MACHINE LEARNING ASSISTED WIRELESS COMMUNICATION PLATFORM

機械学習支援ワイヤレス通信プラットフォーム

DODY ICHWANA PUTRA

Contents

List of Tables

List of Figures

List of Abbreviations

3GPP	3rd Generation Partnership Project
5G	Fifth Generation
6G	Sixth Generation
AMC	Automatic Modulation Classification
API	Application Programming Interface
AWGN	Additive White Gaussian Noise
BPSK	Binary Phase Shift Keying
CCA	Clear Channel Assessment
CCA-ED	CCA with Energy Detection
CCA-SD	CCA with Signal Detection
DAC	Digital-to-Analog Converter
DL	Deep Learning
DMA	Direct Memory Access
E2E	End-to-End
EDA	Electronic Design Automation
FSM	Finite State Machine
FMC	FPGA Mezzanine Card
FPGA	Field-Programmable Gate Array
Gbps	Giga byte per second
HE	High Efficiency
HT	High Throughput
HDL	Hardware Description Language
HW	Hardware
IP	Internet Protocol
LAN	Local Area Network
LTE	Long-Term Evolution
MAC	Media Access Control
MAN	Metropolitan Area Network

Mbps	Mega byte per second
MCS	Modulation and Coding Scheme
MHz	Mega Hertz
ML	Machine Learning
MIMO	Multiple Input Multiple Output
MTL	Multi-Task Learning
MPDU	MAC Protocol Data Unit
MSDU	MAC Sevice Data Unit
MU	Multi-User
NF	Noise Figure
OFDM	Orthogonal Frequency Division Multiplexing
PC	Personal Computer
PHY	Physical Layer
PLCP	Physical Layer Convergence Protocol
PPDU	PLCP Protocol Data Unit
PSK	Phase Shift Keying
QAM	Quadrature Amplitude Modulation
RF	Radio Frequency
RU	Resource Unit
Rx	Receiver
SDR	Software-Defined Radio
SIG	Signal interval guard
SINR	Signal-to-Interference-plus-Noise Ratio
SNS	Social Networking Service
SNR	Signal-to-Noise Ratio
STA	Station (in the context of WLANs)
SW	Software
UL	Uplink
UDP	User Datagram Protocol
USRP	Universal Software Radio Peripheral
USHWAP	Unified Software and Hardware Wireless-AI Platform
VHT	Very High Throughput
WLAN	Wireless Local Area Network
Wi-Fi	Wireless Fidelity

Summary

Recently, the global demand for wireless communication networks has been increasing. Existing wireless networks rely on mathematical models to define their communication structure. While conventional signal processing is effective, it has limitations, especially with complex computations at the physical layer. Optimizing networks requires complex mathematical solutions, which can be inefficient. Machine learning (ML) provides a data-driven solution to these challenges. Integrating ML into wireless system applications has become extensive research. This growing trend results in the need for a comprehensive platform for designing wireless AI applications. Developing ML applications in a wireless system requires a platform including software (SW) and hardware (HW) components. Researchers often use multi-SW to build and evaluate their intelligent algorithms, and then HW platforms are used for real-time testing. It presents challenges in the development process of wireless AI applications. A unified SW and HW platform offers a flexible solution to address this.

This thesis proposes the Unified Software and Hardware Wireless-AI Platform (USHWAP) to facilitate machine learning integration within wireless systems. USHWAP supports various ML applications through SW simulation and HW implementation for testing. It integrates high-level SW (MATLAB, Python) with field-programmable gate array (FPGA) and software-defined radio (SDR), offering unified connectivity for scalable devices and efficient data transfer, making it suitable for device and edge computing. This connectivity eliminates the need for additional MATLAB toolboxes, simplifying the development process for both SW and HW applications. The contributions of USHWAP include capabilities for multi-application development, multi-SW simulation, a generic and flexible design with unified SW/HW connectivity, and high throughput with low latency performance.

To demonstrate the effectiveness and use-cases of the proposed USHWAP, multi-task learning (MTL) with CNN is employed for packet format detection and modulation recognition within wireless LAN (WLAN) signals. The development and testing of this application utilize the proposed USHWAP. This approach allows STAs to discern channel occupancy even without the preamble, improving the coexistence of 5G and Wi-Fi networks.

