT}(\mathbf{r}|m = m_j) > L_{HLRT}(\mathbf{r}|m = m_i) \end{cases}. \tag{2.14}$$

2.3. Feature-Based Methods

Some AMC algorithms aim to extract some useful features from the signals instead of using the whole signal. The primary reason for this is to reduce computational complexity. However, these features must be selected carefully, since they should be distinct enough to separate a signal from another and they should carry useful information about the modulation type.

2.3.1. Spectral-Based Features

One of the feature sets that is used in AMC is spectral-based features. The idea of using these features in AMC is employed in [16–18]. There is a total of 9 spectral-based features. The first one is called the maximum value of the power spectral density of the normalized centered instantaneous amplitude and the mathematical expression is given as

$$\gamma_{max} = max \frac{|DFT(A_{cn}[n])|^2}{N}, \tag{2.15}$$

where $A_{cn}[n]$ defined as

$$A_{cn}[n] = \frac{A[n]}{\mu_A} - 1, \tag{2.16}$$

DFT is the discrete Fourier transform, and μ_A is the mean of all points of $A[n]$. This feature controls the signal's amplitude variation. Therefore, the first feature is significant when the modulation type changes the amplitude with time.

The second feature is called the standard deviation of the absolute values of the centered nonlinear components of the instantaneous phase, and the mathematical expression is

$$\sigma_{ap} = \sqrt{\frac{1}{N_c}(\sum_{\frac{A[n]}{\mu_A} > A_T} \phi_{NL}^2[n]) - (\frac{1}{N_c} \sum_{\frac{A[n]}{\mu_A} > A_T} |\phi_{NL}[n]|)^2}, \tag{2.17}$$

where ϕ_{NL} is one sample point's nonlinear component of instantaneous phase, A_T is the threshold value for one sample point to be included in the calculation, and N_c is the number of samples that exceeds the threshold. A threshold is applied in this case, because low-valued sample points are more sensitive to noise. This feature controls the signal's instantaneous phase variation. Therefore, the second feature is significant when the modulation type causes a change of phases.

The third feature is called the standard deviation of the absolute value of the normalized centered instantaneous amplitude in the non-weak segment of the signal and its mathematical expression is given as

$$\sigma_{dp} = \sqrt{\frac{1}{N_c}(\sum_{\frac{A[n]}{\mu_A} > A_T} \phi_{NL}^2[n]) - (\frac{1}{N_c} \sum_{\frac{A[n]}{\mu_A} > A_T} \phi_{NL}[n])^2}. \tag{2.18}$$

Everything is the same with the second feature except for the absolute value operation in the second sum operation. This feature also measures the instantaneous phase changes but it provides the classifier an ability to classify BPSK.

The fourth feature is the spectrum symmetry around the carrier frequency. The mathematical expression is given as

$$P = \frac{\sum_{n=1}^{\frac{f_c N}{f_s}-1} |X_c[n]|^2 - \sum_{n=1}^{\frac{f_c N}{f_s}-1} |X_c[n+\frac{f_c N}{f_s}]|^2}{\sum_{n=1}^{\frac{f_c N}{f_s}-1} |X_c[n]|^2 + \sum_{n=1}^{\frac{f_c N}{f_s}-1} |X_c[n+\frac{f_c N}{f_s}]|^2}, \tag{2.19}$$

where f_c is the carrier frequency, f_s is the sampling frequency, and X_c is discrete-time Fourier transform of $A[n]$. This feature provides the classifier an ability to distinguish different amplitude modulation schemes.

The fifth feature is called the standard deviation of the absolute value of the normalized centered instantaneous amplitude of the signal segment and its mathematical expression is given as

$$\sigma_{aa} = \sqrt{\frac{1}{N}(\sum_{n=1}^{N} A_{cn}^2[n]) - (\frac{1}{N}\sum_{n=1}^{N} |A_{cn}[n]|)^2}. \tag{2.20}$$

This time, all symbols in the sequence are included in the calculation. Therefore, N is the number of samples in the symbol sequence. This feature also controls the signal's amplitude variation. However, it provides the classifier an ability to classify 2-ASK modulation.

The sixth feature is called the standard deviation of the absolute value of the normalized and centered instantaneous frequency of the signal. The mathematical expression is given as

$$\sigma_{af} = \sqrt{\frac{1}{N_c}(\sum_{\frac{A[n]}{\mu_A} > A_T} f_N^2[n]) - (\frac{1}{N_c}\sum_{\frac{A[n]}{\mu_A} > A_T} |f_N[n]|)^2}, \tag{2.21}$$

where

$$f_N[n] = \frac{f[n] - \frac{1}{N}\sum_{n=1}^{N} f[n]}{f_s}, \tag{2.22}$$

and $f[n]$ is the instantaneous frequency vector. This feature provides the classifier an ability to distinguish 2-FSK and 4-FSK modulations.

The seventh feature is called the standard deviation of the normalized and centered instantaneous amplitude and its mathematical expression is given as

$$\sigma_a = \sqrt{\frac{1}{N_c}(\sum_{\frac{A[n]}{\mu_A}>A_T} A_{cn}^2[n]) - (\frac{1}{N_c}\sum_{\frac{A[n]}{\mu_A}>A_T} A_{cn}[n])^2}. \tag{2.23}$$

The eighth feature is called the kurtosis of the normalized and centered instantaneous amplitude and its mathematical expression is given as

$$\mu_{42}^a = \frac{E\{A_{cn}^4[n]\}}{(E\{A_{cn}^2[n]\})^2}. \tag{2.24}$$

This feature provides the classifier an ability to distinguish amplitude-based analog modulations and amplitude-based digital modulations.

Finally, the last spectral-based feature is called the kurtosis of the normalized and centered instantaneous frequency and its mathematical expression is given as

$$\mu_{42}^f = \frac{E\{f_N^4[n]\}}{(E\{A_N^2[n]\})^2}. \tag{2.25}$$

This feature provides the classifier an ability to distinguish frequency-based analog modulations and frequency-based digital modulations.

All these features can be used to form a decision tree that classifies both analog and digital modulation types.

2.3.2. Wavelet Transform-Based Features

Another feature set that can be used in the AMC area is the wavelet transform-based feature set. The first idea to use the wavelet transform in the AMC area is introduced in [19]. Continuous wavelet transform (CWT) is used for these features and its mathematical formulation is given as

$$CWT(x;a,b) = \int_{-\infty}^{\infty} x(t)\psi_{a,b}^{*}(t)dt, \tag{2.26}$$

where $x(t)$ is the function to be transformed, ψ is the wavelet function that is used to take wavelet transform, and a and b are the parameters for the wavelet function. Notice that the conjugation of the wave function is used in CWT.

There are numerous wave functions that have been used in the literature since the 1970's when the wavelet theory was developed. There are both discrete and continuous wavelets but in this case, continuous wavelets are used since the feature set uses CWT. Morlet wavelet, Meyer wavelet, Poisson wavelet, or Shannon wavelet can be given as examples. Wavelet functions can be relatively computationally expensive functions, such as Shannon wavelet, which can be expressed as

$$\psi_{Shannon}(t) = \operatorname{sinc}\left(\frac{t}{2}\right)\cos\left(\frac{3\pi t}{2}\right). \tag{2.27}$$

This complexity drives the researchers away from complex wavelet functions, such as Shannon wavelets, to less complex wavelets. Therefore, Haar wavelet is used to extract the feature set for AMC. Haar wavelet's mathematical expression is

$$\psi_{Haar}(t) = \begin{cases} 1, & 0 \leq t < \frac{T}{2}, \\ -1, & \frac{T}{2} \leq t < T, \\ 0, & otherwise, \end{cases} \tag{2.28}$$

and $\psi_{a,b}(t)$ can be expressed as

$$\psi_{a,b}(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right). \tag{2.29}$$

CWT of each modulation scheme is computed in [19]. For example, CWT of a FSK signal is

$$|CWT(x;a,b)| = \frac{4\sqrt{S}}{(w_c+w_n)\sqrt{a}}\sin^2\left[\frac{(w_C+w_n a T_s)}{4}\right], \tag{2.30}$$

whereas for a PSK signal, it is

$$|CWT(x;a,b)| = \frac{4\sqrt{S}}{(w_c+w_n)\sqrt{a}}\sin^2\left[\frac{w_c a T_s}{4}\right]. \tag{2.31}$$

Finally, CWT of a QAM signal is

$$|CWT(x;a,b)| = \frac{4|A_n|}{(w_c+w_n)\sqrt{a}}\sin^2\left[\frac{w_c a T_s}{4}\right], \tag{2.32}$$

where w_c is the carrier frequency, w_n is each symbol's frequency, A_n is each symbol's amplitude, T_s is the sampling interval, and S is the energy of the signal. Additionally, CWT of ASK is computed in [20]. Its mathematical expression is

$$|CWT(x;a,b)| = \frac{4|A_n|}{(w_c+w_n)\sqrt{a}}\sin^2\left[\frac{w_c a T_s}{4}\right], \tag{2.33}$$

which is the same with CWT of a QAM signal. Therefore, it is not very easy to classify QAM and ASK signals. However, since the transforms are distinct enough, PSK and FSK signals can be classified accordingly.

2.3.3. Higher Order Statistics-Based Features

Higher-order statistics-based feature sets are also used in the AMC area. Especially, the cumulant-based features are very popular in the literature. These features

are actually related to the moments of the signal. According to [41], cumulants can be defined around characteristic functions, or in other words, moment generating functions. The characteristic function is defined as

$$\Psi(t) = E[e^{ixt}] = \sum_{n=0}^{\infty} \frac{1}{n!} \mu_n (it)^n \tag{2.34}$$

and the rest of the Taylor expansion of exponential function comes after the characteristic function. As a result, the cumulant generating function can be defined by the logarithmic characteristic function,

$$\ln(\Psi(t)) = \sum_{n=0}^{\infty} \frac{1}{n!} \kappa_n (it)^n, \tag{2.35}$$

and its Taylor expansion coefficients are considered as cumulants. One way of obtaining the cumulants is taking the derivative of the function. Therefore, the definition of cumulants is

$$C_n = \left. \frac{\mathrm{d}^n \ln(\Psi(t))}{\mathrm{d}t^n} \right|_{t=0}. \tag{2.36}$$

After the computation of cumulants, higher-order cumulants can be expressed as

$$C_{mn} = cum(\underbrace{r[n], \ldots, r[n]}_{m-n}, \underbrace{r^*[n], \ldots, r^*[n]}_{n}), \tag{2.37}$$

where * denotes the conjugation operation.

These features give a great insight into signals spread on the complex plane. Therefore, as the signal patterns get similar on the complex plane, cumulants also get closer value-wise.

Finally, the cumulants up until the 6th order can be written as

$$C_{20} = M_{20}, \tag{2.38}$$

$$C_{21} = M_{21}, \tag{2.39}$$

$$C_{40} = M_{40} - 3M_{20}^2, \tag{2.40}$$

$$C_{41} = M_{41} - 3M_{20}M_{21}, \tag{2.41}$$

$$C_{42} = M_{42} - |M_{20}|^2 - 2M_{21}^2, \tag{2.42}$$

$$C_{60} = M_{60} - 15M_{20}M_{40} + 30M_{20}^3, \tag{2.43}$$

$$C_{61} = M_{61} - 5M_{21}M_{40} - 10M_{20}M_{41} + 30M_{20}^2M_{21}, \tag{2.44}$$

$$C_{62} = M_{62} - 6M_{20}M_{42} - 8M_{21}M_{41} - M_{20}M_{40} + 6M_{20}^2M_{20} + 24M_{21}^2M_{20}, \tag{2.45}$$

$$C_{63} = M_{63} - 9M_{21}M_{42} + 12M_{21}^3 - 6M_{20}M_{41} + 18M_{20}M_{21}M_{20}, \tag{2.46}$$

where M_{pq} is

$$M_{pq} = E[r[n]^{p-q}(r^*[n])^q]. \tag{2.47}$$

We can approximate M_{pq} by taking the arithmetic average of a signal vector. As a result, the approximation of M_{pq} is,

$$\widehat{M_{pq}} = \frac{1}{N}\sum_{n=1}^{N} r^{p-q}[n](r^q[n])^* \tag{2.48}$$

where N is the length of signal vector.

All cumulant values of some popular modulation types in an ideal scenario where noise does not exist can be seen in Table 2.1. As it can be seen from Table 2.1, values of 16-QAM's cumulants and 64-QAM's cumulants are very similar, since they have very similar patterns. However, they are still distinct enough to be used in classification.

Table 2.1. Cumulant values of some popular modulation types.

	BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
$\mathbf{C}_{20}$	1	0	0	0	0
$\mathbf{C}_{21}$	1	1	1	1	1
$\mathbf{C}_{40}$	-2	1	0	-0.680	-0.619
$\mathbf{C}_{41}$	-2	0	0	0	0
$\mathbf{C}_{42}$	-2	-1	-1	-0.680	-0.618
$\mathbf{C}_{60}$	16	0	0	0	0
$\mathbf{C}_{61}$	16	-4	0	2.08	1.7972
$\mathbf{C}_{62}$	16	0	0	0	0
$\mathbf{C}_{63}$	16	4	4	2.08	1.7972

2.4. Machine Learning-Based Methods

Machine learning tools have the ability to classify a set of elements in the given space by creating decision rules. Therefore, they are also very popular in the AMC area.

2.4.1. K-Nearest Neighbor

K-nearest neighbor (KNN) is one of the machine learning tools that is used in AMC. It is a supervised learning which means it needs the number of classes beforehand. Fortunately, the number of classes is determined before the classification process in typical AMC scenarios. The process of KNN can be defined in 6 basic steps.

(i) First of all, reference signals must be established and a feature set must be extracted. As reference signals, modulated signals are used in AMC. The noise effects on the signal depend on the experiment. Reference signals are used as a reference for the test data. In the previous section, a number of features are introduced. Each one can be used in the KNN algorithm. However, the cumulant

feature set is the popular option in the literature since this feature set is easy to use and efficient.

(ii) The second step is feeding test signals to the system and extracting the corresponding feature set. Modulation types of the test signals are not known.

(iii) After establishing the feature set, the space where classification takes place is also created. Each feature in the feature set acts as a dimension in this space. In the third step, distances between a test signal and reference signals are calculated in the established space. There are many distance metrics that can be used to compute the distance between two signals and in this step, a metric should also be established. One of the popular distance metrics is Euclidean distance and it can be defined as

$$Dist(F(A), F(B)) = \sqrt{\sum_{l=1}^{N} [F_l(A) - F_l(B)]^2}, \tag{2.49}$$

where A and B are the two signals and N is the number of features that a signal has. Another distance metric that is used in the KNN classifier is Minkowski distance. Its mathematical expression is

$$Dist(F(A), F(B)) = \left(\sum_{l=1}^{N} \left| F_l(A) - F_l(B) \right|^p \right)^{\frac{1}{p}}. \tag{2.50}$$

Notice that when p is 2, it becomes equivalent to the Euclidean distance. When p is 1, then it is called Manhattan Distance and it can be expressed as

$$Dist(F(A), F(B)) = \sum_{l=1}^{N} \left| F_l(A) - F_l(B) \right|. \tag{2.51}$$

Another metric that can be used in KNN is Cosine distance. It assumes that the

feature set for one signal is a vector and its mathematical expression is

$$Dist(F(A), F(B)) = \frac{\overrightarrow{F(A)}.\overrightarrow{F(B)}}{||\overrightarrow{F(A)}||.||\overrightarrow{F(B)}||}. \tag{2.52}$$

This distance gives the similarity between two vectors.

(iv) After all distances are calculated for a test signal, all computed distances are sorted from minimum to maximum.

