University of Strathclyde

Masters Thesis

CNN-Based Signal Modulation Classifier for Shared Spectrum in Cognitive Radio Networks

Author:
Paul Osinowo

Supervisor:
Robert Stewart

A thesis submitted in fulfilment of the requirements for the degree of MSc.

in the

Electronics and Electrical Engineering Department

August 2024

University of Strathclyde Glasgow

Declaration of Authorship

I, Paul Osinowo, declare that this thesis titled, 'CNN-Based Signal Modulation Classifier for Shared Spectrum in Cognitive Radio Networks' and the work presented in it is my own. I confirm that this work submitted for assessment is my own and is expressed in my own words. Any uses made within it of the works of other authors in any form (e.g., ideas, equations, figures, text, tables, programs) are properly acknowledged at any point of their use. A list of the references employed is included.

Signed: Paul Osinowo

Date: 16th August 2024

Abstract

This report examines how future cognitive radio networks could improve spectrum utilization, especially in frequency bands below 6 GHz. These bands are ideal for broadband mobile communication due to their balanced coverage and capacity, yet they are still underutilized. To tackle this issue, spectrum sharing is highlighted as a promising approach. It allows more efficient use of both licensed and unlicensed spectrum by enabling primary and secondary users to coexist. We provide a comprehensive review of the spectrum sharing taxonomy, detailing the various approaches to efficiently manage shared access in these frequency bands. Following this, we focus on the critical challenge of spectrum sensing, an important feature of most spectrum sharing strategies. Our review covers traditional spectrum sensing methods as well as more advanced, intelligent alternatives that reduce the need for expert knowledge of radio signals. The report then narrows its focus to the problem of signal modulation classification, a specific type of spectrum sensing task. Leveraging the strengths of Convolutional Neural Networks (CNNs) in pattern recognition, particularly in the field of image processing, we adapted a CNN architecture to learn directly from raw time-series radio signals. Our primary goal was to achieve high classification accuracy across varying noise conditions. Using the Radio ML Dataset, which includes 24 different modulation techniques across a range of SNR values, we trained four CNN models to investigate the impact of convolutional layer filter size on model accuracy. The results of our investigation are presented and the best performance model with a classification accuracy of 91.6% was used in the development of a deep learning processor for a Xilinx Zynq Ultrascale+ device. We also provide a post-implementation analysis of resource utilization and timing to evaluate the performance of the design.

Acknowledgements

I would like to express my deepest gratitude to my parents for their unwavering kindness and support throughout this project. Their encouragement has been invaluable. I also extend my sincere thanks to my supervisor, Professor Robert Stewart, for his exceptional guidance and mentorship, which have been crucial to the successful completion of this work. Finally, I am grateful to Andrew MacLellan, Douglas Allan and the team at the Strathclyde Software Defined Radio (StrathSDR) Lab for their insightful guidance and collaboration, which significantly boosted my confidence to embark on this project and further contributed to the outcomes of the project.

Contents

List of Figures

List of Tables

Dedicated to God Almighty

Chapter 1

Introduction

1.1 Internet Growth Trend

Since the birth of the information age, there has been a steady increase in the number of devices that communicate via the internet. According to a survey by the McKinsey Global Institute, over two billion people are connected to the internet, and commercial activities conducted online amount to up to $8 trillion each year [9].

Beyond mobile phones, various industries, including civilian, utility, production, and military fields, rely on the internet for effective day-to-day communication. The internet supports essential services, from the wireless microphones and cameras used in broadcasting and entertainment to the radar systems ensuring air traffic safety. It is also integral to controlling road traffic, managing utilities, and enabling satellite services for TV, GPS, and weather forecasts. Furthermore, the internet is vital for emergency services and military operations, making it a key part of our daily lives and safety [13]

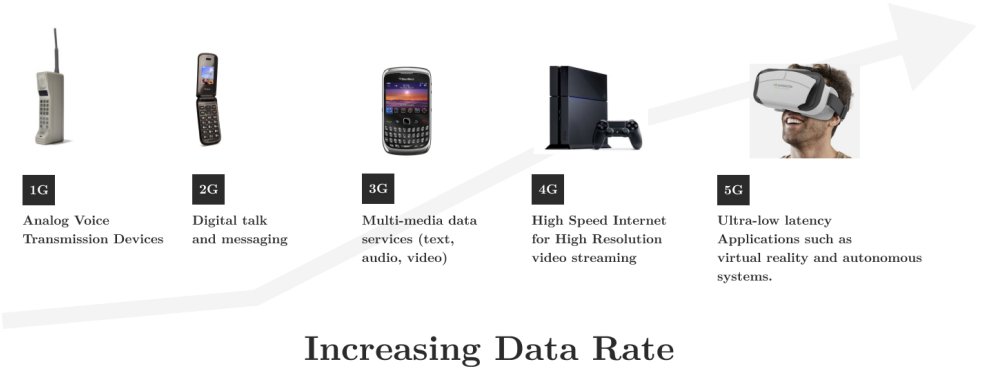

FIGURE 1.1: Internet growth trend

A vast number of communications over the internet are backhauled by a wireless channel composed of electromagnetic (EM) waves. Typically, the portion of the EM spectrum responsible for radio communications ranges from the 30 MHz low band up to 60 GHz millimeter waves (mmWave). This is a finite resource that is highly regulated by national bodies like Ofcom in the United Kingdom and international bodies like the ITU. Regulatory bodies segment portions of the radio spectrum and license these frequency bands to various industries and mobile network operators (MNOs). Therefore, devices are limited to only the portion of the spectrum their network is licensed to operate. The number of connected devices and the complexity of applications running on these devices continue to rise, resulting in high data rates and ultra-low latency requirements that put pressure on existing networks.

1.2 Wireless Network Advancement

Since the development of the first fully digital cellular technology, GSM (2G), which offered data rates of up to 100 kbps [6], network providers and the wireless communications community have continued to devise ways to meet the ever-growing demand for faster multimedia communications. This led to the continuous development and improvement of wireless communication standards, resulting in 3G WCDMA with UMTS technology, which included voice and multimedia capabilities with relatively faster data rates compared to 2G technology. Following that, the fourth generation of wireless communications (4G LTE), developed by the Third Generation Partnership Project (3GPP), used Long Term Evolution (LTE) technology, offering even higher data rates up to several million bits per second (Mbps). It also featured breakthrough modulation techniques like orthogonal frequency division multiplexing (OFDM), which enabled more efficient use of the radio spectrum, operating at high data rates of up to 75 Mbps uplink (UL) and 300 Mbps downlink (DL) [6]. A faster variant, LTE Advanced, was also developed, offering up to three times faster data rates [6].

1.2.1 5G Future Networks

Efforts to meet the continuously growing demand for higher data rates with ultra-low latency for mission-critical applications and complex multimedia applications like virtual reality (VR), coupled with breakthroughs in mobile broadband technology, motivated the development of 5G technology. 5G was developed to meet the projected data rates and bandwidth demands of future devices and systems. By opening more spectrum and utilizing advanced data modulation schemes, 5G can offer blazing fast data rates of up

to 10 and 20 Gbps for uplink (UL) and downlink (DL), respectively [6], and ultra-high bandwidth for the most demanding applications and services.

FIGURE 1.2: 5G Network Bands

Essentially, 5G offers three layers of frequency ranges to serve different applications. The Coverage and Capacity Layer, operating within the 2-6 GHz range, is the primary band for current 5G deployments [6], offering the best trade-off between coverage and capacity. The Super Data Layer operates at frequencies over 6 GHz, suitable for extremely high data rate applications. Lastly, the Coverage Layer, operating at frequencies below 2 GHz, offers lower bandwidth and data rates but provides the best deep indoor coverage due to the superior penetration capacity of EM waves at greater wavelengths.

1.2.2 Problem Landscape

Findings show that at frequencies below 6GHz, the spectrum utilization of primary users with exclusive licenses is only about 10 to 20% in both spatial and temporal usage domains, while the demand for commercial mobile bands continues to rise [19].

To accommodate the ongoing increase in network traffic across various industries, solutions such as Spectrum Refarming and Millimeter Wave (mmWaves) Technology are being considered [11][17]. Spectrum refarming, which involves opening new frequencies or restructuring existing frequency allocations, is projected to meet the growing wireless capacity demand [11]. However, implementing Spectrum Refarming in the microwave

band (below 6GHz) is highly tedious and largely infeasible due to the exclusive fragmentation and licensing of the most useful frequency bands to non-mobile network users such as medical services, television broadcasting, and military services [11][17].

While the millimeter wave portion of 5G bands (30GHz–300GHz) is well-suited to provide extremely high bandwidth, it faces implementation challenges due to its high directionality and the need for line-of-sight (LOS) conditions for optimal propagation. This requires a dense network of antennas, potentially leading to high network densification costs. Furthermore, mmWaves exhibit rapid signal strength fluctuations, resulting in unstable connections that demand more flexible communication methods [17].

The challenges encountered in 5G deployments have highlighted the need to explore alternative solutions to meet the data rate and capacity demands of future devices and systems. One proposed solution is Spectrum Sharing, which aims to increase bandwidth for Mobile Network Operators (MNOs) by enhancing spectrum utilization in existing licensed frequency bands. The goal of Spectrum Sharing is to enable seamless and efficient sharing of spectrum resources among multiple classes of users, thereby maximizing spectrum utilization and providing more reliable connectivity. With Spectrum Sharing, unlicensed operators can opportunistically use idle frequency bands and systematically vacate the band when the licensed user, commonly referred to as the Primary User (PU), is present.

To achieve an effective spectrum sharing system in existing networks, radios need to be upgraded to become "Cognitive". A key aspect of cognition required in spectrum sharing is Spectrum sensing. This involves the ability of a radio to recognize features of surrounding emitters, such as modulation techniques and power levels that can be used for futher decision making in dynamic shared spectrum (DSA) systems [21]. Modulation recognition, in particular, provides substantial information about surrounding signals and is crucial for implementing DSA cognitive radio systems [21]. Among various traditional techniques for modulation recognition, methods based on Artificial Intelligence have garnered significant attention from various stakeholders. The ability of machine learning algorithms to learn features without the need for expert knowledge modeling makes them very attractive for various recognition tasks across different domains, including image processing, video processing, and speech analysis [14][20]. In this project, we explore and investigate the effectiveness of Convolutional Neural Networks in performing the signal modulation classification tasks necessary in future cognitive radio networks for spectrum sharing.

1.3 Project Aim and Objectives

This project aim to give an holistic yet concise view of the role of **deep learning** in future cognitive radio networks that implements **Shared Spectrum** for high **spectrum utilization**, Hence we aim to achieve the following objectives:

- Perform comprehensive review of exiting spectrum sharing and sensing techniques.
- Investigate the effectiveness of Convolutional Neural Network (CNN) deep learning architecture for performing modulation classification of raw radio signals.
- Develop a Deep Learning Processor (DLP) for a trained Convolutional Neural Network (CNN) model, specifically tailored for the Xilinx Zynq Ultrascale+ family of devices. Present and assess the post-implementation resource utilization and timing reports to evaluate performance and efficiency.

Chapter 2

Background

The remainder of this report is organized as follows: Chapter 2 lays the groundwork for understanding the aims and objectives of developing a signal modulation detector and classifier on custom hardware. We start with a discussion of spectrum sharing taxonomy, outlining the different modes of spectrum sharing and briefly exploring the known frameworks for each mode. Next, we review cognitive radios and their various approaches to implementing shared spectrum. We then survey spectrum sensing in both narrowband and wideband scenarios and conclude with an examination of the techniques used in implementing signal modulation classification systems.

In Chapter 3, we outline the steps and decisions involved in designing a CNN-based signal modulation classifier for a target FPGA hardware. We begin by introducing our design tool of choice and then describe the method used to obtain the training dataset for the Convolutional Neural Network (CNN) model. Following this, we provide a detailed exploration of the methodology for constructing, training, and validating the CNN-based signal modulation classification system. Finally, we discuss our approach to implementing the trained CNN model in hardware design for a Xilinx Zynq Ultrascale+ family device.

In Chapters 4 and 5, we present a comprehensive analysis of the performance of the developed CNN-based signal modulation classifier and its implementation on custom hardware. We then conclude with insights into potential future directions for the system.

2.1 Spectrum Sharing Frameworks

As introduced in chapter 1, there is a growing demand for more bandwidth and capacity to serve various data hungry applications. The proposed 5G technology promises to

provide new pathways to high data rate and high capacity for the projected growth in high data rate, ultra-low latency application. However, various challenges due to the inherent nature of radio waves and associated wireless channel makes some aspect of 5G still a promise. The 5G spectrum as shown in Figure 1.2 spans low frequencies below 2GHz up to higher mmWaves greater than 6GHz. The overarching goal of 5G technology is to create an heterogenous network that can meet a diverse range of demand and use-cases.