MTL-CNN performs sophisticated classifications of received signals, enabling STAs to manage transmission timing adeptly. This model complements the existing clear-channel-assessment (CCA) method, achieving 99% accuracy in packet detection and 84.7% in signal modulation classification. This innovative approach shows promise in significantly enhancing the reliability and efficiency of wireless communication systems.

Chapter 1

Introduction

1.1 Motivations

Recently, the global demand for wireless communication networks has been increasing. This growth is attributed to the increasing number of wireless users and the emergence of novel wireless services. The evolution of wireless network systems over decades has introduced in new services for users facilitated by innovative network and device technologies [3]. As wireless communications have advanced, their utilization has become widespread across various sectors, including social networking service (SNS), entertainment, healthcare, industry, education, and military.

IEEE 802.11 Wireless Local Area Networks (WLANs) or Wi-Fi play a dominant role in enabling wireless communication. Their popularity arises from well-defined use cases, adaptability, and broad device compatibility. The 802.11 protocol family undergoes regular updates. For instance, Wi-Fi 6 [5] [6] [7] has emerged as the new standard for recent consumer products, and the development of Wi-Fi 7 [8] [9] [10] is already underway. Concurrently, 5G technology, with its cutting-edge features, leads the mobile technology revolution. The emergence of 6G networks is driven by the demand for services that surpass the already heightened standards set by 5G regarding reliability, security, and efficiency.

In a conventional wireless communication system structure, the signal processing block comprises several consecutive processes, including modulation, channel encoding, channel

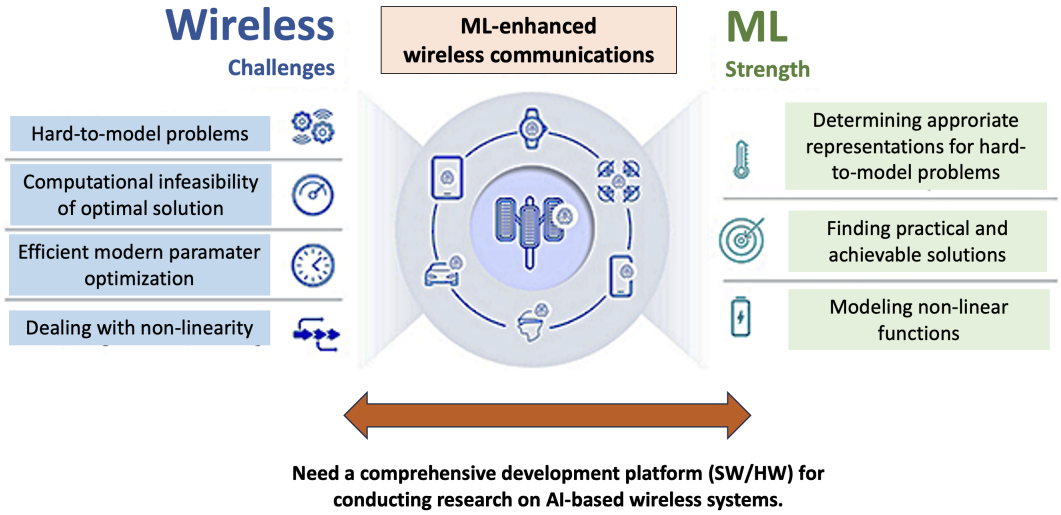


Figure 1.1: Addressing wireless challenges, leveraging ML strength, and needing a comprehensive development platform (SW/HW) in ML-based wireless systems research.

estimation, and data detection. Present wireless networks heavily depend on mathematical models to define their communication structure. This conventional signal processing technologies block has excelled in practical wireless communication systems. However, they are limited, especially when confronted with the heightened computation complexity technology at the physical layer. These conventional models often fall short of providing accurate representations of the systems. Additionally, some essential components of wireless networks and devices need comprehensive mathematical models, making their modeling challenging. The optimization of wireless networks also demands complex mathematical solutions, which can be inefficient in terms of both computational time and energy consumption. Consequently, relying solely on these mathematical models and solutions is unlikely to significantly enhance the capacity and performance of wireless networks.