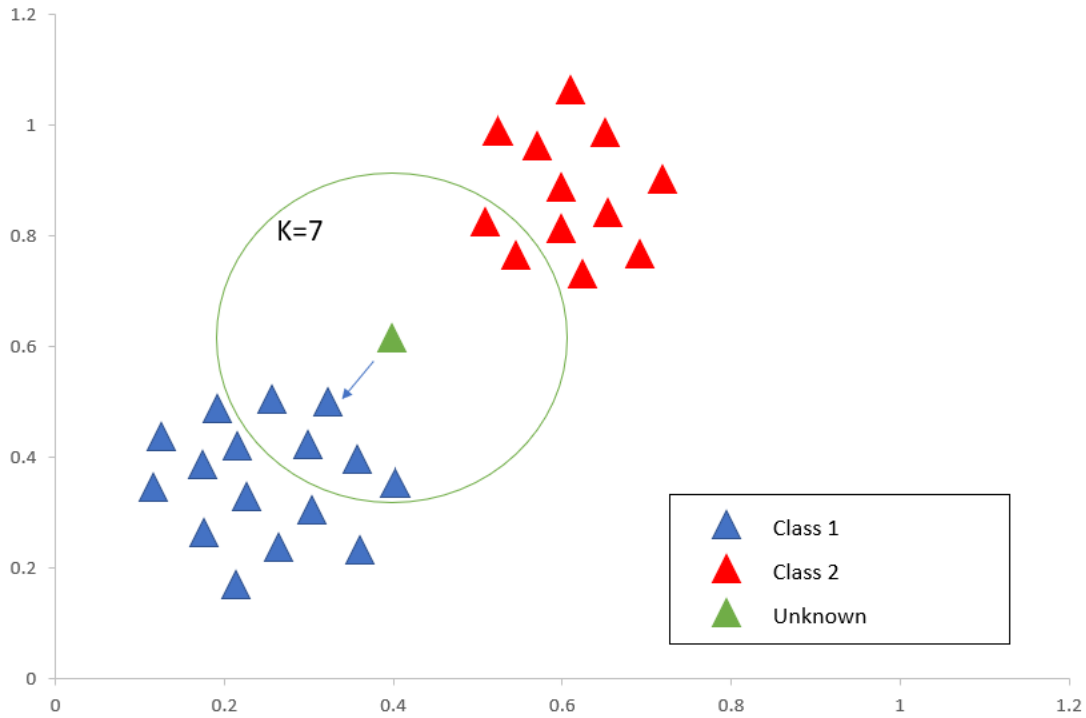


Figure 2.3. 2-D KNN visualization.

(v) k shortest distances are taken into account. Here number k is defined by the user. It should not be too low to miss any information and it should not be too high to include unimportant distances. Additionally, the number k should prevent any evenness as much as possible.

(vi) In the last step, the test signal is classified to the class that has the most samples in the shortest distances.

Visualization of a 2-D KNN algorithm example can be seen in Figure 2.3.

KNN can perform multi-class classifications and does not need to know any parameters related to signals. Therefore, it is very easy to use for many classification problems. The only problem is that when the number of features increases, the computational efficiency decreases. Consequently, some feature reduction methods are generally used before implementing the KNN algorithm.

2.4.2. Support Vector Machine

Support Vector Machine (SVM) is another machine learning tool that is used in the AMC area. In KNN, each sample point is classified individually, but in SVM, it creates a hyperplane in the space which is itself a decision rule. Therefore, it is generally used as a 2-class classifier. The hyperplane can be expressed by an n-dimensional weight vector, $\mathbf{w}$, and an offset scalar, w_o. Creating this hyperplane itself can be summarized in 5 steps.

(i) The first step is establishing the feature set. Again, cumulants are a very popular choice here. Then, training samples are created and these samples are used to create the hyperplane. Lastly, $\mathbf{w}$ and w_0 should also be initialized.

(ii) In the second step, the weight vector and offset scalar are updated according to the reference of training samples. In SVM, two functions must be maximized; one of them is the margin, which is the distance between the hyperplane and the closest point to the hyperplane, the other one is the negative loss function. The margin maximization function is

$$\frac{2}{||\mathbf{w}||^2} \tag{2.53}$$

and the negative loss function is

$$-\sum_{n=1}^{N}[y_i(\mathbf{w}^T\mathbf{x}_i + w_0) - 1], \tag{2.54}$$

where y_i is ± 1, which indicates the class of input sample vector, and N is the number of sample vectors. By maximizing these two functions, $\mathbf{w}$ and w_o are updated.

(iii) If one the stopping conditions is not achieved, repeat the second step; if it is achieved, stop the algorithm. Stopping conditions can either be the successful classification of all samples or reaching the pre-determined number of loops.

(iv) After the updating process is finalized, test samples' locations are calculated with respect to the hyperplane, which can be expressed as

$$g(x) = \mathbf{w}^T\mathbf{x} + w_o. \tag{2.55}$$

(v) Classification is performed by looking at the $g(x)$ value. The decision rule is

$$Class = \begin{cases} m_i, & g(x) = \mathbf{w}^T\mathbf{x} + w_o \geq 0 \\ m_j, & g(x) = \mathbf{w}^T\mathbf{x} + w_o < 0 \end{cases}. \tag{2.56}$$

Visualization of a 2-D SVM algorithm example can be seen in Figure 2.4.

After computing the hyperplane, it is relatively easy to classify test samples, since it is an easy vector multiplication and training samples are no longer inside the picture. The training part has also reduced the computational budget. However, unlike KNN, SVM is generally used for 2-class classification.

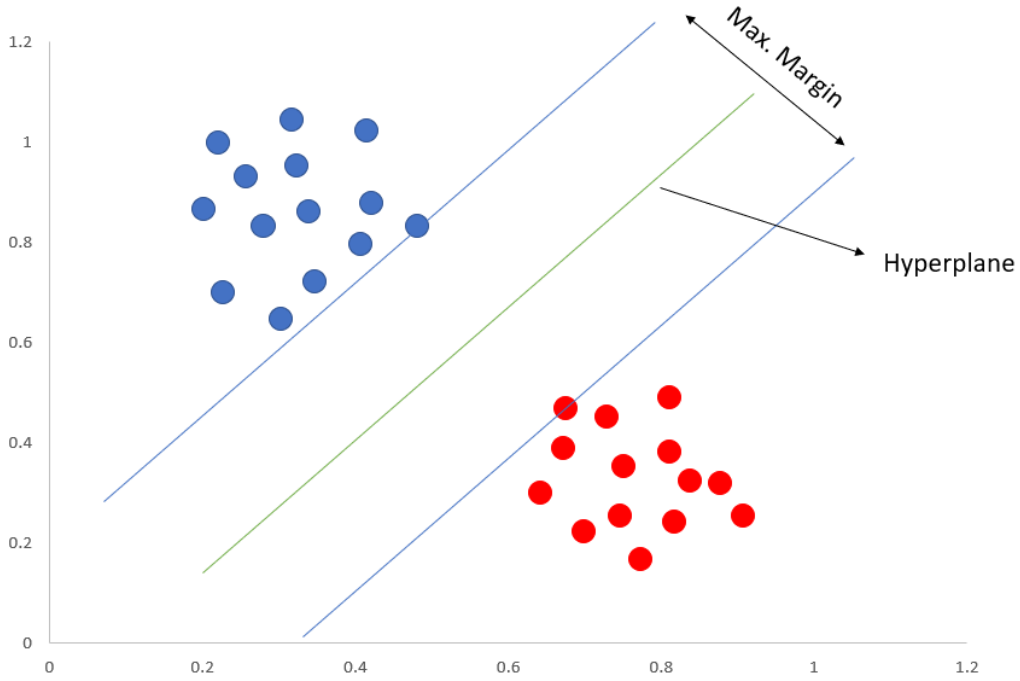


Figure 2.4. 2-D SVM visualization.

2.5. Feature Reduction Algorithms

As the number of features increases, the methods become also computationally expensive. Therefore, in the literature, there are various methods to decrease the number of features by morphing or combining different features in the feature set. In this section; Logistic Regression, Artificial Neural Networks (ANN), and Genetic Programming will be discussed.

2.5.1. Logistic Regression

Logistic regression simply is a data fitting algorithm. It is generally used to fit data that is available in n-dimensional space to a lower-dimensional space. The resulting fit cannot represent all data with complete accuracy but it aims to maximize this accuracy. Logistic regression morphs and combines these dimensions into new dimensions.

Its morphing and combining abilities are also useful for feature reduction. Therefore, it is one of the methods to reduce the number of features in the AMC area. The feature set is treated as space dimensions. This feature reduction can be expressed as

$$\mathbf{f} = \mathbf{W}\mathbf{g} + \mathbf{w}_o, \tag{2.57}$$

where $\mathbf{g}$ is the older $M \times 1$ feature vector, $\mathbf{f}$ is the new $N \times 1$ feature vector, $\mathbf{W}$ is an $M \times N$ weight matrix and $\mathbf{w}_o$ is an $M \times 1$ weight offset vector. $\mathbf{W}$ can be updated by iterative algorithms. In [42], some of the algorithms that update $\mathbf{W}$ matrix are evaluated.

Feature reduction methods also cause some pieces of information to be lost naturally and feature reduction algorithms try to minimize this loss. However, sometimes logistic regression may cause a loss of a significant portion of the information. Therefore, these methods should be operated carefully.

2.5.2. Artificial Neural Networks

Artificial neural networks are first introduced to the literature in [43]. Neural networks in human brains inspire mathematicians to model this chain of command on computation in an entirely artificial environment. As a result, many neural network implementations are introduced to the literature.

One of the popular ANN architectures is Multilayer perceptron model (MLP). MLP consists of layers and nodes. MLP has at least 2 layers and each layer has a number of nodes which are called artificial neurons. Each consecutive layers' neurons are connected with each other, which simulates synapse. All of the connections have a weight attached to them. By updating these weights, the desired outcome is achieved from the end layer's neurons.

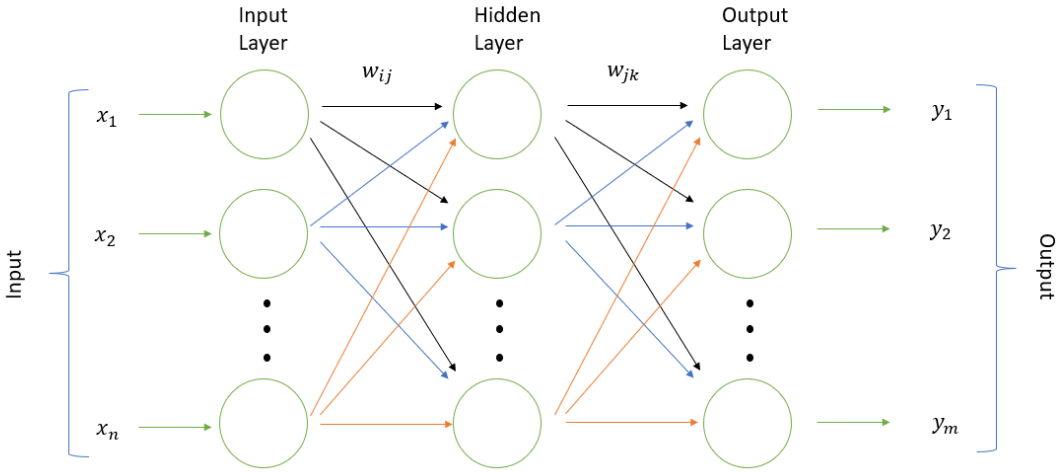


Figure 2.5. 3-Layer MLP Example.

In MLP, the first layer is called the input layer, the last layer is called the output layer, and the rest of the middle layers are called hidden layers. A 3-layered MLP model is given in Figure 2.5. In a 3-layered MLP model, the mathematical expression for an output neuron is

$$y_k = \phi\left(\sum_{i=1}^{N} w_{ki}\phi\left(\sum_{j=1}^{M} w_{ij}x_j\right)\right), \tag{2.58}$$

where y_k is k^{th} output neuron, w is weight on the connections, x_j is the jth input neuron, and ϕ is the activation function. An activation function can be chosen from many functions. One such function is called the sigmoid function and its mathematical expression is given as

$$\phi(x) = \frac{1}{1+e^{-x}}. \tag{2.59}$$

This function is popular because it has a very easy derivative that is given as

$$\frac{\mathrm{d}\phi(x)}{\mathrm{d}x} = [1-\phi(x)]\phi(x). \tag{2.60}$$

Derivation is an important part of the MLP models, since the gradient descent algorithm is used to update weights of the neuron connections. First, the error function is calculated and then by using back propagation and gradient descent algorithms, weights are updated. Each layer results in one gradient and combining these equations results in a gradient chain. One such chain example is

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_i}\frac{\partial y_i}{\partial u_i}\frac{\partial u_i}{\partial w_{ij}} \tag{2.61}$$

and weights can be updated by using

$$w_{ij}(t+1) = w_{ij}(t) - \kappa\frac{\partial E}{\partial w_{ij}}, \tag{2.62}$$

where κ is the learning rate.

MLP models and general neural networks can be used for classification by setting the number of neurons at the output layer to the number of classes in the dataset. In this case, model parameters can be updated accordingly to perform classification. In [31, 32], ANN is used for AMC. ANN models can also be used for feature reduction by setting the number of neurons on the output layer lower than the number of layers on the input layer.

2.5.3. Genetic Programming

Genetic programming (GP), similar to ANN, also emerged by imitating the biological life forms. It is also similar to ANN in the way of updating the parameters of the algorithm by measuring the performance. In [34], the idea of integrating GP to the AMC area is proposed. GP is used as a way to reduce the number of features in a feature set and reduce the complexity along with it. The algorithm for GP can be summarized in 5 steps.

(i) First, the dataset needs to be established as usual. In the AMC case, this dataset consists of a feature set that is extracted from modulated signals. Like the other algorithms, cumulants-based feature sets are also a popular choice for the GP algorithms.

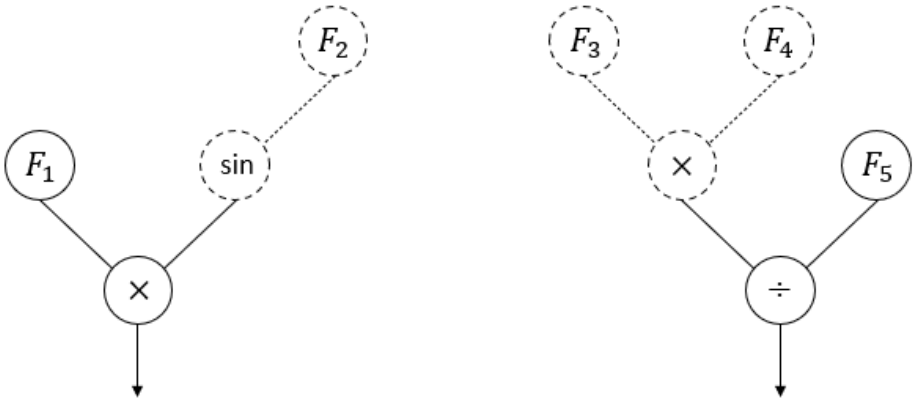


Figure 2.6. Parent branches before cross-over operation.

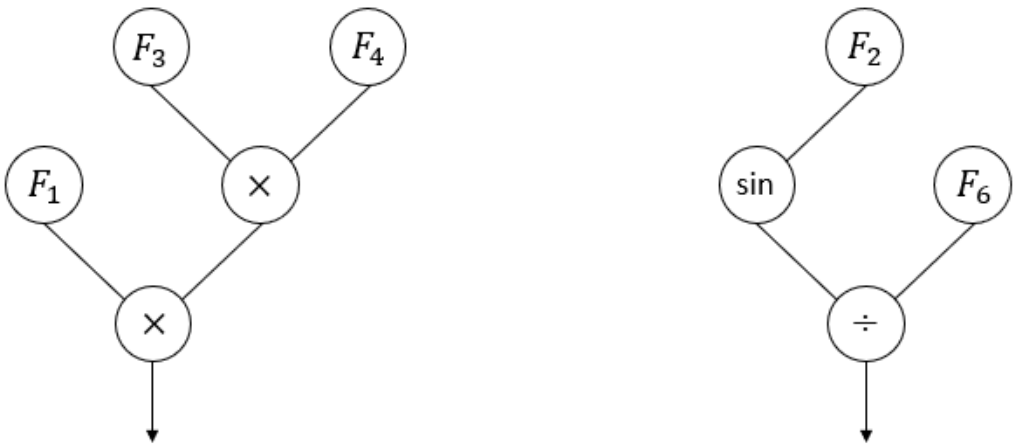


Figure 2.7. Child branches after cross-over operation.