Frequency bands below 6GHz are best suited and currently feasible for common wide area 5G deployment [6]. However, these bands are largely assigned to dedicated operators who own exclusive license to operate on these bands. Albeit these low-mid 5G frequency bands are highly underutilized. For MNOs to meet the growing capacity demands and for 5G to fulfil its promises, there needs to be the development of more effective schemes to improve the utilization of limited spectrum in these bands.

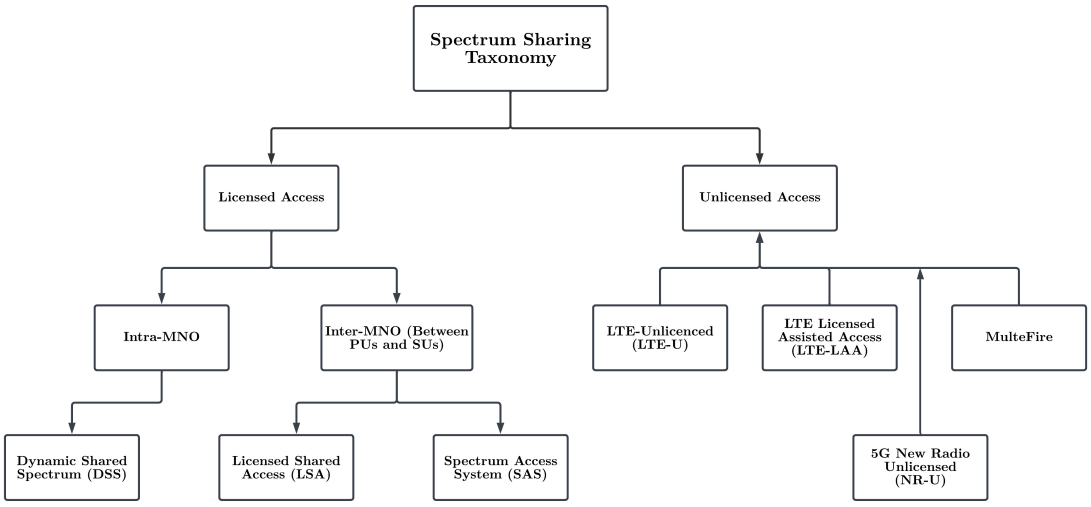

FIGURE 2.1: Spectrum Sharing Taxonomy

Spectrum sharing schemes aim to solve this inefficiency in underutilized spectrum bands in the coverage and capacity 5G layer. Spectrum sharing allows MNOs opportunistically utilize incumbents' frequency bands while it is unoccupied and vacate the band when the incumbent needs to use the band. While it seems trivial, spectrum sharing is complicated and requires careful design framework to provide sufficient incentive to incumbents to lease/share their licensed bands and to MNOs to gain the benefit for increased capacity from utilizing unused frequency bands.

There are various spectrum sharing frameworks with their associated use-cases and tradeoffs. The taxonomy of spectrum sharing is shown in Figure 2.2. Essentially spectrum sharing can be done in two classes of spectrum use cases: sharing in licensed spectrum bands and sharing in unlicensed spectrum bands.

2.2 Spectrum Sharing in Licensed Bands

Licensed portion of the radio spectrum consist of users and operators who own exclusive license to use a dedicated portion of the radio spectrum. These users are known as Primary users (PUs) or Incumbents. Spectrum sharing frameworks in these categories of spectrum use case can be further divided into Intra-MNO spectrum sharing and sharing between MNOs and PUs. In the former, sharing can either be done in a dynamic fashion, known as Dynamic spectrum sharing (DSS) while in the latter case, radio frequency is systematically shared among Licensed operators [21]. Such schemes include the Licensed Shared Access (LSA) and Spectrum Access System as show in Figure 2.2.

2.2.1 Licensed Shared Access

Licensed Shared Access (LSA) is a framework promoted by the Electronic Communications Committee (ECC) of the European Conference of Postal and Telecommunications Administrations (CEPT). Initially targeting the 2.3–2.4 GHz band, LSA provides a regulated approach to sharing spectrum between incumbents (PUs) and Secondary Users (SUs) [17]. The LSA system comprises two principal components:

1. **LSA Repository (LR):** A centralized database that holds information about the incumbent users and the protection requirements needed to avoid interference with their operations.
2. **LSA Controller (LC):** An entity that communicates with the LR to obtain LSA spectrum resource availability information (LSRAI). The LC uses this information to manage and allocate spectrum to licensed operators in a way that respects the protection criteria for incumbent and ensures high QoS for both PUs and SUs.

To enhance LSA, the European Telecommunications Standards Institute (ETSI) is developing an advanced version called enhanced LSA (eLSA). This initiative aims to extend LSA's capabilities to support local, high-quality wireless networks operated by vertical sector operators (e.g., manufacturing, healthcare, and logistics industries). The EU project ADEL has further refined the LSA model by incorporating dynamic behaviors, such as the integration of sensing networks and optimization algorithms, to improve spectrum utilization efficiency. The goal of the ADEL perspective is to enhance spectrum utilization through several key innovations [11]:

1. **Dynamic Radio Resource Management (RRM):** Implementing adaptive techniques for managing radio resources to respond effectively to varying network conditions and demands.
2. **Sensing and Reasoning:** Utilizing database-assisted collaborative sensor networks to improve the detection and understanding of environmental factors affecting radio performance.
3. **LSA Architecture Extension:** Extending the Licensed Shared Access (LSA) architecture to enable more efficient RRM, thereby improving Quality of Service (QoS) and ensuring better compliance with regulatory policies for all stakeholders.

Finally, among other propositions, the ADEL project proposes a dynamic 5G mobile network that allows license holders to be non-MNOs. Other type of network providers including virtual operators, professionally operated TVWS, Private Network Providers are also permitted to own licenses to leverage on the underutilized portions of the radio spectrum [11].

2.2.2 Spectrum Access System

In contrast to the two-tiered LSA system, SAS for the Citizens Broadband Radio Service (CBRS) band (3.55–3.7 GHz) includes an innovative three-tiered sharing framework defined by the FCC, allowing three levels of user access [19][17]:

- Incumbent Users (Tier-1) Access
- Priority Access License (PAL) Users (Tier-2)
- General Authorized Access (GAA) Users (Tier-3)

The top tier (Tier-1) comprises the incumbent users (PUs), including naval radars, Fixed Satellite Services (FSS), and specialized Environmental Sensing Capability (ESC) sensors [19][add]. These users represent existing federal and military operations, primarily situated in coastal regions [19]. Incumbents have absolute priority over the spectrum, meaning they can access the CBRS band without restrictions whenever needed. The system mandates stringent protection against interference from the lower-tier users to safeguard the incumbents operations.

The second tier (Tier-2) consists of Priority Access License (PAL) users, who gain access through competitive bidding on a county-by-county basis [19]. Winning a PAL license grants these users, typically MNOs and industrial firms, preferential access to specific

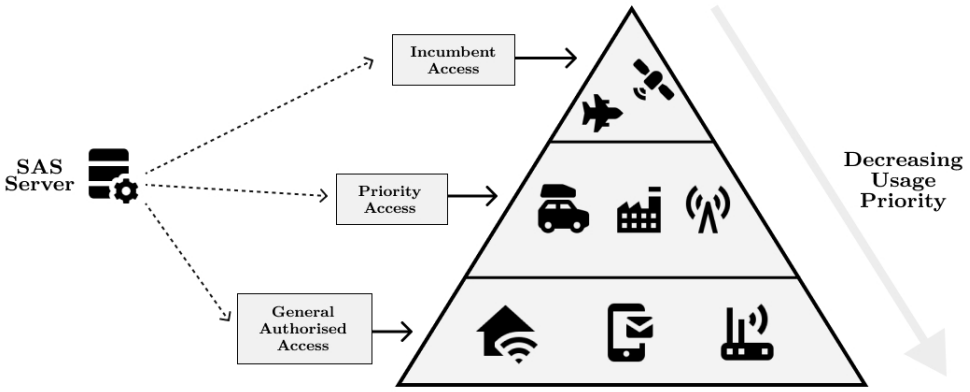

FIGURE 2.2: Spectrum Access System (SAS)

portions of the CBRS spectrum bands, provided incumbents are not using those bands or the PAL users can mitigate interference to an acceptable level. The CBRS band is divided into 15 channels, each 10 MHz wide. A PAL user can utilize up to 7 out of the first 10 channels in any given location, ensuring that at least 80 MHz of the spectrum remains available for General Authorized Access (GAA) users [19].

The third tier is the General Authorized Access (GAA) users, who have the lowest priority. These users can freely access the entire CBRS band, encompassing all 15 channels, as long as they do not interfere with the operations of the incumbents or PAL users. The GAA tier allows for flexible and open access to the spectrum, facilitating a broad range of applications such as private LTE and 5G networks, Internet of Things (IoT) networks, and campus hotspots [19].

The orchestration between these three user tiers is key in the operation of SAS and may be extended to other largely underutilized frequency bands. At the heart of this orchestration lies the SAS server, which is managed by an SAS administrator. The SAS server plays a crucial role in maintaining the balance and efficiency of spectrum usage. The SAS server is central to the spectrum management system. It maintains multiple databases that track all spectrum users and their activities, both current and planned, within a given area. This information is essential for coordinating access and ensuring that the different user tiers can operate without causing harmful interference to one another.

Citizens Broadband Radio Service Devices (CBSDs), which are the devices seeking to use the spectrum, must communicate with the SAS server. They send requests for spectrum use, detailing their intended transmission frequencies and times. The SAS

server processes these requests by checking the databases to determine whether the requested spectrum is available and whether granting access would cause interference with incumbent or higher-priority users. When a CBSD submits a request, the SAS server evaluates it based on the current usage patterns and the priorities of the different user tiers. For instance, if an incumbent user is active on a particular frequency, the SAS server will deny access to that frequency for both PAL and GAA users to prevent interference. If a PAL user has a license for a specific channel and location, the SAS server ensures that GAA users do not interfere with the PAL user's operations.

The SAS server dynamically allocates spectrum based on real-time conditions and the hierarchical rules established by the WInnForum specifications. This dynamic allocation allows for efficient use of the spectrum, as it can adapt to changing conditions and usage patterns. For example, if an incumbent user vacates a frequency, the SAS server can re-allocate that frequency to PAL or GAA users, thereby maximizing spectrum utilization.

2.3 Spectrum Sharing in Unlicensed Frequency Bands

Growing demand for wireless bandwidth has led to the need to develop effective spectrum management schemes that can harness existing unlicensed frequency bands. Unlicensed spectrum, commonly utilized by Wi-Fi and other technologies, offers a valuable prospects for enhancing network capacity in both existing LTE networks and newly emerging 5G networks. Some effective and fair spectrum sharing framework in the unlicensed bands include the following: LTE-Unlicensed (LTE-U), LTE-Licensed Assisted Access (LTE-LAA), MulteFire, and NR-Unlicensed (NR-U) [17].

2.3.1 LTE-Unlicensed

LTE-Unlicensed (LTE-U) is a duty-cycling enabled spectrum sharing framework that enables LTE to operate in the unlicensed 5 GHz band, which is traditionally used by Wi-Fi. It relies on Carrier Aggregation (CA) to combine the unlicensed spectrum with a licensed carrier that serves as the Primary Cell (PCell) [17]. While the unlicensed spectrum is being utilized by the network to boost data throughput, the primary connections including, the critical signaling and control functions is being maintained on the licensed spectrum of the network provider. Hence the licensed carrier serves as the primary reference point for the connection. This combination enhances user performance by providing access to additional spectrum during a connection. However, LTE-U by using a duty-cycling mechanism for channel access, which allows it to transmit in the unlicensed band during ON periods of the duty cycle regardless of the presence of other

devices [17], LTE-U can lead to potential interference with other users in the band, such as Wi-Fi devices.

2.3.2 LTE-Licensed Assisted Access

In contrast to duty-cycle enabled LTE-U, LTE-Licensed Assisted Access (LTE-LAA) utilizes a Listen Before Talk (LBT) scheme and CA for combined (licensed and unlicensed) spectrum access [17]. LBT is a contention-based protocol ensuring fair coexistence with other technologies like Wi-Fi. LBT involves an Energy Detection (ED) procedure to sense the channel's availability; if the detected energy exceeds a certain threshold, the channel is deemed busy, and transmission is postponed. LAA has undergone enhancements resulting in Enhanced LAA (eLAA) and Further Enhanced LAA (feLAA) [17], which provide improved features and performance.

2.3.3 MulteFire

While both LTE-U and LTE-LAA require a licensed anchor channel for signaling and control functions, MulteFire is designed for a more standalone type of radio communication in the unlicensed spectrum [17]. In MulteFire, each connection setup does not require a licensed anchor channel [17]. MulteFire aims to promote the standalone use of LTE in unlicensed bands, making it suitable for a variety of deployment scenarios, including private networks. It also employs the LBT mechanism to ensure fair coexistence with other users in the unlicensed band [17].