In response to these challenges, artificial intelligence (AI) solutions have emerged as pioneering data-driven signal processing technology [11]. As part of AI, machine learning (ML) can approximate complex mathematical functions, demonstrating exceptional effectiveness and efficiency in addressing intricate and extensive issues [12]. Specifically, deep learning (DL), as an ML algorithm, can be employed both as conventional signal processing modules and for enhancing their performance by integrating domain-specific data

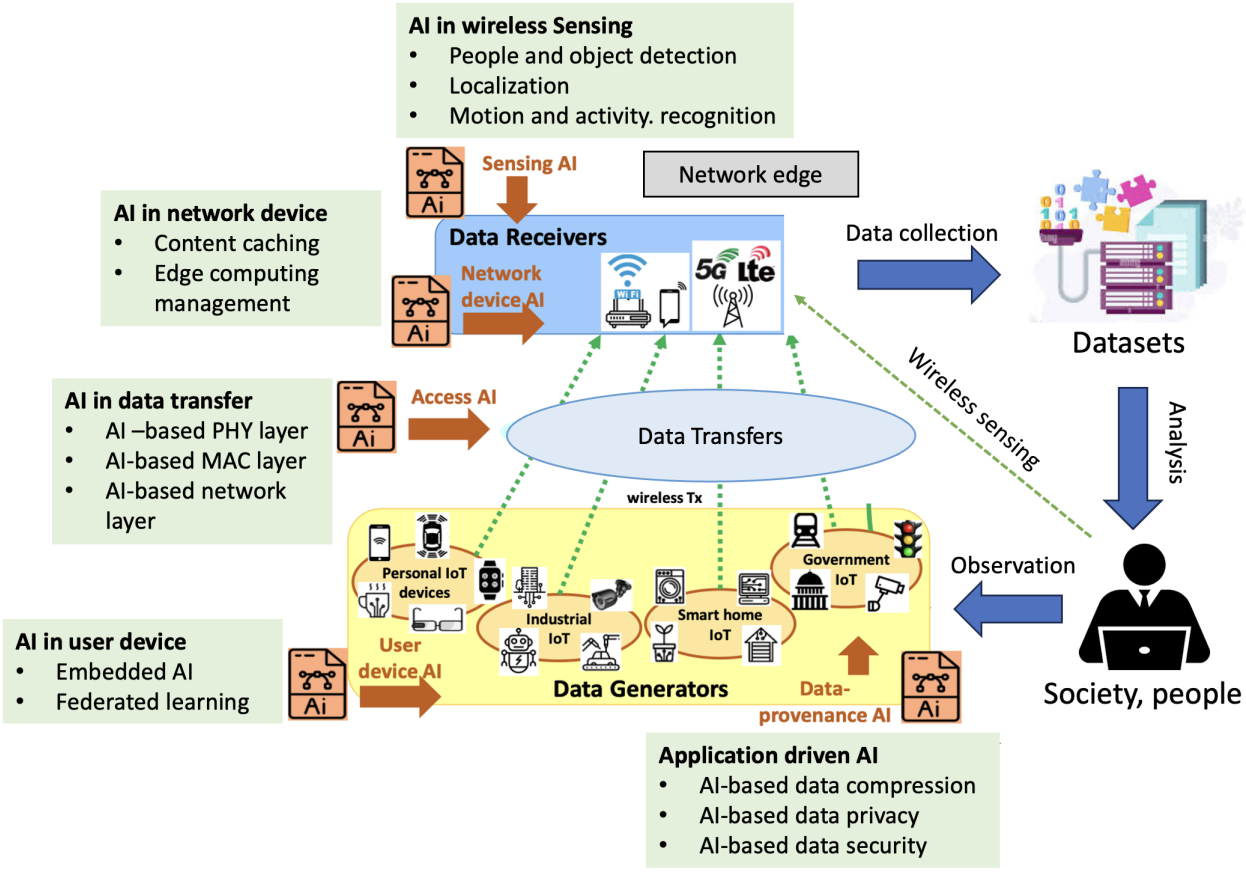


Figure 1.2: AI applications in wireless network.

insights [13] [14]. Based on ML, signal processing modules leverage DL to supplant or augment conventional signal processing modules. DL addresses optimization challenges through pattern recognition, especially in signal processing. DL acts as an alternative solution or an additional computational component designed to optimize specific segments of the physical layer. Figure 1.1 illustrates the wireless challenges and leveraging ML strength.