(ii) In the second step, the starting values are selected from the feature set. These features will be combined and morphed into new features. Thus, the number of features will be reduced.

(iii) The iteration starts in this step. Initialized parameters form trees with each other by combining features by some mathematical operators. Thus, parent trees are created. Then, the fitness evaluations are done. If the stopping conditions are not achieved, the iteration continues. If the iteration continues, new children branches are created by generating new random branches and trees or using crossover and mutations on the available branches. Generating new branches or trees is the same as the initialization process. Crossover is exchanging branches between two existing trees. Finally, Mutation is generating a new branch instead of an existing branch on an existing tree. Crossover operation is given in Figures 2.6 and 2.7 and mutation operation is given in Figure 2.8.

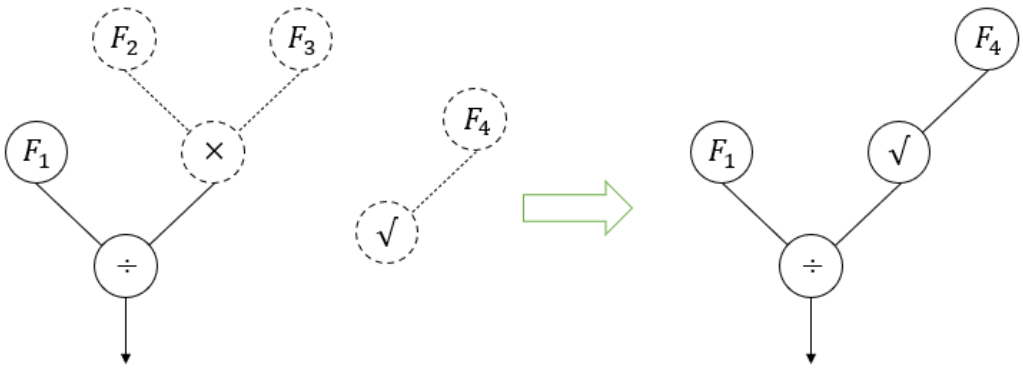


Figure 2.8. Mutation Operation.

(iv) If the desired performance is achieved or the performance stays stagnant over time, the process stops. There are several ways to form a fitness evaluation. One of the methods, which is also used in [34], is using the resulting features in the main problem. If the achieved result is satisfying, then the process would stop.

(v) Finally, obtained features are used for the classification.

GP is a very efficient algorithm. It does not only reduce the computational complexity but also improves the performance of the classifier.

2.6. Convolutional Neural Network-Based Classification

Deep learning algorithms have been immensely popular in recent years because of their ability to solve very hard problems with great performance. There are various deep learning sub-categories such as Recurrent Neural Networks (RNN), Feedforward Neural Networks (FNN), or Convolutional Neural Networks (CNN). However, due to its ability to analyze multidimensional data, CNN is a popular choice among the deep learning networks.

The general principle of CNN is very similar to MLP. It is not surprising because of the fact that MLP is also accepted as a deep learning network. Nonetheless, MLP has only one kind of layer which is called fully connected dense layer but CNN has much more variety when it comes to layers. Therefore, CNN is more flexible when solving problems the AMC problem.

Besides fully connected dense layers, CNN has convolutional layers, pooling layers, dropout layers, and batch normalization layers. While some of these layers include parameters to be trained, others mainly act as regularizers. The key is combining these layers in a way that to the problems with a great performance. Since all these layers are connected consecutively like MLP, training the parameters are also similar to the MLP training. Loss functions are calculated first and then, by back propagation algorithms, each parameter in the layers is trained. For more information about deep learning and its concept, please refer to [44].

2.6.1. Convolutional Layers

Convolution in multi-dimensional space is performed similar to the 1-D convolution. The convolution filter is shifted through the whole space. In each step, the parameters in the filter are multiplied with the values in the space where the filter is located and all multiplied values are added together to find the result of the convolution for the respective location. Performing the same step for each location results in a convolution operation. Visualization of 2-D convolution can be seen in Figure 2.9.

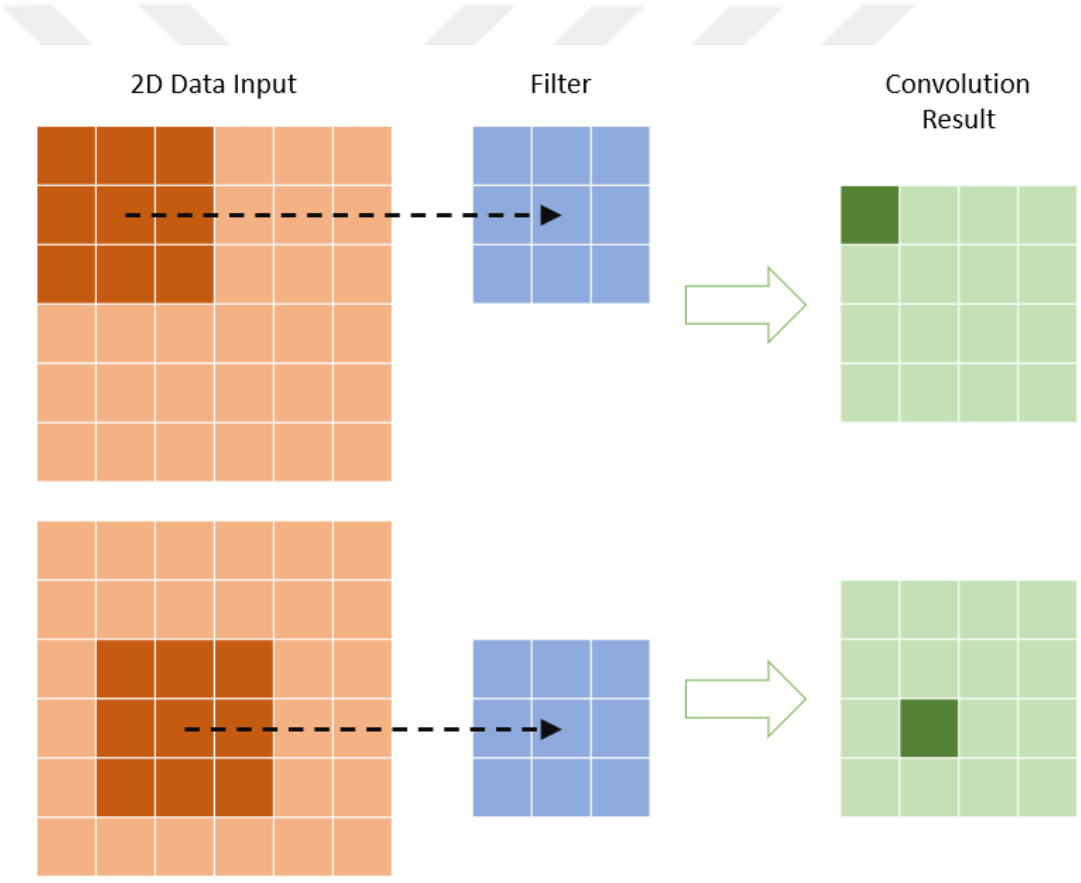


Figure 2.9. 2-D convolution visualization.

Convolution in multi-dimensional space has been very popular because with a specific convolution filter, some features can be extracted. For example, consider

$$\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix} \tag{2.63}$$

as a convolution filter and consider

$$\begin{bmatrix} & & & \vdots & & & & & \\ & 10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 & \\ \dots & 10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 & \dots \\ & 10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 & \\ & 10 & 10 & 10 & 10 & 0 & 0 & 0 & 0 & \\ & & & & \vdots & & & & & \end{bmatrix} \tag{2.64}$$

as the space. If the convolution is performed, the result is

$$\begin{bmatrix} & & & \vdots & & & & & \\ & 0 & 0 & 0 & 30 & 30 & 0 & 0 & 0 & \\ \dots & 0 & 0 & 0 & 30 & 30 & 0 & 0 & 0 & \dots \\ & 0 & 0 & 0 & 30 & 30 & 0 & 0 & 0 & \\ & 0 & 0 & 0 & 30 & 30 & 0 & 0 & 0 & \\ & & & & \vdots & & & & & \end{bmatrix}. \tag{2.65}$$

If the result is correctly analyzed, it can be seen that only the edges in the space are highlighted in the results. Therefore, the specific convolution filter that is used in the convolution operation is used to detect vertical edges. Therefore; if the filters are specified for a certain problem correctly, they can bring up the necessary features. Therefore, convolution can be very powerful in multidimensional spaces. Instead of designing each filter individually, CNN can train these parameters by using loss function and back propagation. One of the most popular methods during back propagation is

gradient descent similar to the MLP.

2.6.2. Fully Connected Dense Layers

Fully connected dense layers or shortly dense layers have the same properties as the dense layers that are explained in ANN. Each layer has a number of neurons and each pair of consecutive layers' neurons are connected with each other.

In CNN, the input data is mostly multi-dimensional, but the desired output is generally a single output or a group of single outputs if the aim is to make a classification or regression. Since the neurons in the dense layers have the ability to give a single output, dense layers are generally chosen as an output layer.

2.6.3. Pooling Layers

Pooling layers are used to make the data smaller and denser by limiting the loss of information. There are not any parameters to train in pooling layers, since they use deterministic ways to shrink the size of the data.

There are two main pooling layers that are used in the industry. One of them is max-pooling layers. Max-pooling layers take a frame with a pre-determined size from the data and give the maximum number in the frame as an output. A 2×2 max-pooling layer example can be given as

$$\begin{bmatrix} 4 & 7 & 6 & 14 \\ 3 & 12 & 11 & 11 \\ 10 & 5 & 14 & 3 \\ 20 & 16 & 18 & 3 \end{bmatrix} \xrightarrow{2\times2 \text{ max pooling layer}} \begin{bmatrix} 12 & 14 \\ 20 & 18 \end{bmatrix}. \tag{2.66}$$

In this example, the 4×4 matrix is divided into 4 2×2 matrices and each $2x2$ matrix' maximum element is written in the place of the 2×2 matrix. As a result, a 2×2

matrix is obtained.

The other one is average pooling layers. Average pooling layers take a frame with a pre-determined size from the data and give the average of all numbers in the frame as an output. A 2×2 average pooling layer example can be given as

$$\begin{bmatrix} 4 & 6 & 6 & 14 \\ 3 & 11 & 11 & 9 \\ 10 & 6 & 14 & 5 \\ 20 & 16 & 18 & 3 \end{bmatrix} \xrightarrow{2 \times 2 \text{ average pooling layer}} \begin{bmatrix} 6 & 10 \\ 13 & 10 \end{bmatrix}. \tag{2.67}$$

In this example, the 4×4 matrix is divided into 4 2×2 matrices and the averages of each $2x2$ matrix' 4 elements are written in the place of the 2×2 matrix. As a result, a 2×2 matrix is obtained.

2.6.4. Batch Normalization

Sometimes input data of one or more of the middle layers may grow out of desired bounds or deviate from the desired operating range. Therefore, depending on the problem, there may be a need of keeping everything in check and stable. Normalization of the batch in the middle of CNN layers may tick those boxes. Therefore, batch normalization is a very popular layer in the industry. Batch normalization can be done through

$$\widehat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}}, \tag{2.68}$$

where

$$\mu_B = \frac{1}{m} \sum_{i=1}^{m} x_i, \tag{2.69}$$

$$\sigma_B^2 = \frac{1}{m} \sum_{i=1}^{m} (x_i - \mu_B)^2, \tag{2.70}$$

and x is the input data. As it can be seen from the equations, batch normalization layers do not have any trainable parameters.

2.6.5. Dropout

In a CNN architecture, there can be thousands of parameters easily and all these kinds of systems tend to prioritize some of the parameters over the others and depending on the problem, it could decrease the performance of the system. To overcome this problem, dropout is developed. Dropout temporarily disables some of the connections. Therefore, each connection contributes to the solution similarly. Additionally, some mistakes may be re-adjusted after disabling some of these connections.

2.6.6. Conclusion

By using all of these layers, various CNN architectures can be constructed. One such example can be seen in Figure 2.10 which is called AlexNet. AlexNet is explained in detail in [45].

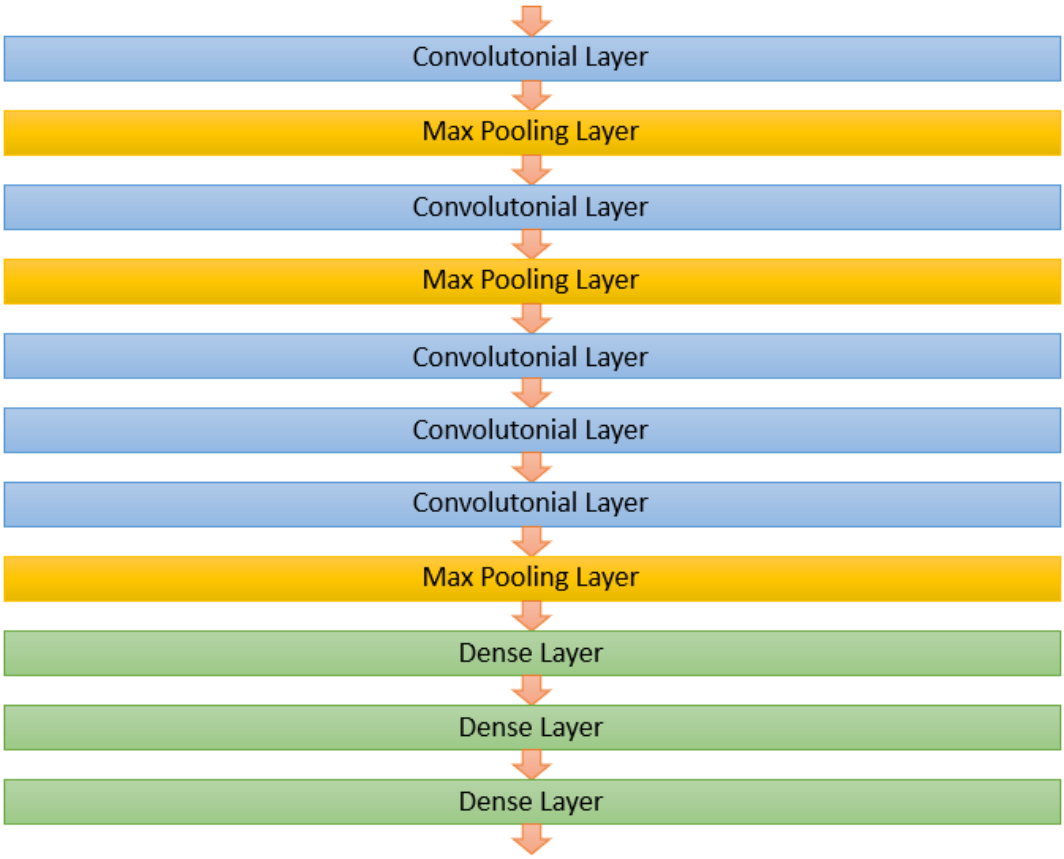


Figure 2.10. AlexNet Architecture.

3. FEATURE-BASED AMC AND PERFORMANCE EVALUATION

In this chapter, we will talk about the performance evaluation of the feature-based method that is proposed in this thesis.