2.4 Cognitive Radio Technology

Radios with the ability to sense surrounding radio spectrum for the presence of radio transmissions and dynamically reconfigure its systems to utilize the most favorable, low interference frequency band are known as Cognitive Radios. First introduced by Mitola and Maguire in 1999 [10], Cognitive Radios are a move from static blind radio nodes that are hard coded to execute predefined radio protocols to intelligent radio-domain aware agents capable of improving radio performance and spectral efficiency through dynamic system reconfiguration to deliver the required QoS to connected users [10].

When SU cognitive radio nodes utilize underutilized radio spectrum, their operations can be categorized into three groups namely: Interference avoiding behavior (spectrum interweave), Interference controlling behavior (spectrum underlay) and Interference mitigating behavior (spectrum overlay) [21].

2.4.1 Spectrum Interweave

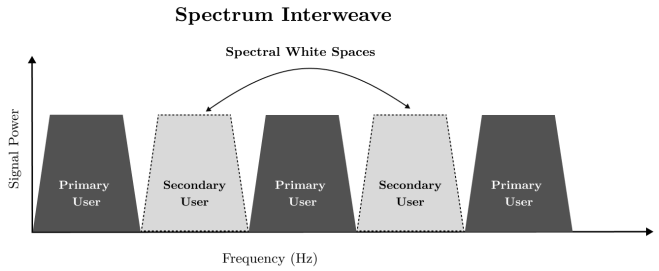

FIGURE 2.3: Spectrum Interweaving

Spectrum Interweaving is an interference-avoiding behavior in cognitive radio networks that allows secondary users to utilize the spectrum (in a licensed spectrum scenario) without interfering with the primary user. Spectrum Interweaving enables the coexistence of primary and secondary users in both FDMA and TDMA modes, provided that interference from secondary users to primary users is completely forbidden [21][20] as illustrated in Figure 2.3. Cognitive radio nodes achieve spectrum interweaving by exploiting spectrum holes in the primary user's operating band, both in the time and frequency domains, and dynamically transmit data as long as the spectrum gap remains present [21]. To mitigate any interference, the secondary user's cognitive node immediately vacates the frequency band as soon as it senses the presence of the primary user [20].

2.4.2 Spectrum Underlay

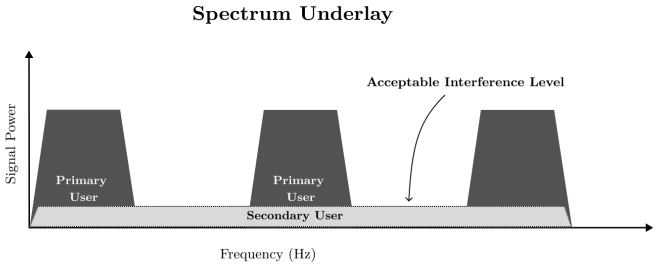

FIGURE 2.4: Spectrum Underlaying

Spectrum Underlay offers a more controlled, interference-permitted mode for secondary users (SUs) to utilize primary user (PU) frequency bands [21][20]. In Spectrum Underlay, the requirement for cognitive nodes to detect spectral holes is removed [21]. Instead, SU cognitive nodes must have knowledge of acceptable interference levels and the effect of their transmission on the PU, both of which are characterized by the theoretical

understanding of the physical wireless channel between the SU node and the PU receiver [21]. Cognitive SU nodes adhere to the "permitted interference" levels as shown in Figure 2.4, by adjusting their transmit configurations, such as transmit power, or by operating exclusively in transmitting regions (geographical locations where sharing is permitted) of the PU network [20]. Information about such geolocation data can be obtained from a central database or GPS data. Additionally, channel state information, which could help the SU reduce interference, can be obtained by sensing the PU's pilot signals [20].

2.4.3 Spectrum Overlay

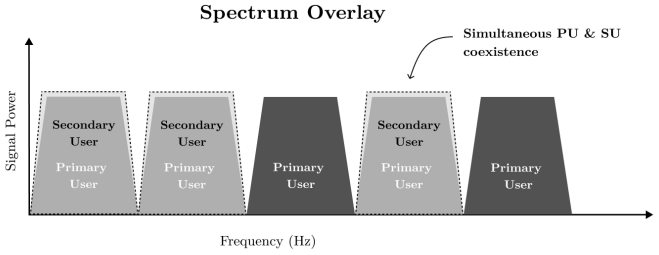

FIGURE 2.5: Spectrum Overlaying

In the Spectrum Overlay sharing technique, cognitive secondary user (SU) nodes can concurrently transmit with the primary user (PU) on the same frequency band [21][20] as shown in Figure 2.5. Under these conditions, the SU cognitive node has additional knowledge about the PU's system. Information such as the PU's encoding methods (codebooks) [21] allows SUs to understand how the PU's radio system encodes data, which is crucial for managing and mitigating interference effectively. By possessing the PU's codebooks, the SU receiver node can systematically cancel interference using techniques like Dirty Paper Coding (DPC) [21][20]. In DPC, the SU's transmitter uses knowledge of the interference from PU transmissions to precode its own signals. This approach ensures that the SU's signals are designed to cancel out or significantly reduce the interference from the PU at the SU's receiver. Essentially, the SU's transmission is tailored to counteract the effects of PU interference. Other advantages of knowledge about PU codebooks in spectrum overlay include the possibility for the SU radio system to cooperate with the PU by relaying or forwarding messages, which could potentially expand the coverage and improve the reliability of the PU's network [20]. These techniques enable effective spectrum overlay without causing detrimental interference to either the PUs or the SUs.

2.5 Spectrum Sensing Techniques

Essentially, spectrum sharing techniques can be further grouped into two categories: Duty Cycle Enabled Spectrum Sharing and Spectrum Sensing Enabled Sharing. The former, as seen in LTE-U, suffers from interference problems due to strict duty cycle constraints. However, Spectrum Sensing Enabled Sharing offers a more dynamic approach by having the radio sense the target frequency band for the presence of transmissions before transmitting. This dynamic approach leads to safer usage of the both licensed and unlicensed spectrum with minimal interference. Spectrum sensing techniques can be classified into two categories: narrowband sensing and wideband sensing.

2.5.1 Narrowband Sensing

In Narrowband sensing technique, the cognitive radio node is interested in utilizing only a specific limited number of frequency bands and the RF environment is relatively stable. The following are some techniques used in narrowband spectrum sensing.

1. **Energy Detection Method**

   Energy detection is a simple and common technique used in spectrum sensing. In this method, a given energy threshold is set in the radio node, and energy measurements are compared against this target energy level. The received signal energy is computed over a fixed period, and the comparison between the measured energy and the target energy threshold is used to decide the presence or absence of the primary user (PU) in the target frequency band. The energy detection method can be modeled as the sum of the squares of the received signal samples for a given sample period N, as shown in Equation 2.1. E represents the total signal energy present in the target frequency band at a given time interval.

$$E = \frac{1}{N} \sum_{n=1}^{N} |x[n]|^2 \tag{2.1}$$

   The target energy threshold can be obtained through a separate process designed to determine the noise-only period of the received signal. This noise-only period provides an estimate of the baseline noise level in the channel, which is essential for setting the energy detection threshold. Using the noise-only period, the estimated detection threshold is calculated (as shown in Equation 2.2) based on the estimated noise variance, σ and the scaling factor α, which represents the probability of a false alarm in the system.

$$\text{Threshold} = \alpha\sigma^2 \tag{2.2}$$

2. **Matched-Filter Detection**

   Compared to the Energy detection method, Matched-filter detection is a more advanced signal detection technique that involves correlating a received signal with a known reference signal, which is a replica of the PU's signal [3]. This method requires precise knowledge of the PU signal waveform to construct an optimal filter that matches the expected signal characteristics. The process entails computing the correlation between the received signal and the reference signal, adjusting for time delay to align them accurately. If the resulting correlation value exceeds a predefined threshold, the detector confirms the presence of the PU signal, otherwise the channel is said to be unoccupied [3]. This technique is particularly effective in low signal-to-noise ratio (SNR) conditions and in environments with additive white Gaussian noise (AWGN) and fading channels [3]. Details of the PU signals might not be available in some scenarios, hence matched-filtering detection might not be always feasible. A match-filter detection system can be modelled as shown in Equation 2.3 where R denotes the correlation between the received signal $x(t)$ against the time delayed reference signal, $s^*(t-\tau)$.

$$R = \int_{-\infty}^{\infty} x(t)\, s^*(t-\tau)\, dt \tag{2.3}$$

3. **Cyclostationary-Feature Detection**

   Cyclostationary-feature detection works by leveraging the periodicity and statistical characteristics of the PU's signal [3][8]. Unlike matched-filter detection, which requires detailed knowledge of the PU signal, cyclostationary detection relies on the cyclic auto-correlation function (CAF) of the received signal. The CAF captures the cyclic correlation between the signal and its delayed versions, revealing periodic patterns corresponding to the PU's fundamental frequency. This demonstrated in Equation 2.4 by calculating the auto-correlation between the received signal, $x(t)$ and a delayed complex exponential reference signal $(x^*(t-\tau)e^{-j2\pi ft})$ with frequency f. This method involves analyzing these cyclic frequencies and their magnitudes against a predefined threshold. If the magnitude of any detected cyclic frequency component exceeds the set threshold, the method confirms the PU's presence; otherwise, it indicates its absence. The downside of cyclostationary-feature detection is the computational complexity involved in calculating cyclic auto-correlation of received signals. Also, in non-stationary environments cyclostationary feature detectors perform poorly due to varying periodicity and statistical properties of PU signal [8].

$$R_{xx}(\tau, f) = \lim_{T\to\infty} \frac{1}{T} \int_{-T/2}^{T/2} x(t)x^*(t-\tau)e^{-j2\pi ft}\, dt \tag{2.4}$$

2.5.2 Wideband Sensing

Wideband sensing monitors a broader range of frequencies, potentially spanning several channels or the entire available spectrum. It is advantageous when a broader view of the frequency spectrum is needed to be observed [3]. Albeit wideband sensing is a relatively newer field, and it includes techniques such as Nyquist-based wideband sensing and compressive wideband sensing [3]. In [18] the authors presented a novel multiband joint detection framework designed to enhance wideband spectrum sensing in cognitive radios focusing on optimizing a collection of narrowband detectors to boost opportunistic throughput capacity of cognitive radios while minimizing interference to primary communication systems. Wideband spectrum sensing problem was formulated as a class of optimization problems aimed at maximizing throughput in interference-limited environments. The proposed multiband joint detection strategy enables cognitive radios to better utilize available frequency bands and control interference, improving overall network performance.

2.6 Signal Modulation Classification

Spectrum sensing techniques involving the identification of the presence of a primary user (PU) in a dedicated frequency band prove to be a critical feature of cognitive radio networks. However, in non-cooperative communication systems [1], to achieve more intelligent spectrum management, adaptive modulation schemes that dynamically encode data using different modulation techniques selected from a group of supported modulation schemes have been developed [7] as illustrated in Figure 2.6. In such an intelligent system, the modulation technique used by the transmitter is selected based on the system specification and the channel quality indicator (CQI) channel-state report [5] on the current in-use frequency band. Hence, it becomes important that the receiver can identify the modulation scheme used before demodulation. Automatic Modulation Classification is designed to solve this problem. By carrying out AMC, the receiver can dynamically detect the modulation used by the transmitter without the need for additional signaling and control [7].

The modulation classification problem can be modeled as follows: Given a single-carrier single-input single-output (SISO) radio system, the channel-impaired transmitted signal $x[n, H_k]$ is described by Equation 2.5.

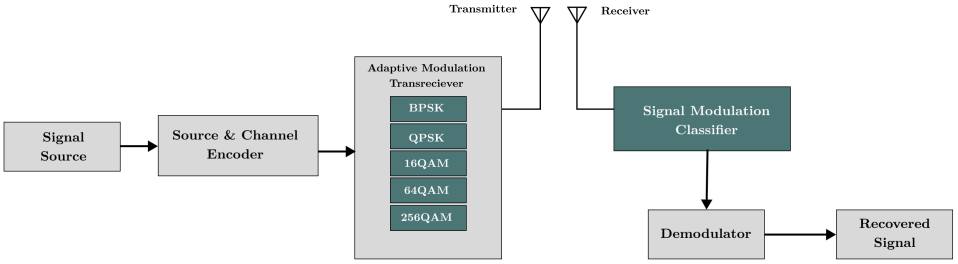

FIGURE 2.6: Adaptive Modulation System

$$x[n, H_k] = A_n e^{j(2\pi f_0 nT + \theta_n)} \sum_{k=-\infty}^{\infty} x[k]h(nT - kT + \epsilon T) \tag{2.5}$$

Where $A_n e^{j(2\pi f_0 nT+\theta_n)}$ is the complex exponential carrier of the original signal $x[k]$ with a varying phase offset of θ_n. Further, $h(nT - kT + \epsilon T)$ is the impulse response response of the baseband channel with timing offset ϵT and $x[k]$ denotes the symbol sequence of the original data being transmitted [7].