3.1. Cumulant-Based Classification

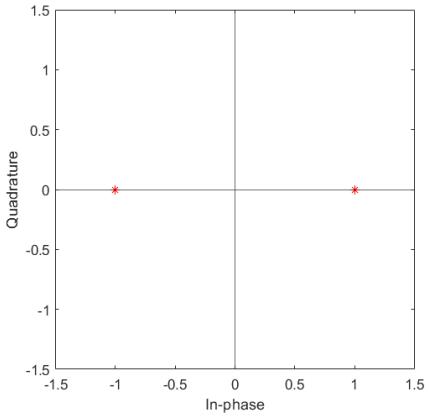
To use a feature-based classification algorithm, the feature set needs to be selected first. Due to their popularity and efficiency in AMC applications, cumulants are used to perform feature-based classification. After choosing the feature set, the classification algorithm should also be defined. Machine learning algorithms are well-suited for this task, so the KNN algorithm is chosen to perform classification. In KNN, a distance metric needs to be determined at the initialization step and Euclidean distance is selected as the distance metric.

The next step is to establish target modulation classes and the corresponding dataset. In this thesis; BPSK, QPSK, 8-PSK, 16-QAM, and 64-QAM are chosen as the target modulation classes. These modulation types' constellation diagrams can be seen in Figure 3.1.

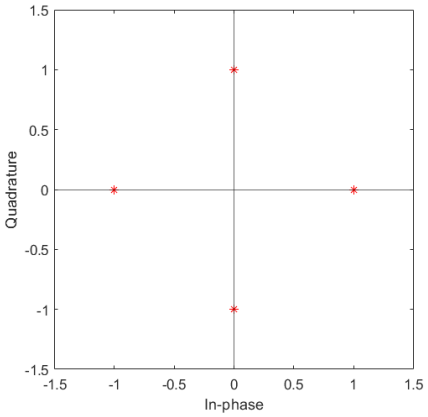
The next step is to generate the dataset. In this part, the only effect that the signal model has is AWGN. Therefore the signal model expression is given as

$$r[n] = x[n] + g[n], \tag{3.1}$$

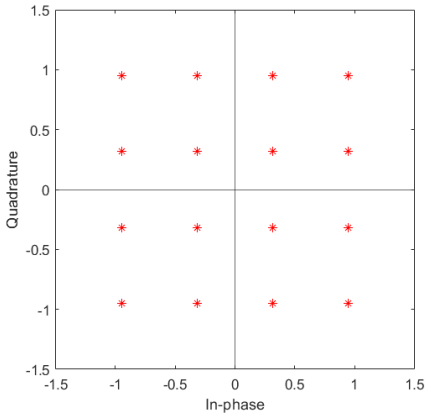
where x[n] is the symbol vector, g[n] is the AWGN component, and r[n] is the received signal. The symbol vector of each modulated signal consists of 2000 symbols. For each modulation type, 1000 test signals have been generated for each even signal to noise ratio (SNR) value from 0 to 20 dB.



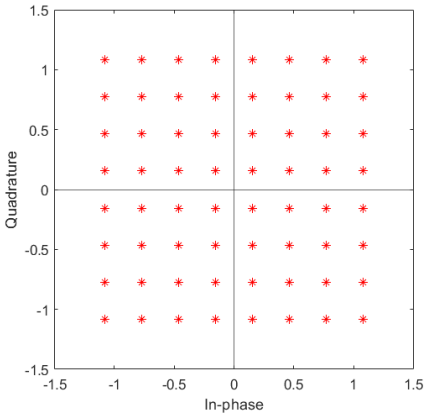
a) BPSK



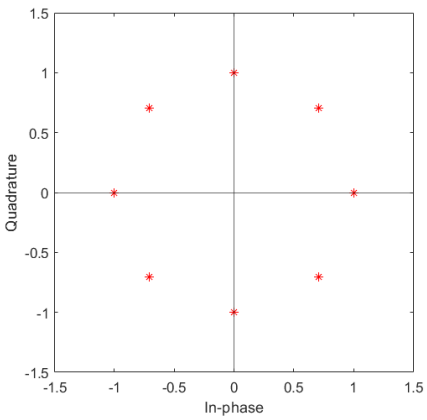
b) QPSK



d) 16-QAM



e) 64-QAM



c) 8-PSK

Figure 3.1. Constellation diagrams of the selected modulation types.

As a result, there are a total of 55000 test signals. After generating test signals, the feature set for each test signal is needed to be extracted. Feature set consists of 9 distinct cumulants: C_{20}, C_{21}, C_{40}, C_{41}, C_{42}, C_{60}, C_{61}, C_{62}, and C_{63}. This also means that the space where KNN is performed is 9-dimensional. In other words, each signal is represented by a 1×9 vector. 10 reference signals for each modulation type at 20 dB SNR are also generated. After establishing the reference and test signals, KNN is utilized to perform modulation classification.

3.1.1. Results

The results can be seen in Table 3.1. The results for BPSK, QPSK, and 8-PSK are encouraging. Cumulants manage to classify these modulations types correctly even in relatively low SNR values. However, from the results, it can also be inferred that the cumulants are not very successful to classify 16-QAM and 64-QAM. At first, the performance improves as the SNR increases but the performance stays the same when the SNR is higher. The problem is clear: The classifier confuses both modulation types to each other even though the SNR is higher.

The bad news is, an error floor is encountered during this classification; the good news is that the bad performance only happens when the modulation type is either 16-QAM or 64-QAM.

Table 3.1. Confusion matrices of the KNN method.

4 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	3685	1315
	64-QAM	0	0	0	1332	3668

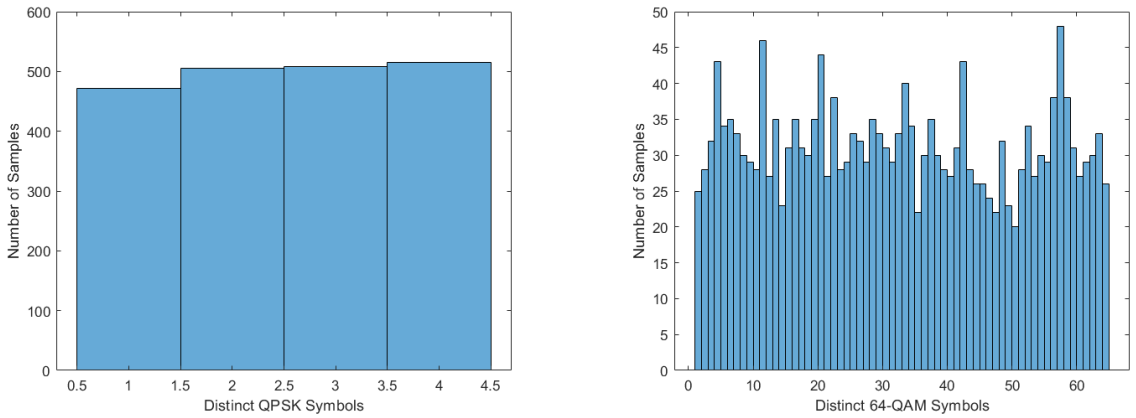
Table 3.1. (cont.)

		BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
12 dB SNR	**BPSK**	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	4276	724
	64-QAM	0	0	0	709	4291
		BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
20 dB SNR	**BPSK**	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	4435	565
	64-QAM	0	0	0	623	4377

3.1.2. Root of the Problems

There could be several reasons for this poor performance. One of them could be the length of the symbol sequence. Even though the distribution of symbols has a uniform distribution, some marginal cases may appear if the number of symbols is not sufficiently high. Since 16-QAM and 64-QAM have more distinct symbols than the other modulation types, the chance of marginal cases to be occurring is not low. Comparison between QPSK and 64-QAM histograms is given in Figure 3.2. Consequently, these marginal cases may affect the cumulants significantly, which causes poor performance.

Another reason is that 16-QAM and 64-QAM have very similar constellation patterns, which can be seen in Figure 3.1. Since patterns affect the cumulants significantly, cumulants for both modulation types tend to have closer values. Every little deviation or marginal case may result in a wrong classification.



a) QPSK histogram **b)** 64-QAM histogram

Figure 3.2. 64-QAM and QPSK symbol histograms. Number of total symbol is 2000.

3.2. How to Improve the Performance

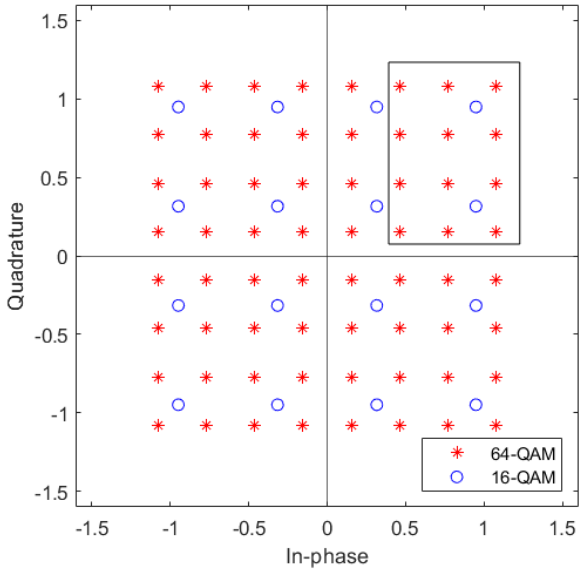
It can be seen from the simulation results that even though we can improve the QAM classification performance, with current features, it is not possible for the system to be completely error-free and in this section, this issue will be addressed and a new solution will be proposed.

3.2.1. Narrower Region of Interest

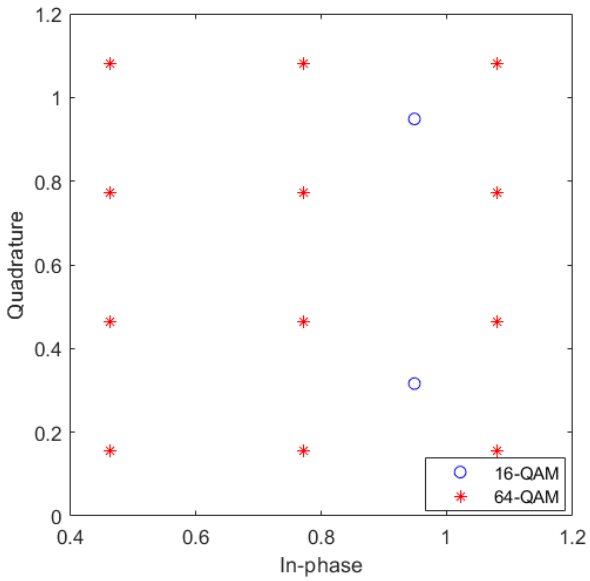
It is already known that cumulants give information about the general pattern of a signal on the 2-D plane. Since all BPSK, QPSK, and 8-PSK constellations have significantly different 2-D patterns, they are easily classified correctly even in low SNR values. The problem between 16-QAM and 64-QAM is, as it's been said before, they have very similar patterns on the 2-D plane, especially with the presence of noise.

If they seem identical in the presence of noise, looking at the plane from another angle may resolve this issue. In this thesis, it is proposed that looking at a narrower region of interest could make the system more separable. This narrower region should be selected such that both modulation types should have distinct patterns in that region. This narrower region is specified in Figure 3.3. Since both modulation types'

patterns are more distinct in this region, their cumulant values are also expected to be more distinct, which leads to better separability.



a) 16-QAM & 64-QAM



b) The specified region

Figure 3.3. Visualization of the narrower region.

The algorithm works still the same; however, the cumulant calculation only includes the symbols inside this specific region. The results can be seen in Table 3.2. For the high SNR values, the results are promising. However; in the low SNR range, the classifier turns into a biased classifier. In other words, the classification process favors one modulation over the other one. In this case, the classifier favors 64-QAM.

Table 3.2. Confusion matrices of the narrower region method.

		BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
6 dB SNR	**BPSK**	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	2	4998
	64-QAM	0	0	0	8	4992
		BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
14 dB SNR	**BPSK**	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	4132	868
	64-QAM	0	0	0	4	4996
		BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
20 dB SNR	**BPSK**	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	5000	0
	64-QAM	0	0	0	0	5000

Biased classifiers are not good, even if it is only in low SNR range. However, its high SNR performance could be very useful. Since the first classifier has a problem in the high SNR values, which we called the error floor, and the second classifier has a problem in low SNR range, which we called biased classifier, both classifiers can be

combined to produce a good classifier. However, high SNR 64-QAM signals should be separated before.

3.2.2. Separating High SNR Signals From Low SNR Signals

In this step, we have 4 types of signals: Low-SNR 16-QAM signals, low-SNR 64-QAM signals, high-SNR 16-QAM signals, and high-SNR 64-QAM signals. High SNR 16-QAM signals are already separated from the other three by using the specific region classifier. In this part, high-SNR 64-QAM signal should also be separated so that the low SNR part of the first classifier and the high SNR part of the second classifier could be combined.

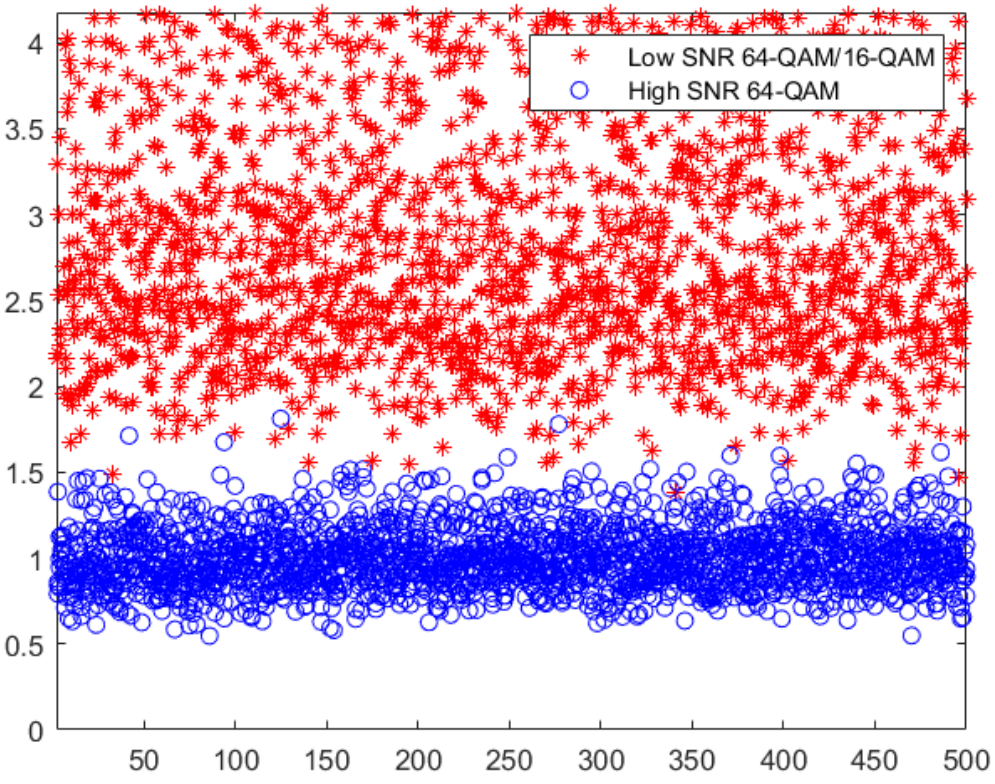


Figure 3.4. Separability after CoV.

One metric to separate the two signal groups is coefficient of variance (CoV). First of all, the received signal is demodulated as if the received signal is 64-QAM. Then, to measure the distribution of the symbols, CoV is utilized. CoV is a very easy and effective algorithm to measure the distribution. The mathematical expression of CoV is given as

$$CoV = \frac{\sigma}{\mu}, \tag{3.2}$$

where σ is the standard deviation of a signal and μ is the expected value of a signal. Since the patterns are similar to each other and to ensure further separability, only the symbols that are very close to the modulation points are included in the calculation. As a result, the separability can be seen in Figure 3.4.

3.3. General Proposed Algorithm

First of all, initial cumulants and KNN are used to make a decision that is called the first decision. If this decision results in a modulation type that is neither 16-QAM nor 64-QAM, the output will be the first decision. If the decision is either 16-QAM or 64-QAM, the algorithm will move on to the second step.

In this step, only the symbols that are inside the specific region are considered. The procedure is still the same as the first step. After computing the cumulants, KNN is used to decide if the data belongs to 16-QAM or 64-QAM. It is already explained that if the decision is 16-QAM, the modulation type is 16-QAM. If the decision is 64-QAM, the algorithm will move on to the third step.

In the third step, the algorithm basically decides whether the signal is in the high SNR region or the low SNR region. First, the algorithm assumes that all the signals are 64-QAM signals and demodulates them accordingly. Then, the CoV metric is calculated for each signal. If any CoV value is below the pre-determined threshold, the algorithm classifies it as a 64-QAM signal, since it means that the signal is in the

high SNR range. If it exceeds the threshold, the algorithm decides that the signal is in the low SNR range and classifies the signal according to the output of the first decision.

As a result, this algorithm combines the good properties of two classifiers into a good classifier. The block diagram of the algorithm can be seen in Figure 3.5.

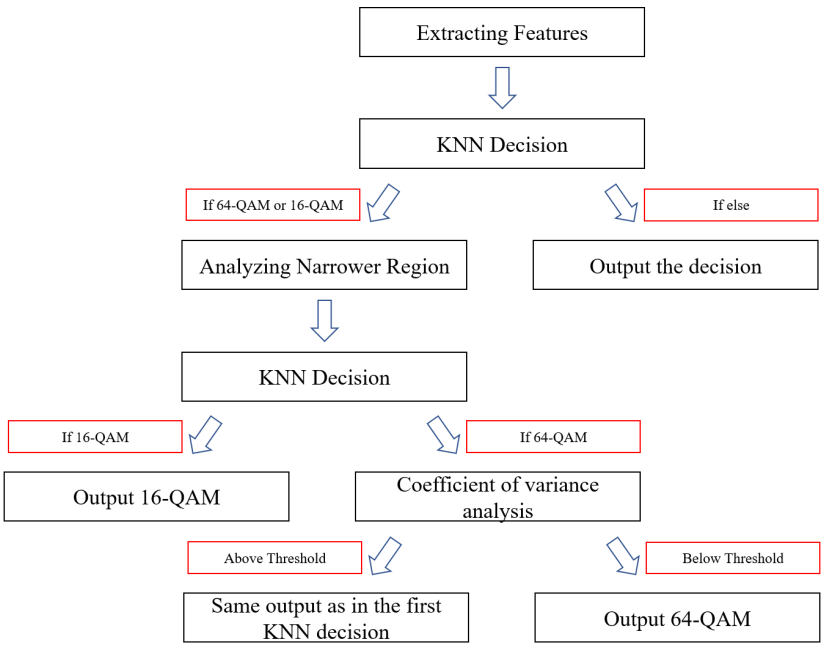


Figure 3.5. Block diagram of the proposed feature-based classifier.

3.4. Results

The results can be seen in Table 3.3. The results are encouraging, because it eliminates the error floor on the high SNR region. However, It would be better to have a lower error rate on the middle SNR regions. As a result, even though the results are promising and encouraging, especially at the high SNR region, the result is not perfect and can still be improved further.

Table 3.3. Confusion matrices of the proposed method.

4 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	3667	1333
	64-QAM	0	0	0	1395	3605
12 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	4282	718
	64-QAM	0	0	0	731	4269
20 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	5000	0	0	0	0
	QPSK	0	5000	0	0	0
	8-PSK	0	0	5000	0	0
	16-QAM	0	0	0	5000	0
	64-QAM	0	0	0	0	5000

4. AMC IN THE PRESENCE OF CARRIER PHASE OFFSET

In this chapter, we will search for a solution when carrier phase offset (CPO) affects the signal on top of AWGN.

4.1. CPO Presence

In the presence of CPO, the whole complex 2-D plane rotates around the origin. However, since the rotation is the same for every symbol, the overall pattern is preserved. Therefore, the challenge of classification does not get very hard. The problem in the presence of CPO is that, for every rotation amount, new features must be introduced to the classifier. As a result, computation gets significantly expensive.

In the light of this problem, improving efficiency is as important as improving performance. Therefore, in this section, the focus will be on improving the efficiency instead of improving the performance and deep learning tools will be used to perform classification. Among deep learning networks, convolutional neural network (CNN) is selected due to its ability to perform classification when 2D data is used.

The challenge is to design a classifier for a signal that is affected by the CPO without using CPO affected signals as the training data. Excluding CPO affected signals from the training dataset would greatly improve the efficiency of the classifier.

4.1.1. Signal Model

Since new effects are introduced to the signal, the signal model changes compared to the signal model in the previous section. The new signal model contains both AWGN

and CPO. Therefore, the received signal can be modeled as

$$r[n] = e^{-j2\pi\theta_o} x[n] + g[n], \tag{4.1}$$

where $r[n]$ is the received signal, $x[n]$ is the symbol vector, $g[n]$ is the AWGN component and θ_o is the CPO value. Here, θ_o is constant. Therefore, each symbol is equally affected by the CPO.

4.1.2. Training Dataset

The CNN classifier aims to classify CPO affected signals, but the training dataset only contains the signals that are affected by the AWGN. Therefore, the signal model for the training dataset consists of received signals of the form

$$r[n] = x[n] + g[n], \tag{4.2}$$

where $r[n]$ is the received signal, $x[n]$ is the symbol vector, and $g[n]$ is the AWGN component.

For every even SNR value from 0 to 20 dB, 1000 signals that consist of 1000 symbols are generated, which results in 11000 samples for one modulation type. Like the discussion provided in the previous section, 5 target modulation classes are considered in this section: BPSK, QPSK, 8-PSK, 16-QAM, and 64-QAM. Therefore, there are a total of 55000 signals for the training dataset.

4.1.3. Training Dataset Generating Procedure

Since CNN shows its ability best when more than 1-dimensional, preferably 2-D, dataset is fed to the system, instead of cumulants, a new dataset needs to be generated for the CNN approach. Fortunately, modulated signals are complex numbers that can be represented on the 2-D plane.

The signal generating procedure can be summarized in four steps.

(i) First, each complex symbol in a modulated signal vector is marked on the 2-D complex plane. Since each signal has 1000 symbols in it, there will be 1000 marks on the 2-D complex plane.

(ii) In the second step, an $n \times n$ grid that covers all the symbols on the 2-D complex plane is drawn. This results in a group of grids with equal size.

(iii) In the third step, the number of symbols in each grid is measured and noted in their respective grids. Thus, a 2-D histogram of symbols is calculated.

(iv) In the fourth step, this $n \times n$ histogram is converted into a 2-D 8-bit gray-scale image by normalizing the histogram. Each grid becomes a pixel and normalization makes the highest value in the histogram 255 and makes the lowest value 0. The rest of the values are scaled accordingly.

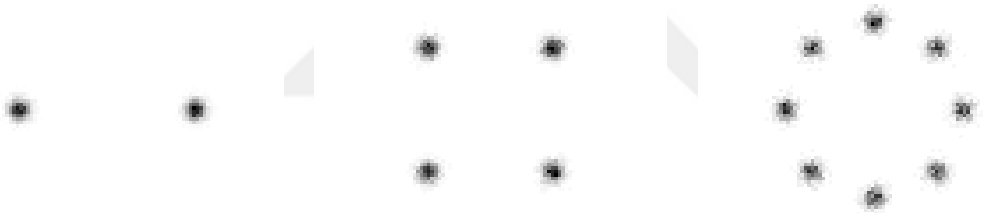
This process is performed for each signal in the training dataset which results in 55000 8-bit gray-scale images. Example images from each modulation type can be seen in Figure 4.1.

4.1.4. CNN Architecture

The AMC problem, in this case, is a relatively simple task, because complex features are not needed to be extracted from the images. Therefore, a simple CNN architecture is used in this case. In [46], a simple but effective architecture in classification problems is introduced, which is called VGG-16. In this section, a very similar architecture to the VGG-16 is used.

The input layer to the CNN architecture is a convolutional layer that has an input size of $48 \times 48 \times 1$. This convolutional layer has 8 3×3 filters. Another convolutional layer follows the input layer with another 8 3×3 filters. The next layer is a max-pooling

layer with a 2×2 window. Following the max-pooling layer, another 2 consecutive convolutional layers are used in the architecture with 16 3×3 filters. Again, a 2×2 max-pooling layer is used after the consecutive convolutional layers. After the second max-pooling layer, 3 consecutive convolutional layers with 32 3×3 filters are used. The third 2×2 max-pooling layer follows the third string of convolutional layers. This time, a flatten layer is used in the architecture, which converts the 3-D data to 1-D data. After that, 2 dense layers with 70 neurons each are added to the architecture. Finally, the output layer, which is a dense layer with 5 neurons, is added to the architecture. 5 neurons are used for the output layer, because the classifier is a 5-class classifier. The architecture can also be seen in Figure 4.2. This network will be called the Cartesian network in the rest of the thesis.



a) BPSK **b)** QPSK **c)** 8-PSK



d) 16-QAM **e)** 64-QAM

Figure 4.1. Samples from the cartesian dataset.

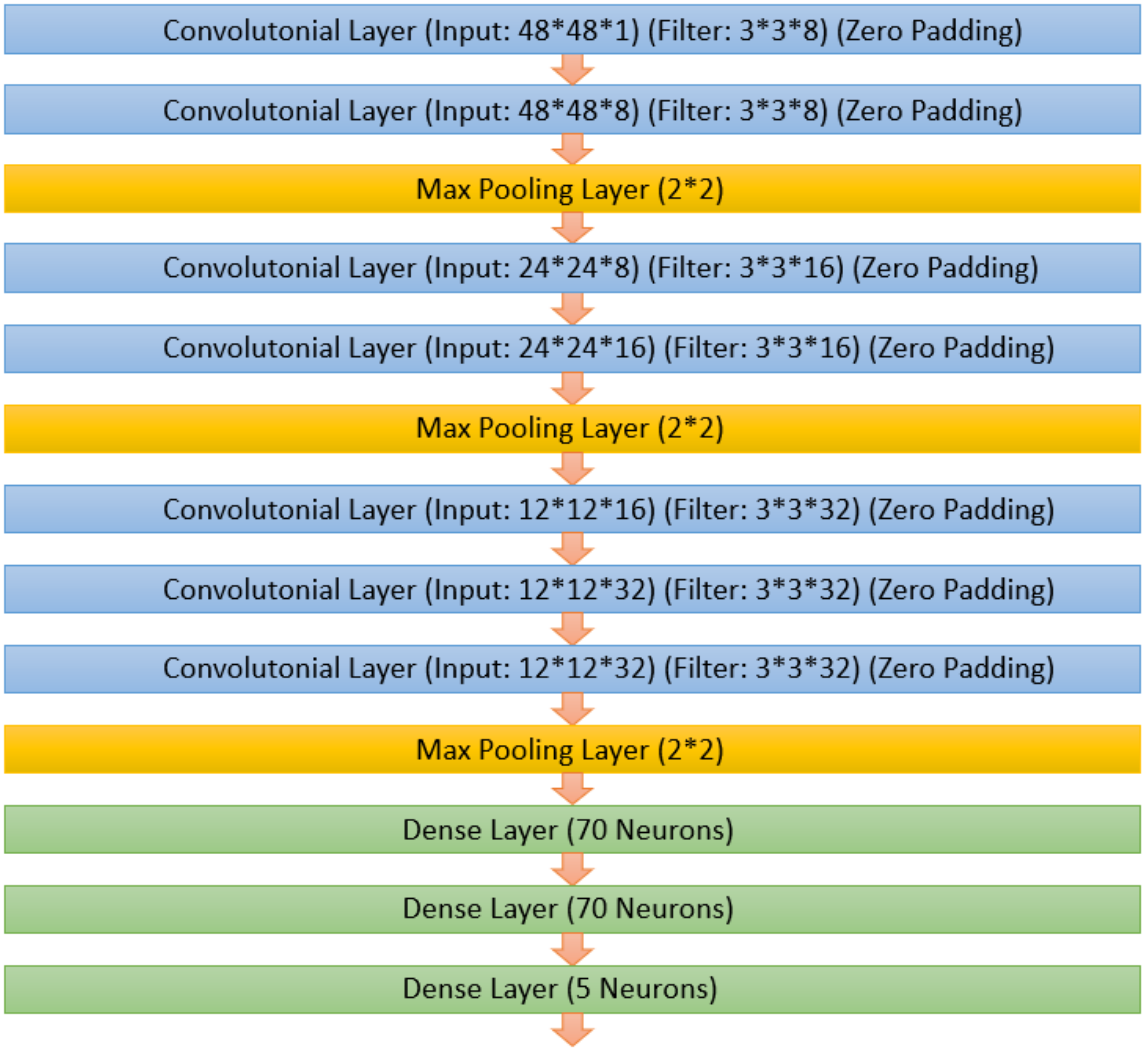


Figure 4.2. CNN architecture for the classification of CPO affected signals with a max-pooling layer.

4.1.5. Results

The results can be seen in Table 4.1. Each test sample has a CPO, θ_o, that is uniformly distributed between 0 and 2π. The results are poor for the Cartesian network. Therefore, feeding the complex data directly to the network does not work clearly and some changes are needed to be done on the training dataset.

Table 4.1. Confusion matrices of Cartesian network with max-pooling layer.

4 dB SNR		BPSK	QPSK	8-PSK	16-QAM	64-QAM
	BPSK	241	227	0	7	0
	QPSK	0	422	576	2	0
	8-PSK	0	13	984	2	1
	16-QAM	0	27	430	346	197
	64-QAM	0	18	293	286	403
12 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	231	257	0	2	1
	QPSK	0	417	578	4	1
	8-PSK	0	26	970	3	1
	16-QAM	0	12	392	354	242
	64-QAM	0	18	306	102	574
20 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	247	309	442	1	1
	QPSK	0	552	445	1	2
	8-PSK	0	19	975	5	1
	16-QAM	0	18	238	359	385
	64-QAM	0	10	301	28	661

4.2. Polar Coordinates and CPO

CNNs are robust to the translational effects on the 2-D plane due to the nature of convolution. Convolution results would not change in the presence of translation, the results would only shift. Thus, the resulting 2-D image would only be the shifted version of the result before the translation effect. However, CPO has a rotational effect on the dataset and CNN does not have the ability to compensate for it. As a result, the results would differ significantly. Therefore, either the network itself or the dataset should compensate for this rotational effect. For the network itself to compensate for

it, rotated images should be fed to the network which, contradicts with the challenge. Therefore, the dataset itself should compensate for this effect.

Thus, we propose using the polar coordinates instead of the Cartesian coordinates. CPO causes rotation in the Cartesian coordinates but rotation in the Cartesian coordinates corresponds to translation in the polar coordinates.

To understand this effect, both coordinate systems should be analyzed. In the Cartesian coordinate system, (x, y), the real part of the modulated signal is represented by x and the imaginary part of the modulated signal is represented by y. Representation of the polar coordinates, (r, θ), in terms of the Cartesian coordinates is

$$r = \sqrt{x^2 + y^2}, \tag{4.3}$$

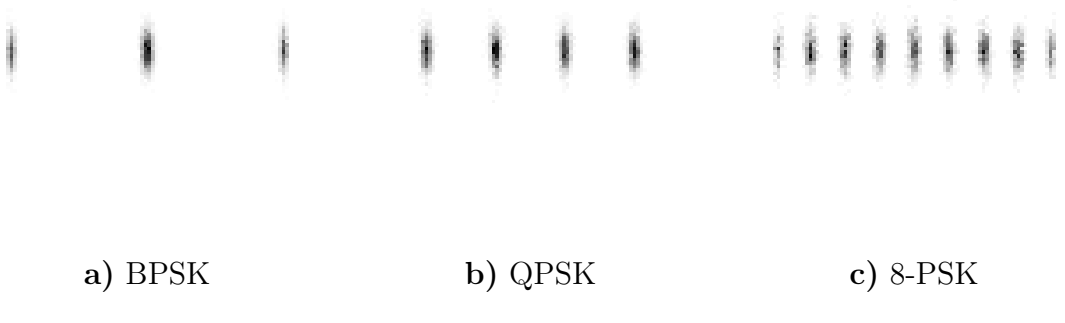
$$\theta = tan^{-1}(y/x), \tag{4.4}$$

where r is the length and θ is the angle of the complex modulated signal.