At the receiver, the received complex signal y[n] is given by Equation 2.6 Where $x[n, H_k]$ is the recieved signal and $w[n]$ is the additive white Gaussian noise (AWGN).

$$y[n] = x[n, H_k] + w[n] \tag{2.6}$$

The AMC problem involves predicting the modulation technique used in the transmitted signal $x[n]$ without any knowledge of the channel state information. This is achieved by learning salient features present in the previously received signal $y[n]$[7]. AMC methods can be divided into two categories namely: Traditional AMC methods and Advanced AMC methods. In the following sections we briefly describe some of these techniques and highlight their key working principle.

2.6.1 Traditional AMC Methods

Traditional AMC methods consist of techniques that rely on the statistical property of the received signal in making decision about the modulation scheme employed. Traditional AMC methods can be further divided into two group namely Decision Theoretic methods and Feature-Based method [2].

1. **Energy Detection Method** Decision-theoretic methods comprises of techniques that rely entirely on statistical characteristics of the received signal in making prediction about its modulation scheme. There are two group of decision-theoretic methods. The first group is known as likelihood-based methods [2] which rely on pre-defined set of hypotheses mapped to different modulation types. The likelihood estimation for each modulation hypothesis formulation is estimated for the received signal, this estimation gives the predicted signal model for the received signal. With that information, the likelihood function is the calculated based on the estimated signal model. Finally, the likelihood ratio test is performed, and result is compared with a threshold, which is used to evaluate the correctness of the predicted modulation scheme used in the received signal.

   The second group of decision theoretic methods is known as the Distribution-Based methods [2]. These methods involve evaluating the empirical distribution of the received modulated signal. Typically, the signal distribution is determined by the radio channel parameters in conjunction with the symbol mapping type of the received signal. Given the channel parameters, the symbol mapping can be used as a major factor for determining the modulation of the incoming signal. Once the distribution of the incoming signal is determined, the Goodness-of-Fit (GoF) function [2] is calculated and used as a metric for distribution matching.

2. **Feature-Based Methods** Feature-based methods involve ways of extracting predefined statistical features of the received signal and using the extracted features as a basis for decision making. Feature-based methods can be grouped into two stages [2]. The first stage is the feature extraction stage, consisting of various methods for extracting unique features from the received signal. Some of these methods include Spectral Based feature extraction [2] which depends on specific spectral characteristic of the received signal, such as the signal amplitude, phase, and frequency. This technique specifically terminates with a decision tree classification method that consists of separate tests nodes dedicated to each spectral feature of the signal. Moment-based feature extraction [2] is another type of feature-based method. This involves formulating hypothesis tests based on mathematically calculating the kth-order moment of the received complex value signal. These moments reveal inherent statistical property of the received signal and is used as the basis for decision making. Cyclostationarity is a technique used in extracting useful features from a receiving signal. As briefly explained in Section 2.5.1, cyclostationary signals are signals that change their statistical properties with time. This time-varying properties of signals can be tested and evaluated using a Spectral Correlation Function (SCF) [2]. Different modulation schemes have different SCF signature, hence by observing the SCF of a signal, we can obtain useful set of features necessary for

modulation classification. Other feature-based methods include Cumulant-based feature extraction, Transform-domain feature extraction etc [2].

The second stage is the classification stage which relies on a Machine learning model to make predictions based on extracted features. Different type of machine learning classifiers represents features in different forms and use various analytical and statistical techniques to make predictions. The following are few a machine learning classifier commonly used in feature-based AMC.

- **Support Vector Machine (SVM) classifiers**: SVM classifiers are designed to work in multi-dimensional feature spaces with the aim of identifying a hyperplane that best separates different classification outcomes [2]. This process can be framed as an optimization problem where the goal is to maximize the margin, or distance, between the hyperplane and the nearest data points from each category, known as support vectors. By focusing on this margin maximization, SVMs aim to achieve robust classification. SVMs employ a supervised learning approach to select the most appropriate kernel function, which enables them to effectively differentiate between various modulation categories based on the extracted feature set.
- **Decision Tree Classifiers**: Decision Tree (DT) classifiers are a type of non-parametric supervised learning algorithm [2] known for their simplicity and efficiency in classification tasks. They work by creating a model that predicts the class of a given sample by learning simple decision rules inferred from the training data. DT classifiers are advantageous for their ease of interpretation and relatively low memory requirements. However, their performance can degrade in noisy environments or when the training dataset lacks sufficient features [2]. They primarily function as binary classifiers but can be categorized into fine-tree, coarse-tree, and medium-tree classifiers based on their structure and accuracy. Fine-tree classifiers [2], with a large number of leaves, are highly accurate and well-suited for datasets with many classes. Coarse-tree classifiers, featuring fewer leaves, offer lower accuracy but are more robust and simpler to interpret for problems with fewer classes. Medium-tree classifiers balance the tradeoff by providing moderate accuracy with a manageable number of leaves, making them a versatile choice for various classification challenges.

K-nearest neighbors (KNN), ANN and Polynomial classifier [2] are other examples of machine learning techniques for processing and making decision based on extracted features.

2.6.2 Advanced AMC Methods

The explosion of Deep Learning algorithms in various field such as image processing, speech analysis etc. have witnessed substantial progress in recent years [1]. As a result, the application of Deep learning in wireless communications networks has sparked the interests of research and industry [2]. A common use-case of DL in wireless system is in recognition and classification of signals based on modulation scheme used [2]. However, DL algorithms also can be applied at the PHY layer of wireless communication systems for tasks like Interference alignment, Jamming resistance, and physical coding [1]. Essentially, there are three types of Deep learning architectures that can be implemented. Muti-layer Perceptron, Deep Neural Network and Recurrent Neural Network [2].

1. **Multilayer Perceptron** – Deep Belief Networks: Deep Belief Networks (DBNs) a type of multilayer perceptron model are a sophisticated class of generative models that excel in capturing intricate data distributions through multiple layers of Restricted Boltzmann Machines (RBMs). Each RBM within a DBN consists of a visible layer and a hidden layer, and these RBMs are stacked sequentially to form a deep network architecture. The DBN learns hierarchical representations of data by first pre-training each RBM in an unsupervised manner to capture low-level features, followed by fine-tuning the entire network using supervised methods if labeled data is available. This hierarchical approach enables DBNs to effectively model complex patterns and relationships within the data, making them highly effective for automatic modulation recognition and classification tasks. By leveraging this multi-layered feature learning process, DBNs can extract increasingly abstract and meaningful representations, leading to advanced pattern recognition on new data [2]. Fully connected, Feedforward MLP models that employ supervised learning in an end-to-end fashion to map input features directly to output labels can also be used to learn features necessary for modulation classification.

2. **Deep Neural Network** – Convolutional Neural Networks: Convolutional Neural Networks (CNNs) are of primary interest in this report. They are a specialized type of feed-forward deep neural network that excel in processing high-dimensional, unstructured data, particularly images. Unlike traditional neural networks that rely heavily on general matrix multiplications, CNNs utilize convolution operations to enhance learning efficiency. This process involves three primary layers: convolutional layers, pooling layers, and fully-connected layers as illustrated in Figure 2.7.

    - **Convolutional Layers (CONV)** are an essential component of Convolutional Neural Networks (CNNs) which distinguishes them from other types of

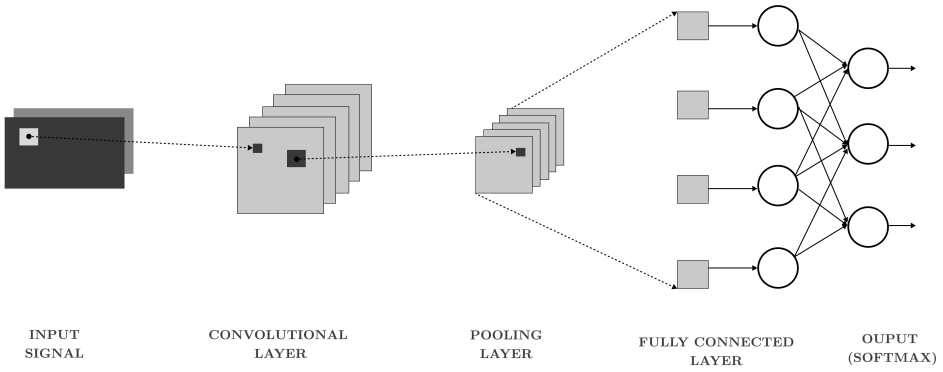

FIGURE 2.7: Convolutional Neural Network Architecture

Deep Neural Networks (DNNs). The convolutional layer has the unique ability to process spatial hierarchies in data. The fundamental operation within a convolutional layer involves a set of learnable filters, also known as kernels, which are convolved with the input data to produce feature maps. Each filter is designed to detect specific patterns and features in input signal which contributes to the network's overall ability to learn and recognize complex structures in the data [12].

Also, another notable characteristic of convolutional layers is their sparse connectivity. Unlike traditional neural networks, where each input unit connects to every output unit, CNNs use filters that are smaller than the input dimensions. This design significantly reduces the number of parameters required, as each filter is applied across different spatial locations of the input, hence creating a sparse connection pattern [13] which enhances the statistical efficiency of the network and lowers memory consumption, making it possible to handle large inputs batches.

In addition to sparse connectivity, CNNs leverage parameter sharing [2][12] through their convolutional layers. This technique allows the same set of filters to be applied to different regions of the input, resulting in the reuse of parameters across multiple spatial location in the input feature map. As a result, the overall storage requirements of the model are greatly reduced. The convolutional operation is mathematically expressed in Equation 2.7:

$$O[z][u][x][y] = B[u] + \sum_{k=0}^{C-1} \sum_{i=0}^{S-1} \sum_{j=0}^{R-1} I[z][k][U_x + i][U_y + j] \times W[u][k][i][j] \quad (2.7)$$

Where C, S, R represents the number of channels, width, height of the convolutional filters which forms the spatial connection pattern from the input signal. The output, $O[z][u][x][y]$ represents the output feature map of the convolutional layer which is determined by the convolution between the input feature map I and the filter matrices W. z is the batch index and U_x and U_y is the convolutional layer stride in the vertical and horizontal dimension [12].

Parameter sharing due to filter reuse in the convolutional layer both optimizes the memory usage and enhances the model's ability to generalize by focusing on local patterns rather than the entire input space [12]. With the combination of sparse connectivity and parameter sharing, the convolutional layer enables CNNs to efficiently process high-dimensional inputs while maintaining a decent model size. As the network deepens, successive convolutional layers build on increasingly abstract features, leading to a powerful hierarchical representation of the input data [12][2].

- **Pooling Layer** are primarily aimed at reducing the spatial dimensions of feature maps while maintaining the essential characteristics of the data. Pooling layers achieve dimensionality reduction by aggregating features from local regions of the input, which helps in making the network less computationally intensive and more efficient [12]. Two common pooling operations are max pooling and average pooling. Max pooling selects the maximum value from a grid of values within the feature map and ensures that the most prominent features are preserved while reducing the overall size of the data. This enhances the network's ability to maintain feature invariance, particularly translation invariance which implies that small shifts or translations of the input image do not significantly impact the output feature map, as the pooled representation remains largely unchanged [2][15]. In addition to max pooling, average pooling is another method where the average value of a grid patch is computed. While both techniques reduce dimensionality, max pooling is often preferred due to its ability to preserve the most salient features of the input, contributing to better performance in various applications [12][2]. Other advantages of the pooling layer include lower computational complexity due to down-sampled feature maps and reduced sensitivity to local transformations in the input feature map.

- **Fully Connected (FC)** layers generally follow the convolutional and pooling layers in CNNs. They are primarily responsible for the final classification tasks which later terminates with a SoftMax layer. After a series of convolutions, nonlinear activations (e.g., sigmoid, ReLu), and pooling operations

that extract and aggregate features, the FC layers serve as the final stage where these features are transformed into class scores or prediction probability [4][12]. FC layers are made up of dense connections i.e., every neuron in an FC layer is connected to all activations in the preceding layer. This dense connectivity allows the FC layers to perform complex transformations on the high-level features extracted by the preceding convolutional and pooling layers. The computations within an FC layer involve affine transformations [12], which include the multiplication of activation values from the previous layer by a weight matrix and the addition of learned bias terms. Affine transformations transform the high-level features to a format suitable to present the resulting classification output of the model.