The rotational effect changes both x and y values in the Cartesian coordinates, but, in the polar coordinates, it only changes the θ values. Therefore, it can be seen that rotational effects correspond to the translational effects in the polar coordinates.

4.2.1. Methodology

The methodology is the same as the one presented in the previous section. The only difference is that before generating the dataset, all the signals are converted into polar coordinates instead of Cartesian coordinates. Example images from the polar dataset can be seen in Figure 4.3.



a) BPSK **b)** QPSK **c)** 8-PSK



d) 16-QAM **e)** 64-QAM

Figure 4.3. Samples from the Cartesian dataset.

4.2.2. Results

The results can be seen in Table 4.2. Each test sample has a CPO, θ_o, that is uniformly distributed between 0 and 2π.

Table 4.2. Confusion matrices of the polar network with max-pooling layer.

4 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	330	401	131	104	34
	QPSK	0	392	601	7	0
	8-PSK	0	1	998	0	1
	16-QAM	0	5	47	306	642
	64-QAM	0	1	10	139	850

Table 4.2. (cont.)

		BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
12 dB SNR	**BPSK**	390	0	610	4	0
	QPSK	669	240	91	0	0
	8-PSK	0	0	1000	0	0
	16-QAM	0	0	158	469	373
	64-QAM	0	0	306	10	684
		BPSK	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
20 dB SNR	**BPSK**	376	574	0	50	0
	QPSK	585	394	3	18	0
	8-PSK	0	0	1000	0	0
	16-QAM	5	2	56	937	0
	64-QAM	0	0	96	0	904

The results are better than the previous results but they are still not very good. Therefore, the network should be more robust to the translational effects, which CNN is known to be.

4.3. Global Average Pooling

Even though the convolution results in a shifted outcome, the poor results in the previous section show that the classifier is also sensitive to the shifted results. The problem arises at the transition between convolutional layers and dense layers. If the network manages to transfer the information from convolutional layers to dense layers without losing information pertaining to the translation effect, then the classifier would be expected to work. To compensate for this effect, a global average pooling layer is added after the third string of convolutional layers instead of a max-pooling layer. The global average pooling layers take the average of the whole plane. Therefore, shifting does not change the outcome of the convolutional layer group. As a result, the new CNN architecture can be seen in Figure 4.4.

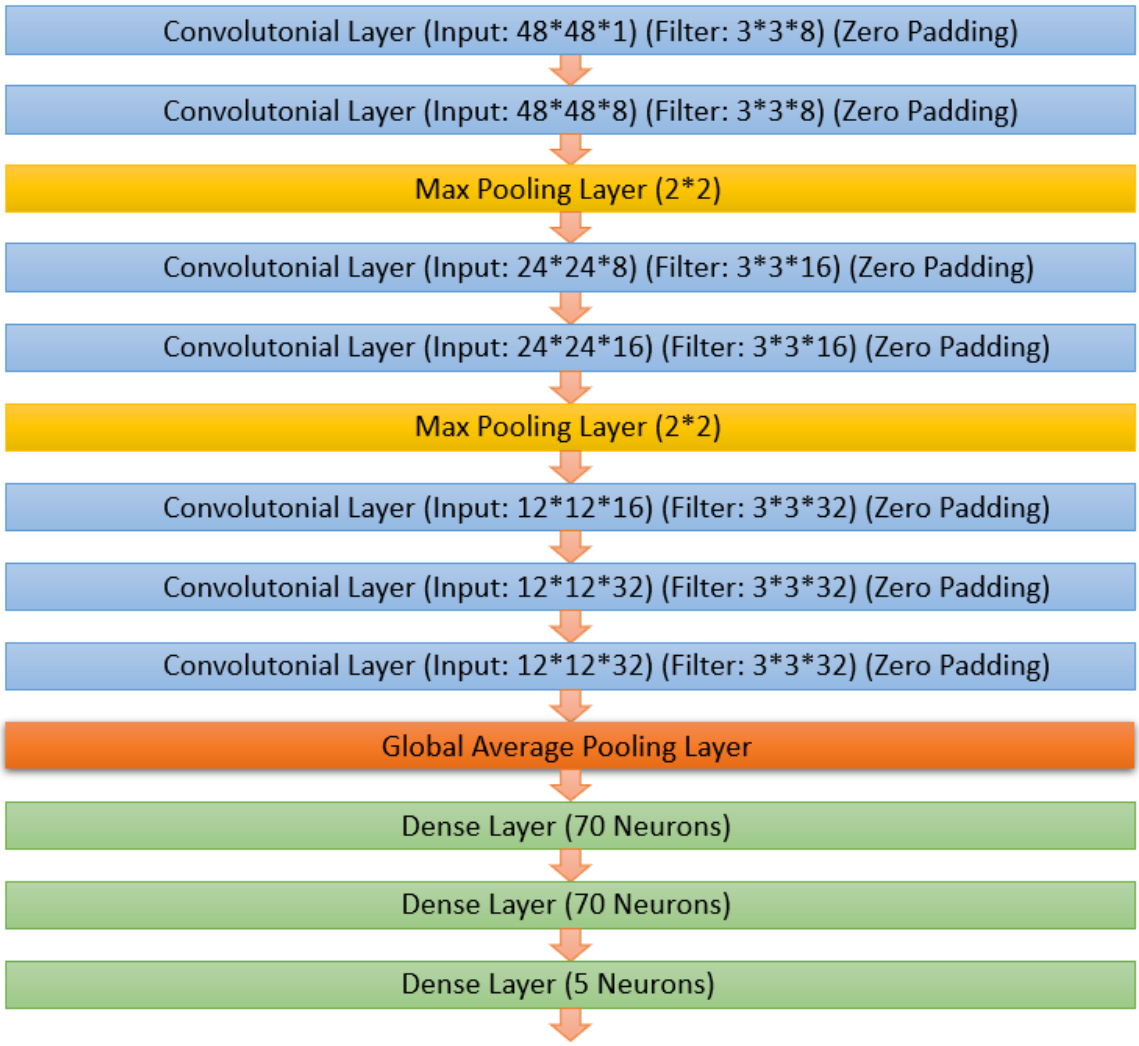


Figure 4.4. CNN architecture for the classification of CPO affected signals with a global average pooling layer.

4.3.1. Results

The results can be seen in Table 4.3. Each test sample has a CPO, θ_o, that is uniformly distributed between 0 and 2π. The results are promising and show that the combination of the polar dataset and global average pooling layer grants to the network the ability of good classification performance without using a CPO affected dataset.

Table 4.3. Confusion matrices of the polar network with a global average pooling layer.

4 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	1000	0	0	0	0
	QPSK	0	643	342	15	0
	8-PSK	0	0	1000	0	0
	16-QAM	0	13	159	420	408
	64-QAM	0	1	23	111	865
12 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	1000	0	0	0	0
	QPSK	0	1000	0	0	0
	8-PSK	0	0	1000	0	0
	16-QAM	0	0	0	821	179
	64-QAM	0	0	0	0	1000
20 dB SNR		**BPSK**	**QPSK**	**8-PSK**	**16-QAM**	**64-QAM**
	BPSK	1000	0	0	0	0
	QPSK	0	1000	0	0	0
	8-PSK	0	0	1000	0	0
	16-QAM	0	0	0	1000	0
	64-QAM	0	0	0	0	1000

5. AMC UNDER THE PRESENCE OF CARRIER FREQUENCY OFFSET

Since the signals do not lose their characteristics under the CPO effect, the signals are still very easy to identify. Their complex 2-D plane representation only rotates around the origin which preserves the pattern. However, the CFO effect is harder to deal with, since all the symbols on the complex 2-D plane rotate around the origin by a different amount. The only thing that is constant under the presence of CFO is the difference of rotation amount between two consecutive symbols.

If the received signal is only affected by the CPO and AWGN, it is easier to analyze the signal if the modulation type is classified correctly due to preserved characteristics. However, it is harder to analyze the signal if the signal is under the presence of CFO. Therefore, estimating the CFO amount to recover the signal is as important as classifying the signal correctly.

Considering the effects of CFO, in this chapter, we will try to estimate the CFO amount of the received signal and then, perform classification. We will use deep learning tools to perform regression and classification. Finally, BPSK, QPSK, and 8-PSK are chosen as the modulation classes.

5.1. Signal Model

The signal model evolves further compared to the CPO case. The signal model with the added effect of CFO is defined as

$$r[n] = e^{-j2\pi(f_o nT + \theta_o)} x[n] + g[n], \tag{5.1}$$

where $r[n]$ is the received signal, $x[n]$ is the symbol vector, $g[n]$ is the AWGN component, θ_o is the CPO amount, and f_o is the CFO amount. Here, f_o is constant, but

since it is multiplied with n, it affects every signal differently.

5.2. CFO Estimation

In this section, we will try to estimate the CFO amount of the received signal.

5.2.1. Training Dataset

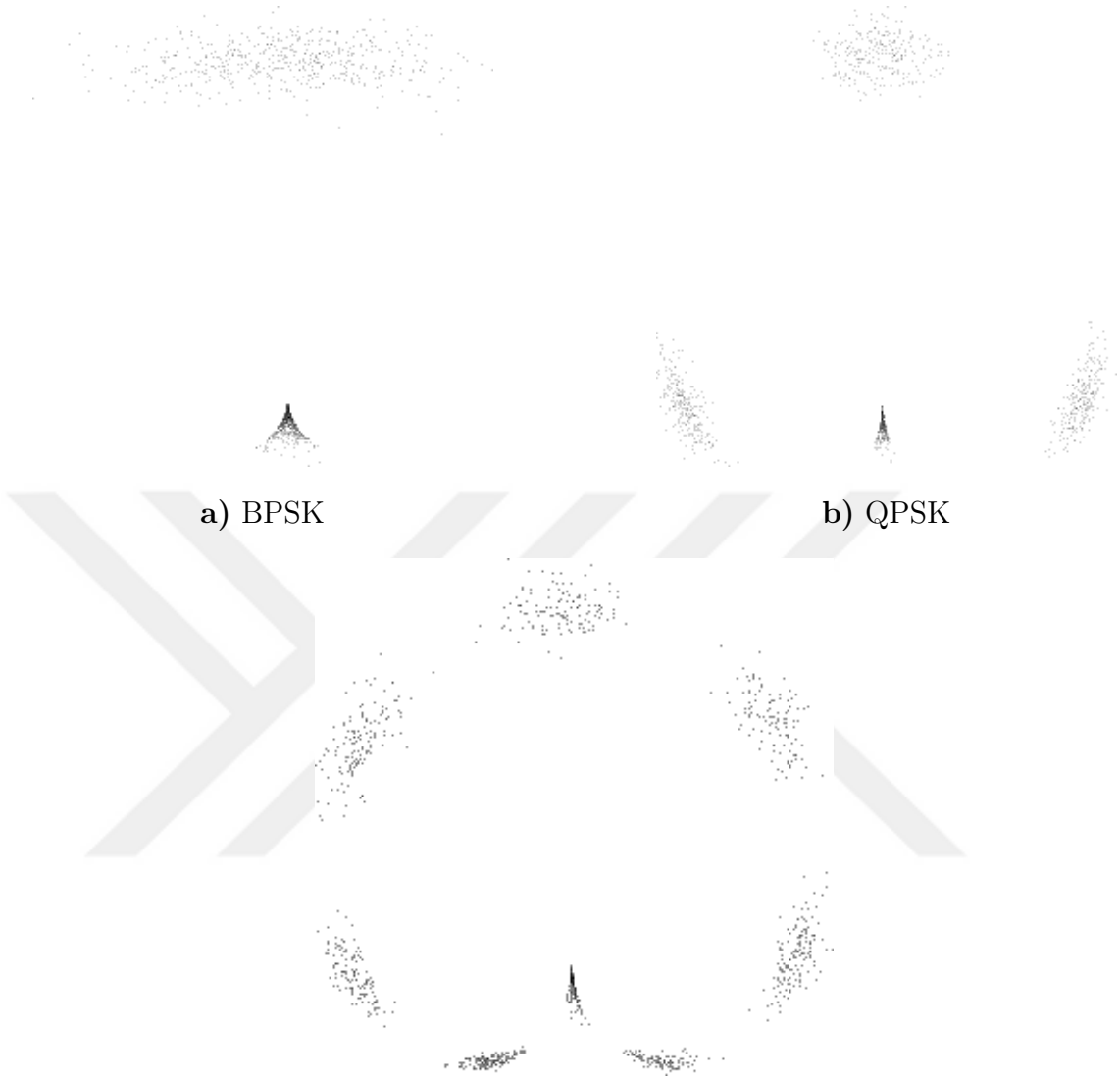
In this part, it is not possible to use the complex 2-D plane representation to generate the dataset, unlike in the case of CPO. Therefore, a new representation that preserves the characteristics of the signals is required to be used under the presence of CFO. Since relative rotation between consecutive symbols remains the same, differences between consecutive symbols are used to create a new 2-D representation. The relative angle between two consecutive symbols, θ_f, and the distance between two consecutive symbols, d_f, are used to create the axes of the new 2-D representation. The mathematical expression of the (d_f, θ_f) is given as

$$d_f = |r[n+1] - r[n]|, \tag{5.2}$$

$$\theta_f = \tan^{-1}\left(\frac{Im\{r[n+1] - r[n]\}}{Re\{r[n+1] - r[n]\}}\right). \tag{5.3}$$

As a result, a more stable 2-D representation of the data with its own characteristics can be obtained.

Generating the training dataset is very similar to the procedure that is used for the CPO case. Only the new 2-D representation is used under the CFO case instead of a Cartesian or polar representation. For every even SNR value from 0 to 20 dB, 2000 signals that consist of 1000 symbols each are generated, which results in 22000 samples for one modulation type. 3 modulation types are considered in this part. As a result, 66000 8-bit gray-scale images are generated for the training dataset. Samples from this dataset, which we call differential dataset in the rest of the thesis, are given in Figure 5.1.



a) BPSK **b)** QPSK

c) 8-PSK

Figure 5.1. Samples from the differential dataset.

5.2.2. CNN Architecture

In this part, a very similar CNN architecture that is used in the previous section is used. However, since this problem is a regression problem rather than a classification problem, some changes are required. The most notable change is using a dense layer with only 1 neuron as an output layer, since only a single output is needed. Robustness to translational effect is also not important in this case. Therefore, a max-pooling layer is used instead of a global average layer for further sensitivity. The CNN architecture

can be seen in Figure 5.8. The sensitivity is important in this case, since even the tiniest difference could affect the signal significantly. Therefore, depending on the desired sensitivity, the input size may be re-adjusted. For higher sensitivity, for instance, using a larger input size is more reasonable. The trade-off is, the larger input size is also computationally more expensive.

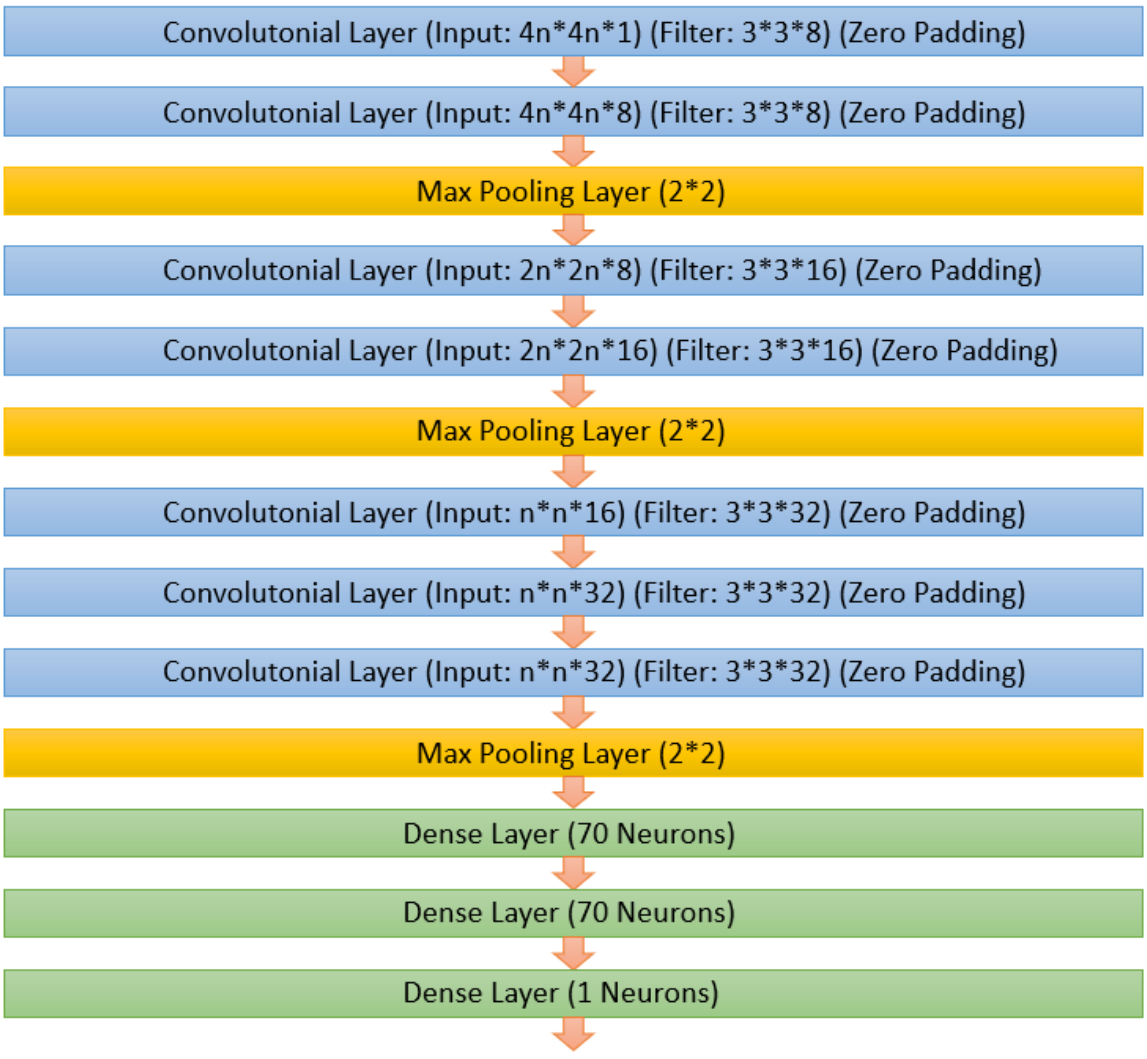
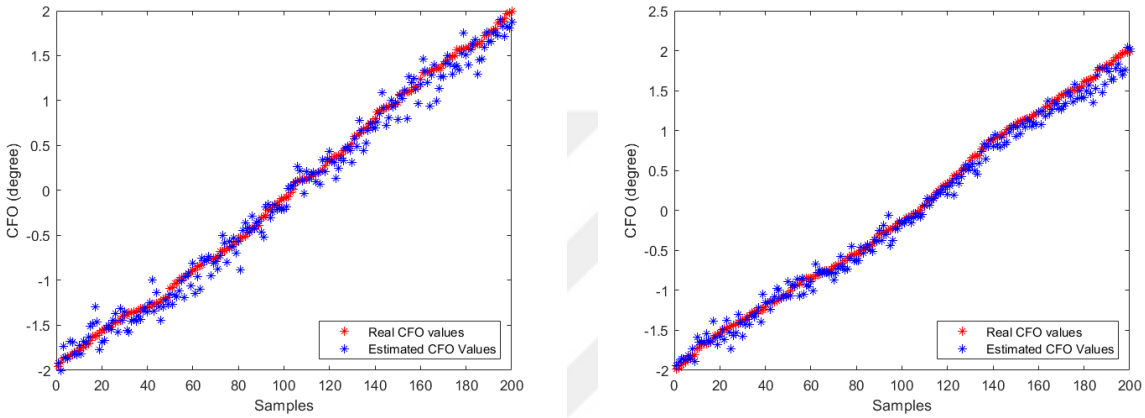


Figure 5.2. CNN architecture for the CFO estimation.

5.2.3. Results

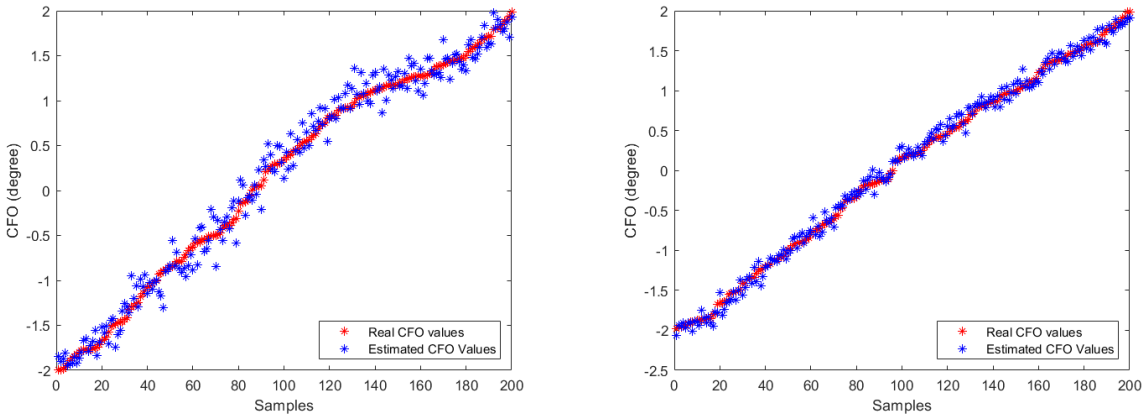
The results are encouraging in this part. When the input size is larger, the network gives very precise results, especially in the high SNR range. When the input

size is not large, the results are still very consistent but not very precise which is expected. The results when the input size is larger are given in Figures 5.3, 5.4, and 5.5. Especially, in the high SNR region, the performance is very good. As the SNR decreases, the results look more scattered but they are still very consistent apart from one or two samples.



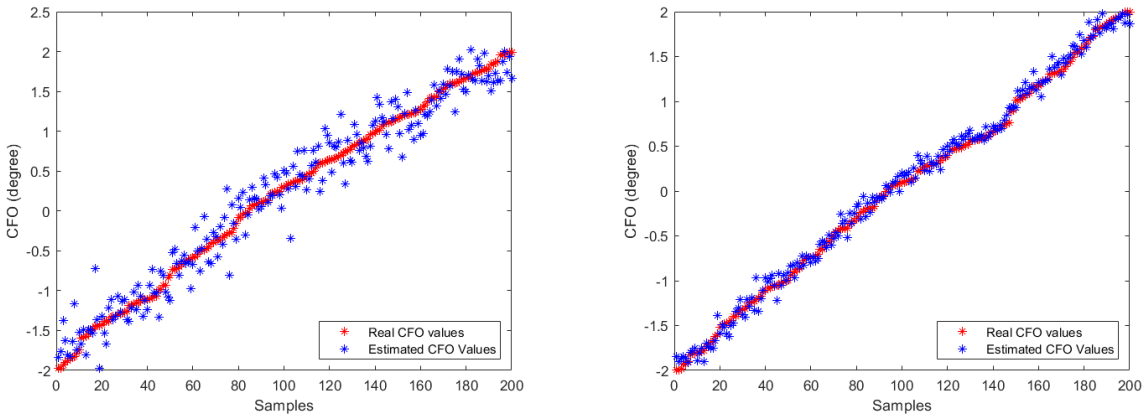
a) 14 dB **b)** 20 dB

Figure 5.3. CFO Estimation comparison of BPSK in terms of SNR. The CFO is uniformly distributed between 2 and -2.



a) 14 dB **b)** 20 dB

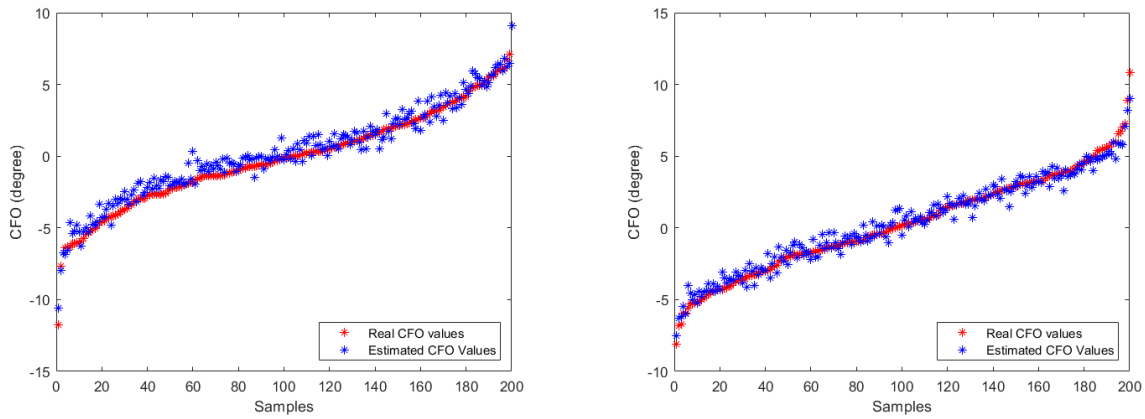
Figure 5.4. CFO Estimation comparison of QPSK in terms of SNR. The CFO is uniformly distributed between 2 and -2.



a) 14 dB **b)** 20 dB

Figure 5.5. CFO Estimation comparison of 8-PSK in terms of SNR. The CFO is uniformly distributed between 2 and -2.

The results when the input size is smaller are given in Figure 5.6. Precision is not very good compared to the other results but it is still very consistent as expected. Therefore, it can be concluded that the proposed network can be utilized to estimate the CFO value if the given signal belongs to BPSK, QPSK, or 8-PSK classes, especially if the signal is in the high SNR region.



a) QPSK **b)** 8-PSK

Figure 5.6. CFO Estimation of QPSK and 8-PSK when the input size is smaller. The CFO is uniformly distributed between 10 and -10.

5.3. Classification

In this section, we will try to classify the signals after minimizing the CFO effect.

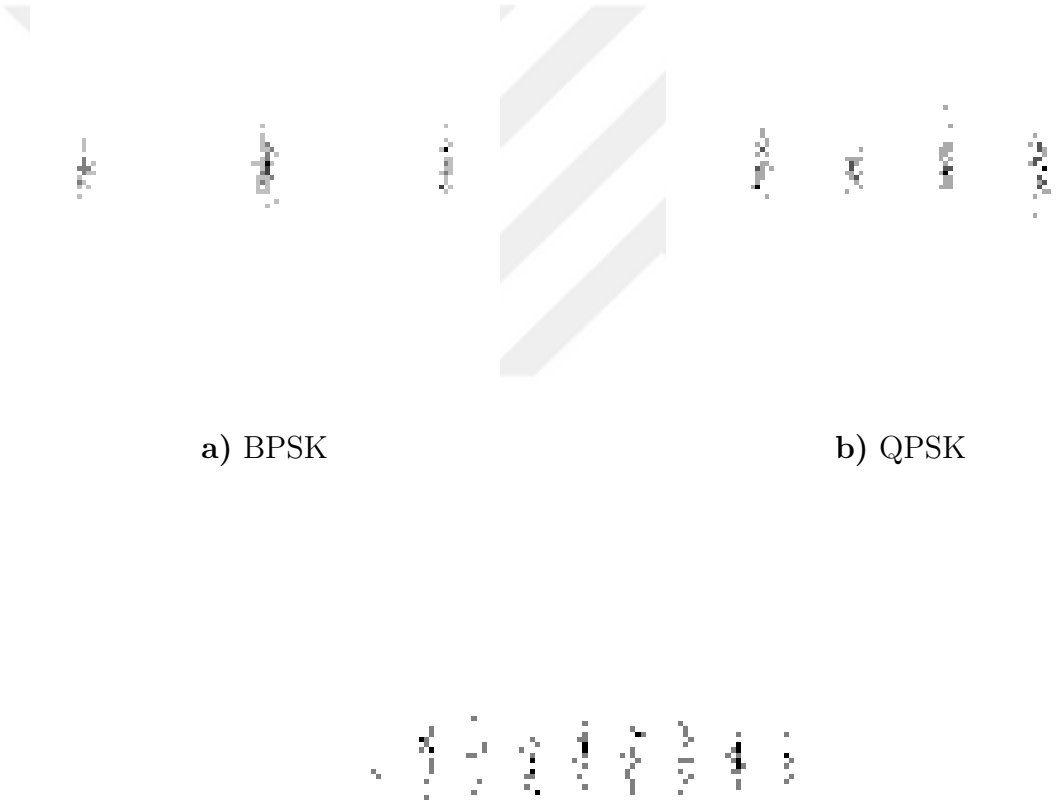
5.3.1. Training Dataset and CNN Architecture

Hypothetically, after eliminating the CFO effect from the received signal, the signal turns into a signal that is only affected by the CPO and AWGN. Therefore, the trained network that is used to classify CPO affected signals can also be used here. However, eliminating the CFO entirely is not very possible and even a minuscule CFO value like 0.1 degrees per symbol may cause the signal to lose its own characteristics after 1000 symbols.

To prevent this issue, only 100 symbols per signal are considered for the test data considering the accuracy of the CFO estimation. This solution comes with its own problems. Since the trained network is trained by using signals with 1000 symbols, the results are not as accurate as expected. Therefore, a new network is needed to be trained.

First of all, a new training dataset needs to be created. This training dataset will be similar to the dataset that is created before the classification of signals under the presence of CPO. For each even SNR value from 0 to 20, 1000 signals are generated for each modulation type. However, this time, each signal has only 100 symbols. The dataset generation procedure is also the same as the procedure of the polar dataset generation. Samples from the new polar dataset are given in Figure 5.7. Since only BPSK, QPSK and 8-PSK are included in the dataset, using 100 symbols is still enough to define these modulation types. However, they are not as dense and smooth as the CPO affected dataset that is used in the previous chapter. Using 100 symbols would not have been enough to define 16-QAM and 64-QAM signals due to their number of distinct symbols.

Second, a CNN architecture needs to be established to classify the received signals. Since the problems are very similar, the same CNN architecture that is used for the CPO case is used here which is given in Figure 4.4. The only difference is that the dense layer has 3 neurons, since the number of modulation classes is 3. Therefore, the network is revised to classify these modulation types. The new CNN architecture can be seen in Figure 5.8.



a) BPSK **b)** QPSK

c) 8-PSK

Figure 5.7. Samples from the new polar dataset.