In summary, Convolutional Neural Networks (CNNs) are highly effective for classifying multi-dimensional data, such as images and videos and other multi-dimensional data (e.g., complex – IQ radio signal) due to their ability to automatically learn and extract hierarchical features from raw input. However, the success of a CNN in achieving high classification accuracy depends not only on the basic parameters like weights and biases, which are learned during training, but also on systematic selection of hyperparameters that guide the training process. Some important hyperparameters include the number of hidden layers and units, dropout probability, activation function, learning rate, loss function, optimization algorithm, and number of epochs. Hyperparameter tuning is a critical and iterative process that involves adjusting these settings to optimize the model's performance. By following best practices and leveraging on insights from existing literature, we can arrive at effective hyperparameters that ensures the CNN models achieve high accuracy on both the training and test dataset.

Other advance methods for AMC include the use of sequential neural networks like Recurrent Neural Networks (RNNs), Long short-term memory (LSTM) and possible combination of sequential and spatial networks e.g., CNNs and LSTM known as Hybrid models [12]

Chapter 3

Methodology

3.1 Design Objectives

In this section we focus on achieving the following objectives

1. Retrieve the RadioML dataset and perform a brief exploratory data analysis to become familiar with the dataset's content.
2. Train and evaluate the performance of a Convolutional Neural Network (CNN) for signal modulation classification.
3. Investigate the effect of the model's receptive field (filter size) on the classification accuracy.
4. Develop and optimize a deep learning processor for the CNN model, and generate an implementation and timing report for the design.

3.2 Design Tool Selection

When designing a system like a Signal Modulation Classifier to run on dedicated hardware, one important development decision to make is the appropriate design tool that would ensure the right type of design is made with the right flexibility enough to meet all design and project requirements. For this project, our goal is to design a signal modulation classifier for a dedicated software-defined radio that runs on a Xilinx FPGA (design requirement are listed in Section 3.1). The two design tool options that were considered are MATLAB and The FINN Project.

3.2.1 The FINN Project

The FINN project is an experimental framework by AMD Research and Development (RAD) team aim for exploring the development of Quantized Neural Network (QNN) accelerators with a dataflow-style architecture on FPGAs. FINN uses the following toolchain to achieve an iterative end-to-end flow as shown in Figure 3.1. At the top of the stack, FINN utilizes Brevitas, a Pytorch Add-on, for achieving efficient Quantization Aware Training (QAT) of deep learning models. QAT enables flexible quantization of deep learning network parameters such as weights and biases by allowing custom customization of each layer's dynamic range, scaling factor etc. An alternative to QAT is Direct Quantization, which quantizes network parameters post-training is said incure more losses in terms of accuracy in the deep learing model. After designing and training the model using Pytorch and Brevitas, the next step in to export the model to a generic ONNX-based intermediate representation.

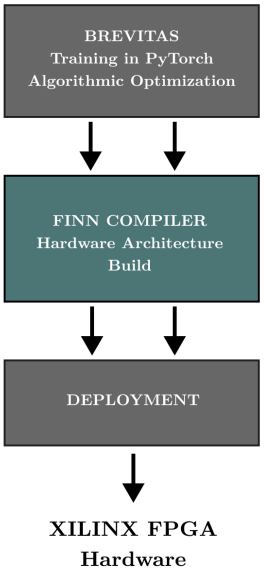

FIGURE 3.1: FINN Project End-to-end Flow

The next step is the FINN compiler stage. The FINN compiler consists of suite of graph transformations on the generated ONNX representation. Graph transformations like streamling, lowering convsToMatMul (lowering convolutions to matrix multiplication) etc are performed on the ONNX representation. After applying the graph transformation, the FINN compiler generates and high-level synthesis (HLS) of the design with can be synthesized in register-transfer logic (RTL) followed by mapping the generated RTL design to the target FPGA hardware in an efficient manner. Finally, the output of the FINN compiler is the generated IP blocks for the deep learing processor for Vivado IP Integrator to be integrated into a larger system e.g., a cognitive radio transceiver

running a Xilinx RFSoC platform. Finally, the generated IP block can be deployed to a physical hardware target using the PYNQ python library.

3.2.2 MATLAB

MATLAB, developed by MathWorks, is a high-level programming language and environment designed for numerical computing and data analysis. It is widely used in academia and industry for prototyping and developing various complex systems. MATLAB provides toolboxes, which are specialized collections of tools and functions designed to address specific domain-based tasks and problems through well-streamlined development processes. The MATLAB Deep Learning Toolbox and a collection of other toolboxes are of interest in this project.

The Deep Learning Toolbox provides a collection of useful tools for designing, training, and testing various deep neural network architectures, including CNNs and transformers. The MATLAB Deep Learning Toolbox enhances the development experience by enabling network visualization and verification through a GUI called the Deep Network Designer. The Deep Network Designer application also allows for editing and interactive analysis of both new and pretrained models. Additionally, it supports exporting the developed model to a generic intermediate representation in ONNX.

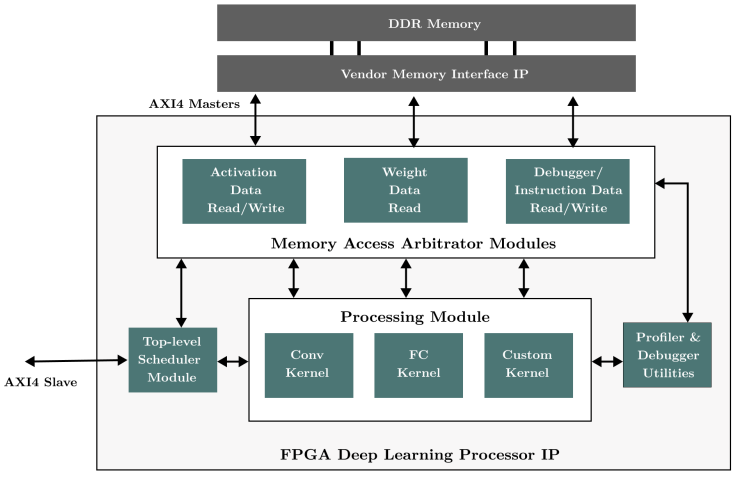

FIGURE 3.2: Matlab Deep Learning FPGA Processor Architeture

To develop a Deep Learning Processor that can be implemented on custom hardware, MATLAB provides the Deep Learning HDL Toolbox for creating target-independent, generic deep learning processor IP cores that are platform-agnostic, as shown in Figure 3.2. This toolbox includes an AXI4 Master interface for reading from and writing

to external off-chip DDR memory, which can store items such as signal data and network weights and biases. The Memory Access Arbitrator Modules act as arbitrators for the Processing Modules containing the deep neural network. They use the AXI Master interface to read and write network activations, weights, and biases to and from the processing modules. Specialized kernels, such as the Conv Kernel and FC Kernel, implement convolutional layers and fully connected layers, supporting tensors of any size. Additionally, custom kernels can be used to implement custom layers in the deep learning network. Other features of the processor, such as the Top-Level Scheduler, Profile, and Debugger utilities, ensure that the developed processor IP integrates well with the target device. The Deep Learning HDL Toolbox also provides workflows for creating design testbenches and Vivado Designs, which can be useful for further design optimization on Xilinx FPGA targets.

Of the two design tool options, MATLAB was chosen due to its learning curve and project time constraints. Nonetheless, FINN offers an interesting set of tools and features that would be worth exploring in the future.

3.3 Dataset Generation

To train a CNN-based signal modulation classifier, we require a dataset that contains received radio signals under various channel conditions that are representative of the real world. Given that this is a supervised learning task, we also need the radio signals to be labelled according to their respective modulation technique. Since radio signals are inherently synthetic in nature—that is, they are digitally generated by the transmitting radio processor—they can similarly be simulated to produce synthetic data for training and testing a deep learning model.

However, to avoid reinventing the wheel, we use the Radio ML Dataset for training, verification, and testing the models developed in this project. Radio ML is an RF dataset for machine learning created by Deepsig Inc., consisting of 24 modulation schemes with 26 SNR levels per modulation type. Details of the dataset are summarized in Table 3.2.

RadioML, developed by [14], consists of synthetic signals and signals received via an over-the-air transmission channel. The signal generation and impairment process involves stages of signal pulse shaping, interpolation, and carrier frequency mixing. The generated signal is convolved with a Rayleigh Channel Impulse Response to simulate multipath fading in wireless channels. Finally, additive Gaussian white noise (AGWN) is applied to the signal [14]. The over-the-air signal capture was performed in an indoor wireless 900 MHz ISM band channel. The clean modulated signals were transmitted

TABLE 3.1: Radio ML Dataset Properties

Property	Value
SNR levels	26 SNRs per modulation (-20 dB to +30 dB in steps of 2dB).
Frame size	1024 complex time-series samples per frame.
Dataset size	2,555,904
Modulation types	24
Signal sourcing	Mixed (synthetic and recieved over-the-air signals)

using a Universal Software Radio Peripheral (USRP) device and received on a B210 software-defined radio. The signals were then stored in frames of 1024 complex time-series samples, along with the ground-truth label for the modulation type used. The overall dataset is a 21.45 GB file stored in HDF5 format, which is designed to store and organize large amounts of data. This dataset is too large to be fully loaded into MATLAB all at once. Therefore, to properly read the dataset, we utilized MATLAB's `h5read` function and developed a wrapper datastore class, `RadioMLDatastore`, which implements a `MATLAB.IO.Datastore` class for fetching the data and labels in small batches.

3.4 Exploratory Data Analysis

Before training, we conduct a brief exploration into the dataset to get a sense of what the complex signal frames look like. First, we extracted a batch of the data consisting of a single frame from each modulation technique at a good SNR value for proper visualization, this is shown in Figure 3.3.

It can be easily observed that while some signals are visibly easily distinguishable, some others aren't. Simple analog modulation techniques like FM appear to be distinct from the rest of the signals. The same observation can be partially said about BPSK, and OOK. However more advanced digital modulation techniques like QAM, PSK, QPSK appear to be highly similar.

From a different perspective, we explore the data in terms of the SNR value. To see how the time signal varies with SNR, we plot the time-domain plot of the signals for the two lowest SNR values (-20dB, -18dB) and for the two highest SNR values (28dB, 30dB) for a selected number of modulation types as shown in Figure 3.4.

From the plots in Figure 3.4, we visually observed the extent to signal degradation in the time-domain plots. At these low SNRs the time-domain signals are barely distinguishable and could pose a tougher challenge for a trained deep learning model.

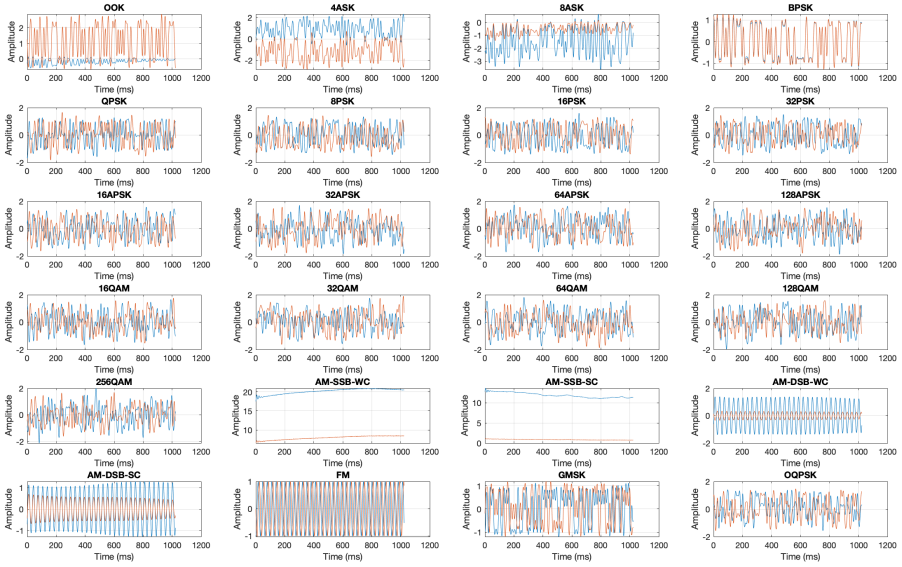

FIGURE 3.3: All Modulation Types @30dB SNR

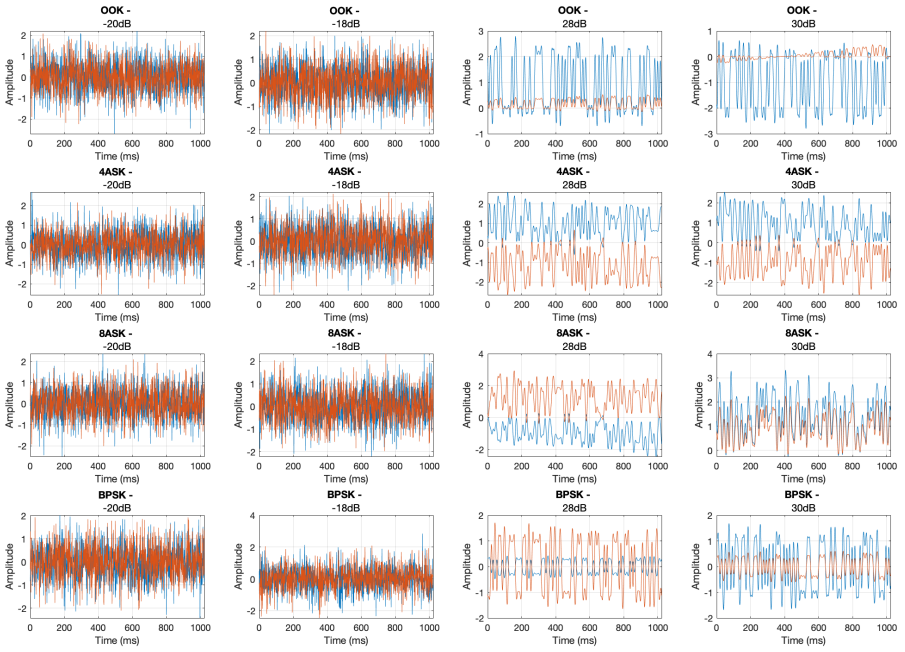

FIGURE 3.4: Comparison Between Signal SNR Extremes

TABLE 3.2: Radio ML Dataset Subset

Property	Value
SNR levels	15 SNRs per modulation (2 dB to +30 dB in steps of 2dB).
Frame size	1024 complex time-series samples per frame.
Dataset subset size	1000 frames per SNR levels
Modulation types	11
Subset size	165,000

To simplify the classification problem, we choose to use only a subset of the dataset in terms of modulation techniques and SNR levels. Table 3.2 summarizes the details of the data subset that will be used to train and verify a CNN-based signal modulation classifier.