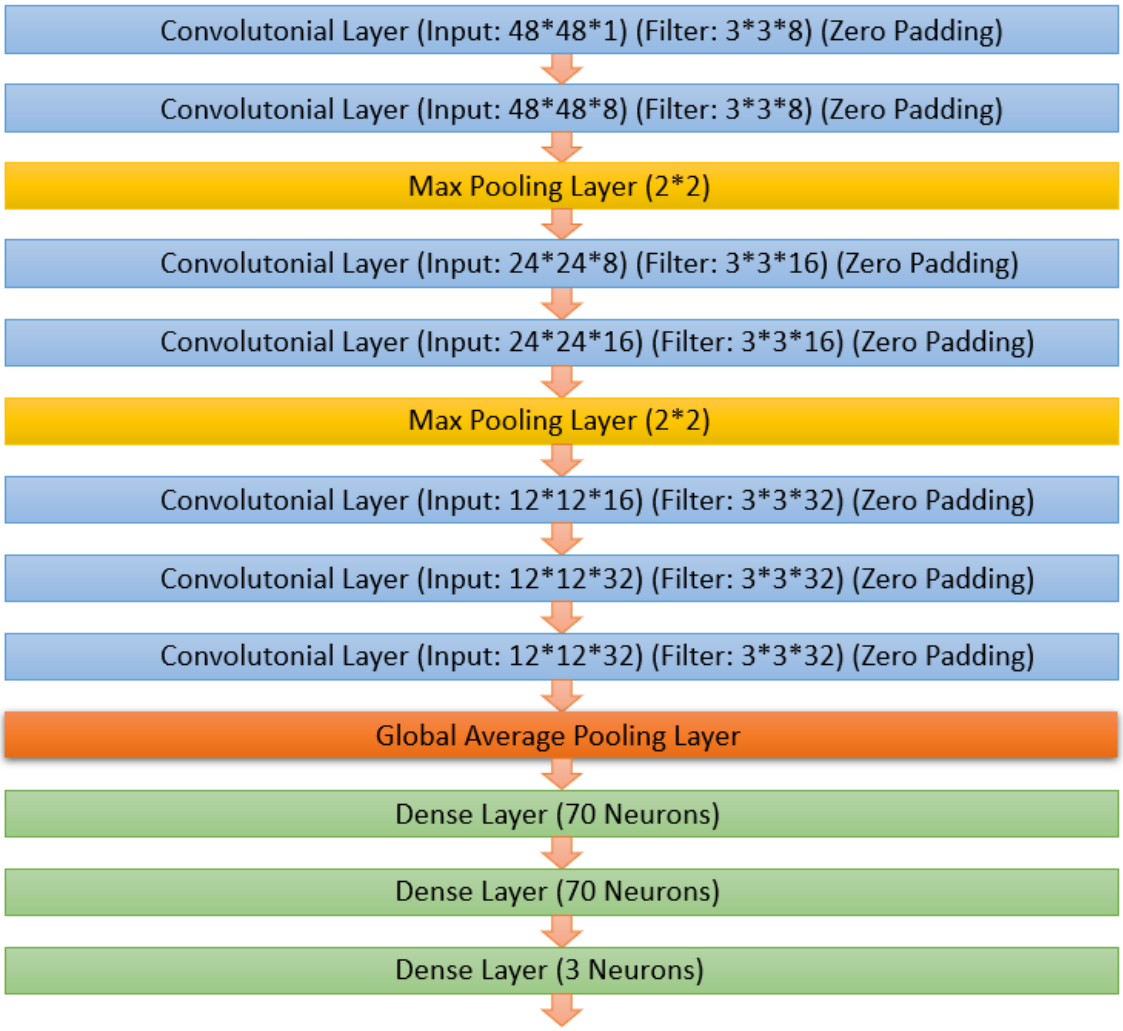


Figure 5.8. CNN architecture for the classification of CFO affected signals.

5.3.2. Results

The results can be seenin Table 5.1. The classification results are also very promising and encouraging. Especially, at the high SNR values, the performance is very good. Clearly, not having both 16-QAM and 64-QAM signals in the classification process also helps. Considering the results that are obtained in the previous chapters, we can conclude that the classification performance decreases with the CFO effect. However, it is expected, since the number of symbols per signal is reduced significantly because of the CFO.

Table 5.1. Confusion matrices of the polar network with CFO affected test data.

8 dB SNR		**BPSK**	**QPSK**	**8-PSK**
	BPSK	2000	0	0
	QPSK	1	1885	114
	8-PSK	12	52	1936
14 dB SNR		**BPSK**	**QPSK**	**8-PSK**
	BPSK	2000	0	0
	QPSK	0	2000	0
	8-PSK	0	1	1999
20 dB SNR		**BPSK**	**QPSK**	**8-PSK**
	BPSK	2000	0	0
	QPSK	0	2000	0
	8-PSK	0	0	2000

To conclude, both regression and classification networks show good performances despite some of the drawbacks, such as reduced number of symbols, CFO effects, etc. However, the performance is still open to improvements. Including other modulation types and improving the classification and regression accuracy could be counted as some of the improvements.

6. CONCLUSION

To summarize, AMC denotes the autonomous process of classification of the modulation types of any given modulated signal. First, we surveyed the various AMC methods that are employed in the literature. The likelihood-based classifiers are one of the first methods that are utilized in the AMC area. The goodness of fit tests, feature-based methods, machine learning-based methods, and deep learning-based methods followed the likelihood-based classifiers in the AMC field. While some of these methods perform classifications blindly, others assume prior knowledge ahead of the classification process. In this thesis, blind AMC methods have been investigated.

The first algorithm that we proposed in this thesis uses feature-based methods and machine learning tools. Since high-order cumulants are a popular choice in the literature, they are chosen as the feature set and KNN is chosen as the machine learning tool. The problem in the feature-based algorithms is the error floor that is encountered during the QAM classification in the high SNR region. Our proposed 3-staged algorithm manages to eliminate this error floor in the high SNR region. However, it performs relatively poorly in the middle SNR region.

We then propose a deep learning-based AMC algorithm under the presence of CPO in Chapter 4. This algorithm focuses on efficiency rather than performance. By using a novel polar coordinate approach, we manage to build a CNN architecture that classifies CPO-affected signals without using any CPO-affected signals in the training phase. The polar coordinate approach allows us to convert the rotation effect that is caused by the CPO, into the translation effect. Additionally, the global average pooling layer helps the network compensate for the translation effect. Therefore, we do not have to use the CPO-affected signals to train the network. As a result, the network performs very well, especially in the high SNR region.

Finally, we propose a deep learning-based AMC algorithm under the presence of CFO in Chapter 5. This algorithm consists of two stages. In the first stage, a network is built to estimate the CFO amount by using regression. The reason behind the CFO estimation is to make it easier to analyze the signal. The results show that the first stage works very well and the network performs with great precision. After mitigating the CFO effect, the second stage comes into play. In the second stage, the same network that is used for the CPO-affected signals is used to perform classification. However, the number of symbols inside each signal is reduced from 1000 to 100 to prevent any adverse effects of CFO. As a result, the classification network performs well. The results are not as good as the results in the previous chapters, but it is expected under the presence of CFO. Overall, the whole algorithm works well to perform classification.

Even though each algorithm that we proposed in this thesis performs well, they are still open to improvement. As feature works, the performance of the proposed feature-based algorithm is relatively weak in the middle SNR region. Another stage that improves the performance in that region may be added to the system. The deep learning-based algorithm in Chapter 4 shows great efficiency but its performance can also be improved. Since the angles of the signals are cyclic, some information can be lost during the transition from 2π to 0. Some kind of cyclic convolution may be added to the network to improve the performance. The deep learning-based algorithm in Chapter 5 also shows great performance, but it only includes m-PSK signals. Therefore, other modulation types may be added to the target modulation classes to expand the coverage of the algorithm.

REFERENCES

1. Huan, C.-Y. and A. Polydoros, "Likelihood Methods for MPSK Modulation Classification", *IEEE Transactions on Communications*, Vol. 43, No. 2/3/4, pp. 1493–1504, 1995.

2. Wei, W. and J. Mendel, "Maximum-Likelihood Classification for Digital Amplitude-Phase Modulations", *IEEE Transactions on Communications*, Vol. 48, No. 2, pp. 189–193, 2000.

3. Chavali, V. G. and C. R. C. M. da Silva, "Maximum-Likelihood Classification of Digital Amplitude-Phase Modulated Signals in Flat Fading Non-Gaussian Channels", *IEEE Transactions on Communications*, Vol. 59, No. 8, pp. 2051–2056, 2011.

4. Ramezani-Kebrya, A., I.-M. Kim, D. I. Kim, F. Chan and R. Inkol, "Likelihood-Based Modulation Classification for Multiple-Antenna Receiver", *IEEE Transactions on Communications*, Vol. 61, No. 9, pp. 3816–3829, 2013.

5. Polydoros, A. and K. Kim, "On the Detection and Classification of Quadrature Digital Modulations in Broad-Band Noise", *IEEE Transactions on Communications*, Vol. 38, No. 8, pp. 1199–1211, 1990.

6. Hong, L. and K. Ho, "BPSK and QPSK Modulation Classification with Unknown Signal Level", *Military Communications Conference*, Vol. 2, pp. 976–980, 2000.

7. Panagiotou, P., A. Anastasopoulos and A. Polydoros, "Likelihood Ratio Tests for Modulation Classification", *Military Communications Conference*, Vol. 2, pp. 670–674, 2000.

8. Massey, F. J., "The Kolmogorov-Smirnov Test for Goodness of Fit", *Journal of the American Statistical Association*, Vol. 46, pp. 68–78, 1951.

9. Conover, W. J., *Practical Nonparametric Statistics*, John Wiley & Sons Inc., New York, USA, 1999.

10. Wang, F. and X. Wang, "Fast and Robust Modulation Classification via Kolmogorov-Smirnov Test", *IEEE Transactions on Communications*, Vol. 58, No. 8, pp. 2324–2332, 2010.

11. Urriza, P., E. Rebeiz, P. Pawelczak and D. Cabric, "Computationally Efficient Modulation Level Classification Based on Probability Distribution Distance Functions", *IEEE Communications Letters*, Vol. 15, No. 5, p. 476–478, 2011.

12. Zhu, Z., M. W. Aslam and A. K. Nandi, "Genetic Algorithm Optimized Distribution Sampling Test for M-QAM Modulation Classification", *Signal Processing*, Vol. 94, pp. 264–277, 2014.

13. Anderson, T. W., "On the Distribution of the Two-Sample Cramer-von Mises Criterion", *The Annals of Mathematical Statistics*, Vol. 33, pp. 1148–1159, 1962.

14. Honda, C., I. Oka and S. Ata, "Signal Detection and Modulation Classification Using a Goodness of Fit test", *2012 International Symposium on Information Theory and its Applications*, pp. 180–183, 2012.

15. Anderson, T. W. and D. A. Darling, "A Test of Goodness of Fit", *Journal of the American Statistical Association*, Vol. 49, pp. 765–769, 1954.

16. Azzouz, E. E. and A. K. Nandi, "Automatic Identification of Digital Modulation Types", *Signal Processing*, Vol. 47, pp. 55–69, 1995.

17. Azzouz, E. E. and A. K. Nandi, "Procedure for Automatic Recognition of Analogue and Digital Modulations", *IEEE Proceedings Communications*, Vol. 143, pp. 259–266, 1996.

18. Nandi, A. K. and E. E. Azzouz, "Automatic Analogue Modulation Recognition",

Signal Processing, Vol. 46, pp. 211–222, 1995.

19. Hong, L. and K. C. Ho, "BPSK and QPSK Modulation Classification with Unknown Signal Level", *Military Communications Conference*, Vol. 2, pp. 976–980, 2000.

20. Hassan, K., I. Dayoub, W. Hamouda and M. Berbineau, "Automatic Modulation Recognition Using Wavelet Transform and Neural Networks in Wireless Systems", *EURASIP Journal on Advances in Signal Processing*, Vol. 2010, p. 42, 2010.

21. Hipp, J. E., "Modulation Classification Based on Statistical Moments", *IEEE Military Communications Conference*, Vol. 2, pp. 20.2.1–20.2.6, 1986.

22. Soliman, S. S. and S. Z. Hsue, "Signal Classification Using Statistical Moments", *IEEE Transactions on Communications*, Vol. 40, pp. 908–916, 1992.

23. Hero, A. O. and H. Hadinejad-Mahram, "Digital Modulation Classification Using Power Moment Matrices", *IEEE International Conference on Acoustics, Speech and Signal Processing*, Vol. 6, pp. 3285–3288, 1998.

24. Swami, A. and B. M. Sadler, "Hierarchical Digital Modulation Classification Using Cumulants", *IEEE Transactions on Communications*, Vol. 48, No. 3, pp. 416–429, 2000.

25. Edinger, S., M. Gaida and N. J. Fliege, "Classification of QAM Signals for Multicarrier Systems", *15th European Signal Processing Conference*, pp. 464–468, 2007.

26. Aslam, M. W., Z. Zhu and A. K. Nandi, "Automatic Digital Modulation Classification Using Genetic Programming with K-Nearest Neighbor", *Military Communications Conference*, pp. 1731–1736, 2010.

27. Benedetto, F., A. Tedeschi and G. Giunta, "Automatic Blind Modulation Recognition of Analog and Digital Signals in Cognitive Radios", *IEEE 84th Vehicular*

Technology Conference, pp. 1–5, 2016.

28. Zhechen, Z., W. A. Muhammad and A. K. Nandi, "Augmented Genetic Programming for Automatic Digital Modulation Classification", *IEEE International Workshop on Machine Learning for Signal Processing*, pp. 391–396, 2010.

29. Zhechen, Z., W. A. Muhammad and A. K. Nandi, "Support Vector Machine Assisted Genetic Programming for MQAM Classification", *10th International Symposium on Signals, Circuits and Systems*, pp. 1–6, 2011.

30. Zhechen, Z., A. K. Nandi and W. A. Muhammad, "Robustness Enhancement of Distribution Based Binary Discriminative Features for Modulation Classification", *IEEE International Workshop on Machine Learning for Signal Processing*, pp. 1–6, 2013.

31. Nandi, A. and E. Azzouz, "Modulation Recognition Using Artificial Neural Networks", *Signal Processing*, Vol. 56, pp. 165–175, 1997.

32. Zhao, Y., G. Ren, X. Wang, Z. Wu and X. Gu, "Automatic Digital Modulation Recognition Using Artificial Neural Networks", *International Conference on Neural Networks and Signal Processing*, Vol. 1, pp. 257–260, 2003.

33. Koza, J. R., *Genetic Programming: On the Programming of Computers by Means of Natural Selection*, MIT Press, Cambridge, 1992.

34. Wong, M. L. D. and A. K. Nandi, "Automatic Digital Modulation Recognition Using Artificial Neural Network and Genetic Algorithm", *Signal Processing*, Vol. 84, pp. 351–365, 2004.

35. Peng, S., H. Jiang, H. Wang, H. Alwageed and Y. Yao, "Modulation Classification Using Convolutional Neural Network Based Deep Learning Model", *26th Wireless and Optical Communication Conference*, pp. 1–5, 2017.

36. Meng, F., P. Chen, L. Wu and X. Wang, "Automatic Modulation Classification: A Deep Learning Enabled Approach", *IEEE Transactions on Vehicular Technology*, Vol. 67, No. 11, pp. 10760–10772, 2018.

37. Ramjee, S., S. Ju, D. Yang, X. Liu, A. E. Gamal and Y. C. Eldar, "Fast Deep Learning for Automatic Modulation Classification", *arXiv:1901.05850*, 2019.

38. Wang, Y., J. Wang, W. Zhang, J. Yang and G. Gui, "Deep Learning-Based Cooperative Automatic Modulation Classification Method for MIMO Systems", *IEEE Transactions on Vehicular Technology*, Vol. 69, No. 4, pp. 4575–4579, 2020.

39. Sills, J. A., "Maximum-Likelihood Modulation Classification for PSK/QAM", *IEEE Military Communications Conference*, Vol. 1, pp. 217–220, 1999.

40. Zhu, Z., A. K. Nandi and W. W. Aslam, "Approximate Centroid Estimation with Constellation Grid Segmentation for Blind M-QAM Classification", *IEEE Military Communications Conference*, pp. 46–51, 2013.

41. Welling, M., "Robust Higher Order Statistics", *Tenth International Workshop on Artificial Intelligence and Statistics*, pp. 405–412, 2005.

42. Minka, T., "A Comparison of Numerical Optimizers for Logistic Regression", *CMU Technical Report*, Vol. 2003, pp. 1–18, 2003.

43. W. S. McCulloch, W. P., "Modulation Recognition Using Artificial Neural Networks", *Bulletin of Mathematical Biophysics*, Vol. 5, pp. 115–133, 1943.

44. Goodfellow, I., Y. Bengio and A. Courville, *Deep Learning*, MIT Press, 2016.

45. Krizhevsky, A., I. Sutskever and G. E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks", *Advances in Neural Information Processing Systems*, Vol. 25, pp. 1097–1105, 2012.

46. Simonyan, K. and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition”, *Computing Research Repository*, Vol. abs/1409.1556, 2015.