3.5 Model Design & Training

We aim to design and train convolutional neural networks that recognizes modulation technique used in a signal given a short 1024 long frame of the signal. There is no expert feature extraction phase, rather, the input to our models are raw in-phase (I) and quadrature (Q) components of the received signal samples which are both translated to real numbers to from a 2-dimensional tensor as shown in Figure 3.5. Hence the network learning solely on raw time-domain signal features, and the spatial relationship between in-phase and quadrature samples.

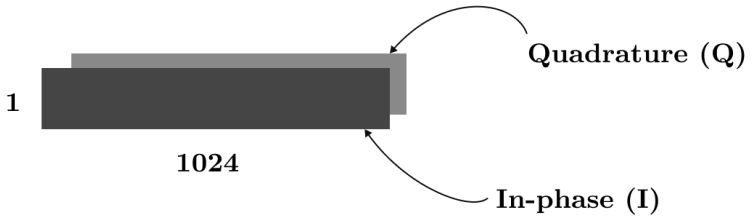

FIGURE 3.5: CNN Model Input Data Schema

3.5.1 Network Design

We train a VGG-adapted [14] 2-D Convolutional Neural network with properties as given in Table 3.5. The network receives a batch of $1 \times 1024 \times 2$ time-domain radio signal and transforms the signal into higher domain signal representation with through multiple convolutional layers. The filter size remains constant throughout the network

Table 3.3: CNN Network Parameters

Property	Value
Conv Layer	2-D Conv → Batch Norm → ReLU → 2D Max Pool
Output Layer	Softmax → Output (cross entropy)
Network Input	$1 \times 1024 \times 2 \times$ miniBatchSize
Number of FC Layers	11
Number of Conv Layers	6

Table 3.4: Model Hyperparamters

Property	Value
Solver	sgdm (stochastic gradient descent with momentum)
Learning Rate	1×10^{-1}
Max. Epochs	20
Minibatch size	1024
Gradient Threshold Method	*l2norm*
Validation Frequency	1 per epoch
Training set	80%
Validation set	10%
Test set	10%

and varies across the different models trained as per our receptive field investigation. Finally, the network terminates with fully connected layers and SoftMax layer to give the final predictions.

We develop four different CNNs each with different receptive field (convolutional layer filter size) while keeping other network parameters like depth fixed.

3.5.2 Hyperparameters

To obtain decent set of hyperparaterters, we implement parameters used by common state-of-the-art networks like AlexNet for training. We use a stochastic gradient decent with momentum (SGDM) solver as the optimization algorithm during training. SGDM combines stochastic gradient decent with momentum enhancement to accelerate the gradient vector in the right direction during training and is known to have faster convergence than plain SGD optimization routines. The network hyperparameters are summarized in Table 3.4.

Table 3.5: Model Receptive Fields

Model	Receptive Field
Model I	1×2
Model II	1×4
Model III	1×8
Model IV	1×16

3.5.3 Receptive Field Investigation

The region in the input data space that affects a particular feature in a network is known as the network's receptive field. The receptive field is critical because, for any image or signal classification task, there is an optimal length of samples or size of filters that efficiently captures both high-level and low-level intricacies of the signal necessary for high classification performance. Similar to other hyperparameters of a CNN, such as the learning rate and cost function, the choice of the model's receptive field is crucial. The receptive field of the model is determined by the size of the filter used at each convolutional layer of the model.

Our analysis focuses on investigating the effect of different chosen receptive fields (filter sizes) on the classification performance of the model on the RadioML dataset.

3.5.4 Train, Validation and Testing

We divide the RadioML segmented dataset into train, test and validation set as shown in Table 3.4. We use 80% of the data for training, 10% for validation and 10% for testing. Training is done for a total of twenty epochs, i.e., we train the models on the entire training set twenty times. The validation frequency is one validation per epoch. By doing so, we can effectively track the progress of the model at every epoch.

3.6 Deep Learning Processor Design

To further validate our proposed signal modulation classification system, we develop a deep learning processor for a target Xilinx FPGA board via MATLAB Deep Learning HDL Toolbox. The Toolbox provide a set of pre-built bitstreams for implementing various deep learning network on Xilinx and Intel FPGA and SoCs. It also provides a set of MATLAB APIs to easily initialize and optimize a deep learning processor in few lines of code. The steps to achieve this is given the following steps.

Algorithm 1 Configure and Build FPGA for Deep Learning Network

Input: A trained deep learning network `trainedNet`
Output: FPGA configuration and performance estimates

Load the best performing model
```
load trained_model_small4.mat
trainedNet ← trained_model_small4
```

Retrieve the layers of the trained network
```
trainedNet.Layers
```

Set the tool path for Xilinx Vivado
```
hdlsetuptoolpath('ToolName', 'Xilinx Vivado', 'ToolPath',
C:\Xilinx\Vivado\2020.2\bin)
```

Create a processor configuration for the FPGA
```
hpc ← dlhdl.ProcessorConfig('Bitstream', 'zcu102_single')
```
Set the target frequency
```
hpc.TargetFrequency ← 100
```

Optimize the processor configuration for the network
```
hpc.optimizeConfigurationForNetwork(trainedNet)
```

Set module properties for the convolution layer
```
hpc.setModuleProperty('conv', 'InputMemorySize', [150, 150, 7])
hpc.setModuleProperty('conv', 'OutputMemorySize', [150, 150, 7])
```

Validate the processor configuration
```
hpc.validateProcessorConfig()
```

Estimate the performance of the processor configuration
```
hpc.estimatePerformance(trainedNet)
```

Build the processor configuration
```
dlhdl.buildProcessor(hpc)
```

At the final stage, we generate the processor test bench, synthesizable VHDL code, Vivado Project and network bitstreams to be programmed on a Xilinx MPSoC FPGA system-on-chip.

3.7 Modulation Classification Demo Application

To demonstrate the model's performance, we developed a simple MATLAB-based Graphical User Interface (GUI). This GUI provides a visual representation of the trained model's performance across different aspects of the data. It allows users to select from

a range of SNR values and a fixed set of modulation techniques for deep learning inference. The classified signals are then plotted on a graph to visually illustrate the signal's characteristics. The GUI is depicted in Figure 3.6.

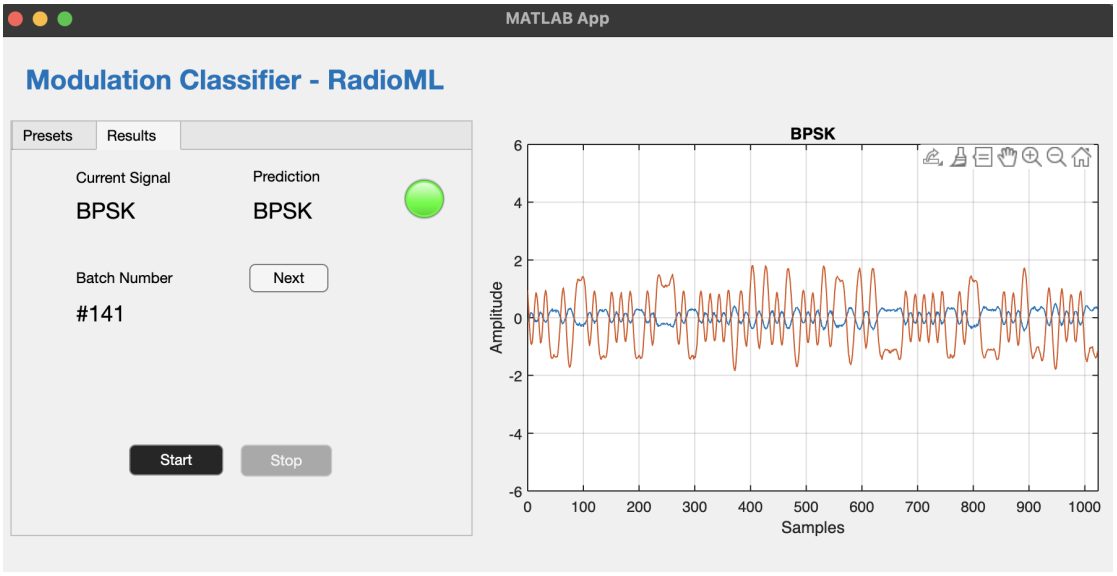

FIGURE 3.6: Signal Modulation Classification Application

Chapter 4

Result and Discussion

In this section, we discuss the results of the signal modulation classification system designed in Section 2. We provide both empirical and graphical representations of the outcomes of the trained model during training and inference. Additionally, we present the results of our investigation into the impact of the receptive field on the designed CNN deep network. Finally, we present the timing and implementation report obtained from designing a deep learning processor for the best-performing model.

4.1 CNN Models

As highlighted in Section 3, three different CNN deep networks were trained and benchmarked to investigate the effectiveness of CNNs in performing signal modulation classification and to examine the impact of filter size on CNN networks designed for this task. We present the results in two phases: the Training Phase and the Inference Phase.

4.1.1 Training Phase

By training each CNN network for a total of 20 epochs (as highlighted in Section 3.5), we observed an upward trend in model accuracy and a steady decrease in model loss. Model I (see Table 3.5) achieved the highest training and validation accuracy; however, the accuracy plot during training, as shown in Figure 4.1, exhibited significant fluctuations. These fluctuations posed a challenge for the optimization procedure, as the stochastic gradient descent solver frequently had to shift from regions of lower accuracy to regions of higher accuracy in the model parameter space. Despite these fluctuations, an overall upward trend was still observed, indicating the effectiveness of the 1×2 filter size in

extracting salient features from various signals and modulation techniques, which was crucial for making accurate classification predictions.

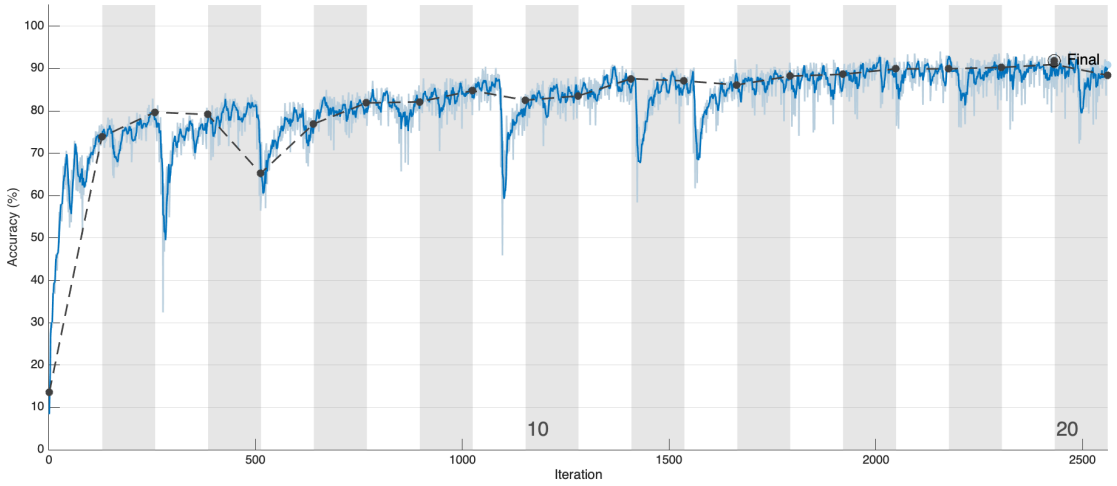

FIGURE 4.1: Model I Training Accuracy Plot

Model II yielded similar training classification accuracy to Model I. However, the accuracy plot for Model II had fewer sharp fluctuations, as shown in Figure 4.2. During training, the network's validation performance, which is assessed at the end of each epoch, maintained an overall upward trend with only occasional downward fluctuations.

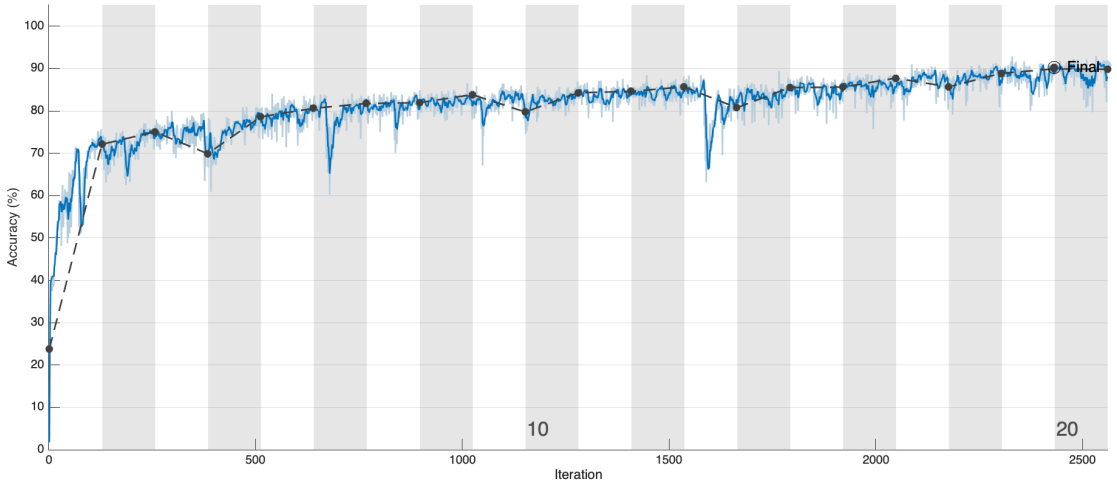

FIGURE 4.2: Model II Training Accuracy Plot

Model III, with a filter size of 1 x 8, exhibited a much smoother accuracy trend throughout training, with minimal fluctuations during the early epochs as shown in Figure 4.3. This consistent trend was also observed in the model's validation accuracy across epochs. Despite this, the model did not achieve accuracy above 90% within the training period. It is speculated that the network might reach higher accuracy if trained for a longer period; however, for benchmarking purposes, the same number of epochs was used for each model in the investigation.

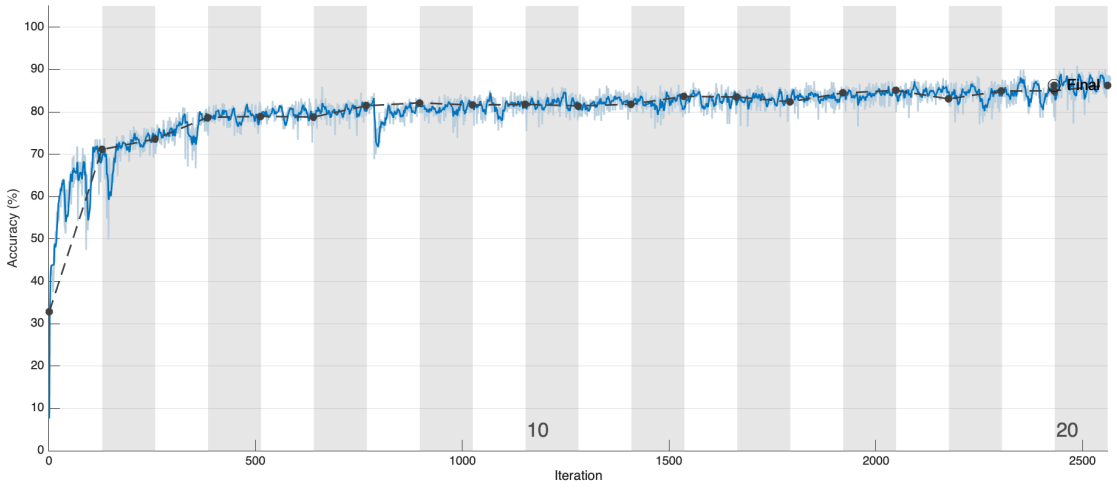

FIGURE 4.3: Model III Training Accuracy Plot

Model IV, the least performing model, exhibited only minor accuracy fluctuations during the training lifecycle, as shown in Figure 4.4. However, the network's accuracy struggled to gain significant upward momentum, resulting in a relatively low final training accuracy of approximately 83%.

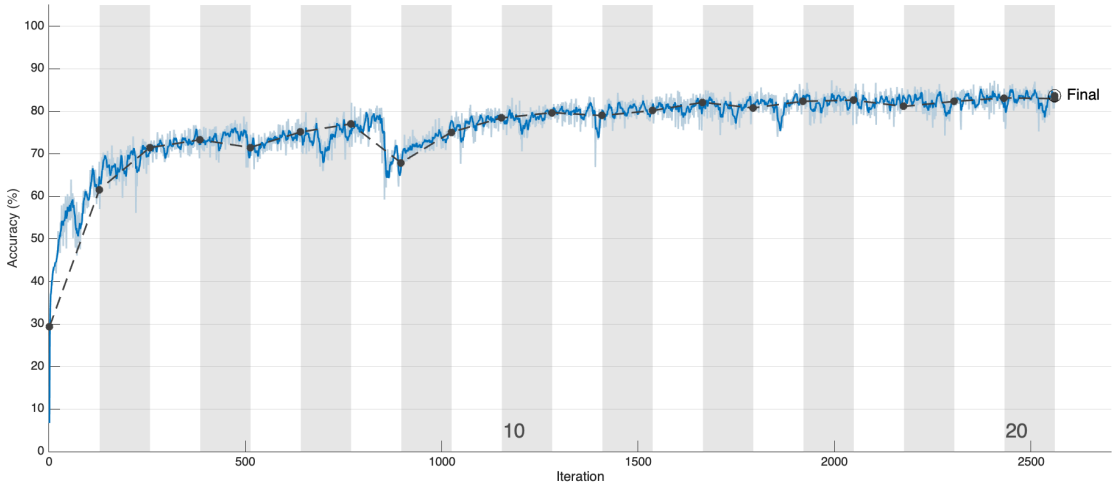

FIGURE 4.4: Model IV Training Accuracy Plot

Summary of the each model training details and performance is presented in Table 4.1.

4.1.2 Inferencing Phase

To properly evaluate the performance of the trained CNN models, we perform inference with the test set, which constitutes 10% of the RadioML dataset subset extracted for this project. The accuracy achieved on the test set was similar to that obtained during training. This indicates that the models did not overfit and are capable of generalizing

TABLE 4.1: Model Training Results

Model	Training Duration	Total Learnables	Accuracy
Model I	114min 2secs	25.7k	91.68%
Model II	102 mins 12secs	49.6k	90.19%
Model III	91 mins 33secs	97.3k	86.26%
Model IV	109 mins 59secs	192.8k	83.72%

their performance to previously unseen signal data. The confusion matrix plots for each trained network on the test set are provided in Figure 4.5,4.6,4.7,4.8 .

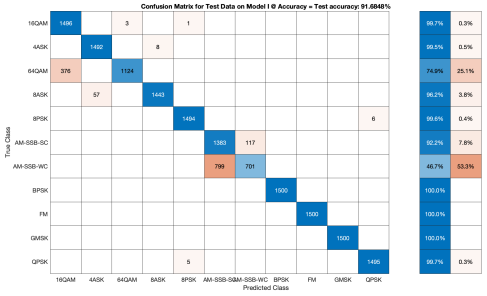

FIGURE 4.5: Model I (Accuracy 91.68%)

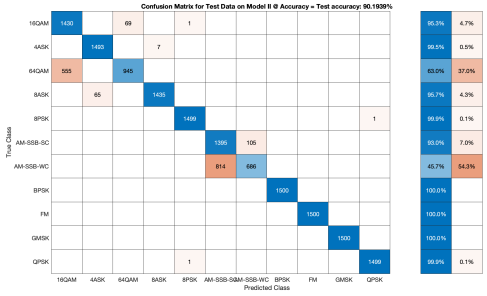

FIGURE 4.6: Model II (Accuracy 90.19%)

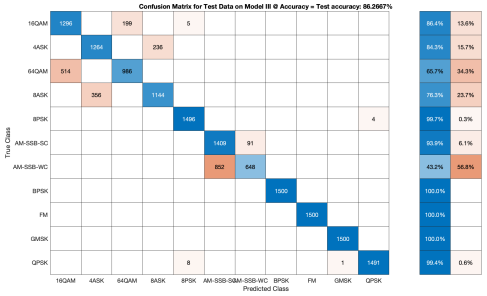

FIGURE 4.7: Model III (Accuracy 86.26%)

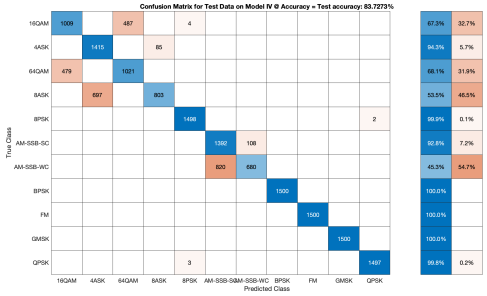

FIGURE 4.8: Model IV (Accuracy 83.72%)

Finally, we evaluate the performance of the model across all 15 SNR values, ranging from +2 dB to +30 dB. The resulting accuracy, as shown in Figure 4.9, exhibited a varied trend, with the lowest accuracy occurring at 4dB. Ideally, signals with lower SNRs are expected to yield lower accuracy; however, for the investigated SNR range, the accuracy results did not show a distinct upward trend.

4.1.3 Receptive Field Investigation

The receptive field, which refers to the patch area of the signal that contributes to a feature in each convolutional layer of the CNN, was found to be negatively correlated with the overall CNN network accuracy. Increasing the receptive field—achieved by enlarging the spatial dimensions of the convolutional layer filters—resulted in lower network classification accuracy, whereas reducing the filter size led to improved model performance. This observation suggests that the filter size determines the granularity with which the model can recognize unique features that distinguish different classes in the classification problem. The receptive field can also be interpreted as the model's resolution, with higher resolution (smaller filter sizes) yielding better performance. To illustrate the contrast in performance, we plot the accuracy of each network on the test set in Figure 4.10.

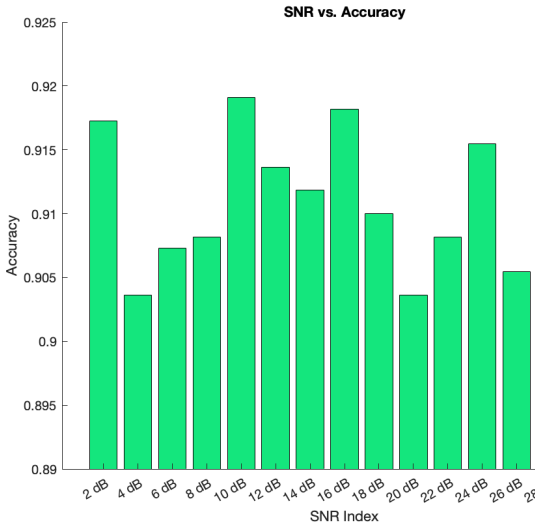

FIGURE 4.9: Signal SNR Versus Model Accuracy

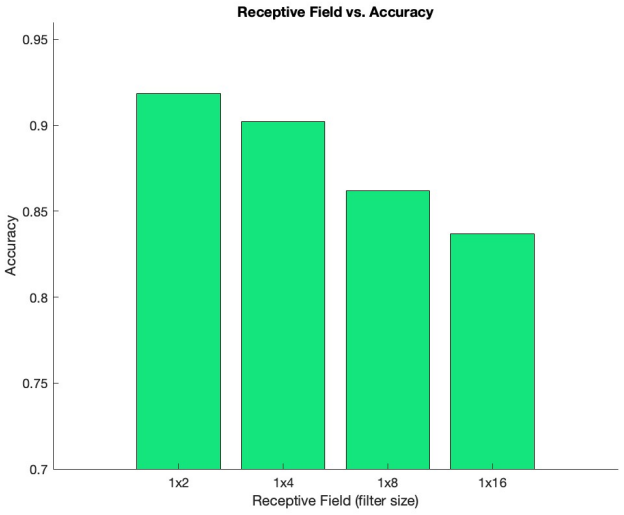

FIGURE 4.10: Model Accuracy Trend With Receptive Field

4.2 Deep Learning Processor

To evaluate the cost in terms of timing and resources of the deep learning network, we generated synthesizable HDL code using the MATLAB Deep Learning HDL Toolbox and MATLAB Code Generator. Before generating the HDL code, we optimized the configuration of the initialized deep learning processor, which includes predefined functional blocks for popular deep learning architectures. After optimization, we obtained an estimation of the processor's performance results, as shown in Table 4.3.

Table 4.2: Processor Performance Estimate

Property	Value
Total Latency (clk)	1467925
Clock Frequency (MHz)	100
Data Rate (Frames/second)	68.1

Table 4.3: Post Implementation Hardware Utilization Report

Resource	Utilization	Available	Utilization %
LUT	236554	274080	86.31
LUTRAM	33290	144000	23.12
FF	250539	548160	45.71
BRAM	555.50	912	60.91
IO	52	328	15.85

The total estimated latency, reported in terms of clock cycles, is 1,468,083 clock cycles, which is equivalent to 68.1 frames per second. This represents the total number of clock cycles required for the deep learning network hardware implementation to process a single input through the network. The time in seconds can be calculated using the following formula.

$$\text{Time (in seconds)} = \frac{\text{Latency (in clock cycles)}}{\text{Clock Frequency (in Hz)}}$$

$$\text{Time} = \frac{\text{1,467,925 clock cycles}}{100 \times 10^6 \text{ Hz}} = 0.01468 \text{ seconds or } 14.68 \text{ milliseconds}$$

A latency of 14.68 milliseconds may be sufficient for testing purposes, but to deploy this signal modulation classifier in spectrum sensing for cognitive radio networks using 4G LTE and 5G technology, the processing delay must be within 1–10 milliseconds for 4G LTE and 1–5 milliseconds for 5G networks [16]. Meeting these requirements is crucial to guarantee that the network's Quality of Service (QoS) will be met.

After the HDL code generation stage, a Vivado project containing the processor IP core is created. We manually synthesized the design on Vivado and executed the implementation to obtain the implementation report, which provides the most accurate estimate of the processor's performance when implemented in hardware. The resource utilization report and timing report are summarized in Tables 4.3 and 4.4, respectively.

The model weights and biases are represented using floating-point data types, which results in a larger hardware footprint and consequently higher resource utilization of

TABLE 4.4: Post Implementation Timing Summary

	Setup (ns)	Hold (ns)
Worst Negative Slack (WNS)	0.312	0.010
Total Negative Slack (TNS)	0.000	0.000
Total Number of Endpoints	833611	833023

LUTs. Using quantized fixed-point weights could improve resource utilization, though it may come at the expense of classification accuracy. Design tools like the Finn Project, introduced in Section 3.2, could be employed to train quantization-aware deep learning networks, reducing the impact of quantization on network performance while optimizing hardware implementation costs.

In terms of timing performance, the deep learning processor design was targeted for a clock frequency of 100 MHz. After implementation in Vivado, the resulting design timing summary is provided in Table 4.4.

The design as shown in Figure 4.11 meets all user-specified timing constraints with a worst negative slack (WNS) of 0.312 ns. This could be further improved by implementing a quantized network, which would reduce the critial path i.e. the longest combinatorial path in the design, thereby increasing the WNS. With a shorter critical path, we can operate the design at a faster clock speed, allowing the signal modulation classification system to meet the processing delay requirements for modern wireless technologies such as 4G LTE and 5G.

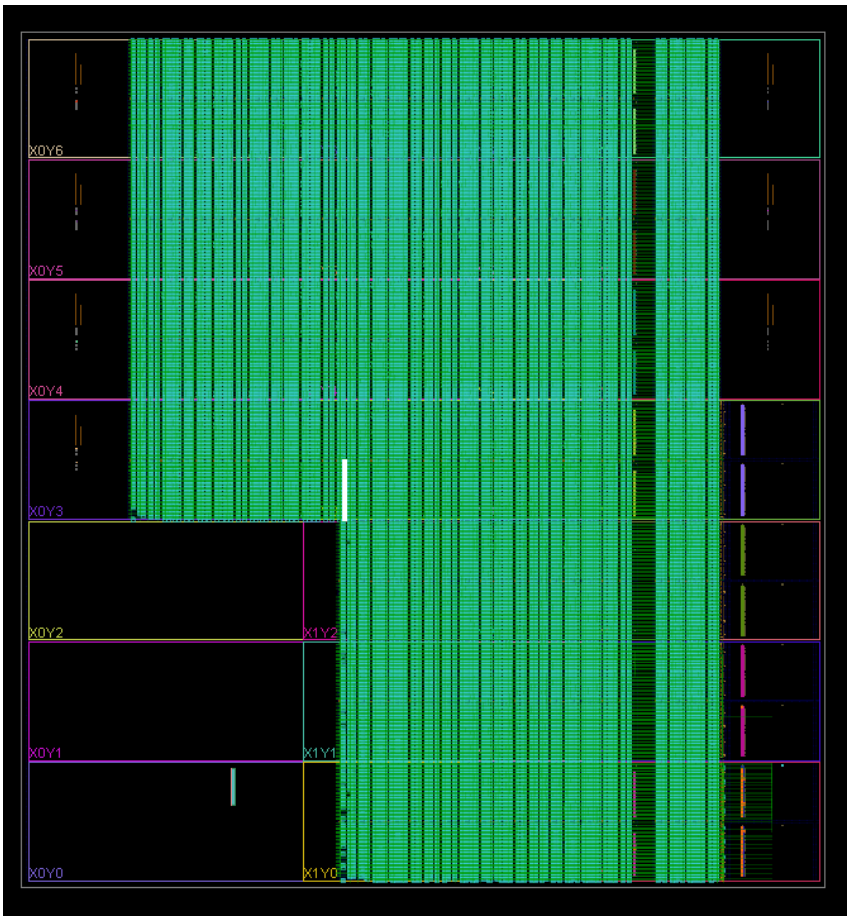

FIGURE 4.11: FPGA Routing Resources on Xilinx Zynq UltraScale+ Product Family

Chapter 5

Conclusions

So far, we have developed and trained a Convolutional Neural Network for classifying signals based on their modulation technique. Analysis of classification results on the test set indicates that, although CNNs are predominantly optimized for image recognition tasks, they are also highly effective in achieving significant classification accuracy for radio signal modulation detection problems.

Subsequently, to evaluate the practical application of our trained model, we designed a deep learning processor specifically optimized for our most accurate model. Our simulations demonstrated the processor's performance on a Xilinx Zynq Ultrascale+ device. Through iterative design optimizations, we successfully met timing constraints at a 100 MHz clock speed, providing a solid foundation for implementing large deep learning networks with floating-point weights and biases. Additionally, the implementation report offered insights into the FPGA hardware resource utilization for our design.

We can conclusively affirm that deep learning algorithms, such as CNNs, are likely to play a crucial role in spectrum sensing for future cognitive radio networks operating on shared spectrum. The deep learning approach represents a significant shift from traditional manual statistical methods reliant on handcrafted expert features to a more adaptive system that evolves in tandem with advancements in wireless network technologies. This technique also enables receivers in non-cooperative communication systems to autonomously identify the modulation technique employed by transmitters and dynamically adjust their parameters for accurate signal decoding with minimal signaling overhead. Such deep learning methods significantly streamline the process of granting cognitive radios the capability to properly interpret signals in dynamic spectrum environments, offering a promising trajectory for future wireless networks, including 5G and 6G technologies.

5.1 Future Directions

The following are some highlighted future direction that the application of deep learning for signal modulation classification in dynamic spectrum sensing application can take

CNN Model Complexity: To enhance CNNs' ability in detecting signal modulation schemes, we can implement deeper architectures with more convolutional and fully connected layers. These advanced deeper connections can leverage CNNs' capacity to extract even more complex, multi-dimensional features from signals, encompassing both spatial and temporal correlations crucial for classification. However, training very deep neural networks may lead to challenges such as vanishing or exploding gradients. To mitigate these issues, state-of-the-art techniques like ResNet, which utilizes skip connections, can be employed. By extending training duration, increasing epoch count, and incorporating diverse real-world signal datasets, we can anticipate CNNs achieving performance levels suitable for integration into existing radio systems. Finally, other deep learning architectures, such as Recurrent Neural Networks (RNNs) and Transformers, which are well-suited for sequential data processing and inference, can also be further applied to signal modulation classification tasks.

Hardware Optimization: As highlighted in Section 4.2, the hardware implementation cost of deep neural networks for signal modulation detection and classification is crucial for ensuring that designs are both efficient and environmentally friendly, particularly in terms of energy consumption for sustained operational hours. Our initial design, utilizing floating-point network weights, resulted in high utilization of LUTs and other logic and memory components. This resource consumption can be significantly reduced through weight and bias quantization, converting floating-point parameters to fixed-point representations. While this approach may impact model accuracy, tools like the FINN project (introduced in Section 3.2) enable the development of quantization-aware neural networks. These tools allow designers to specify per-layer dynamic ranges for network parameters during training, resulting in improved balance between precision and accuracy of the deep learning network. Additionally, quantization offers the benefit of increased clock speeds, reducing processing latency during inference in real-time cognitive radio systems.

References

[1] O. F. Abd-Elaziz, M. Abdalla, and R. A. Elsayed. Deep learning-based automatic modulation classification using robust cnn architecture for cognitive radio networks. Sensors, 2023.

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

[3] S. G. Bilén. Cognitive radio. The Advancing World of Applied Electromagnetics, 2024. URL https://www.springerprofessional.de/cognitive-radio/26956988. [Accessed: Aug. 16, 2024].

[4] P. Dhilleswararao, S. Boppu, M. Manikandan, and L. R. Cenkeramaddi. Efficient hardware architectures for accelerating deep neural networks: Survey. IEEE Access, 2022.

[5] S. S. Hadi and T. C. Tiong. Adaptive modulation and coding for lte wireless communication. IOP Conference Series: Materials Science and Engineering, December 2014.

[6] Huawei. 5g spectrum: Public policy position, February 2020. URL https://www.huawei.com/en/public-policy/5g-spectrum. [Accessed: Aug. 16, 2024].

[7] T. Huynh-The, V. Pham, T.-V. Nguyen, T. Nguyen, R. Ruby, M. Zeng, and D.-S. Kim. Automatic modulation classification: A deep architecture survey. IEEE Access, 2021.

[8] A. Jovičić and P. Viswanath. Cognitive radio: An information-theoretic perspective. IEEE Transactions on Information Theory, September 2009.

[9] J. Manyika and C. Roxburgh. The great transformer: The impact of the internet on economic growth and prosperity. McKinsey Global Institute, October 2011. URL https://www.mckinsey.com/

industries/technology-media-and-telecommunications/our-insights/the-great-transformer. [Accessed: Aug. 16, 2024].

[10] J. Mitola and G. Q. Maguire. Cognitive radio: making software radios more personal. IEEE Personal Communications, August 1999.

[11] A. J. Morgado et al. Dynamic lsa for 5g networks: the adel perspective. Proc. European Conference on Networks and Communications (EuCNC), 2015.

[12] A. Munir, J. Kong, and M. A. Qureshi. Accelerators for convolutional neural networks, 2024.

[13] Ofcom. Mobile networks and spectrum: Meeting future demand for mobile data - welsh overview. Discussion Paper, February 2022.

[14] T. O'Shea, T. Roy, and T. Clancy. Over the air deep learning based radio signal classification. IEEE Journal of Selected Topics in Signal Processing, 2017.

[15] T. J. O'Shea, J. Corgan, and T. C. Clancy. Convolutional radio modulation recognition networks. arXiv preprint, 2016. URL https://arxiv.org/abs/1602.04105.

[16] I. Parvez, A. Rahmati, I. Guvenc, A.I. Sarwat, and H. Dai. A survey on low latency towards 5g: Ran, core network and caching solutions. IEEE Communications Surveys & Tutorials, 2018. URL https://doi.org/10.1109/comst.2018.2841349. [online].

[17] M. Parvini et al. Spectrum sharing schemes from 4g to 5g and beyond: Protocol flow, regulation, ecosystem, economic. IEEE Open Journal of the Communications Society, 2023.

[18] Z. Quan, S. Cui, A. Sayed, and H. V. Poor. Wideband spectrum sensing in cognitive radio networks. Proc. IEEE International Conference on Communications (ICC), May 2008.

[19] S. Shi et al. Challenges and new directions in securing spectrum access systems. IEEE Internet of Things Journal, April 2021.

[20] C. Tellambura and S. Kusaladharma. An overview of cognitive radio networks. Encyclopedia of Wireless and Mobile Communications, 2017.

[21] A. M. Wyglinski, M. Nekovee, and Y. T. Hou. Cognitive radio communications and networks: Principles and practice. Academic Press, 2010.