The integration of AI into wireless system applications has become an extensive area of research, as depicted in Figure 1.2. A comprehensive platform encompassing software (SW) and hardware (HW) components is essential for developing AI applications within wireless systems. This integrated platform empowers researchers to construct and assess

the performance of their intelligent algorithms within wireless systems. The open AI libraries like TensorFlow, Keras, and PyTorch, integrated with Python, speed up the process of building and testing intelligent algorithms. Additionally, many researchers turn to MATLAB Simulink toolboxes, including WLAN and signal processing toolboxes, to develop algorithms for wireless systems. To ensure the algorithm's viability, validating its performance in a real-world scenario is critical. This often necessitates deployment on a real-time HW testbed within a laboratory environment. Consequently, the utilization of multi-SW and HW platforms for development and testing poses a common challenge in wireless-AI application development.

This thesis proposes a unified SW and HW co-working for developing wireless AI applications to address this challenge. This platform allows for the flexible development of both SW and HW components on a single platform, eliminating the need for separate platforms for each. The versatility of such a unified platform in accommodating various research scenarios streamlines the process for researchers to create and evaluate new AI models within wireless systems.

Moreover, the research to investigate the application of Convolutional Neural Networks (CNNs) for packet format detection and modulation recognition within WLAN signals poses a challenge due to the intensive growth of WLAN [15] [16]. In the realm of IEEE 802.11, stations (STAs) rely on Clear Channel Assessment (CCA) to ascertain channel occupancy. Channel occupancy is the condition of the communication channel to indicate whether STA currently uses the channel or is available for transmission. Understanding channel occupancy is essential to avoid collisions and interference transmission. When the STA transmits the packet on the occupied channel, it results in data loss and reduced network performance.

CCA indicates an occupied channel, and STAs judiciously manage their back-off counters. Conversely, in idle states, they commence the countdown. Upon detecting a packet's preamble, the STA proceeds to process the header (preamble) and data symbols for signal classifying [2]. Classifying received signals is vital for deciding how and when to transmit the packet. The received signals can be classified by their packet frame, modulation, signal strength, and noise level. This information is essential for optimizing transmission

parameters to avoid collision and interference. However, challenges arise when the preamble is absent and received power falls below the prescribed threshold. This scenario often results in packet collisions [17]. Furthermore, enhancing the coexistence of 5G New Radio Unlicensed (NR-U) networks and next-generation Wi-Fi (known as Wi-Fi 6E) is essential, as they will operate in the same 6 GHz frequency range. This potentially may create interference issues affecting both Wi-Fi and 5G performance."

In this thesis, we also propose multi-task learning (MTL) with convolutional neural networks (CNNs) for packet format detection and modulation recognition within WLAN signals. The proposed unified SW and HW co-working platform is utilized to develop this application. This method enables stations (STAs) to discern channel occupancy, even without the preamble. Furthermore, this model improves the coexistence of 5G NR and Wi-Fi networks. MTL-CNN conducts sophisticated classifications of received signals, allowing STAs to manage transmission timing effectively. This pioneering approach holds promise for significantly enhancing the reliability and efficiency of wireless communication systems.

1.2 Research Objectives

The proposed works in this thesis specifically deal with assisted machine learning in the physical layer of the wireless communication system. The objectives of this thesis are to present:

1. Designing a SW/HW co-working platform for wireless-AI development and testing to facilitate AI within wireless systems. This platform is intended to enhance the integration of ML in wireless technology. This platform accommodates various SW options, including MATLAB and Python, for simulation and employing real-time HW, such as field-programmable gate array (FPGA) and software-defined radio (SDR) for implementation and testing.

2. Employing MTL-CNN for packet format detection and modulation recognition of WLAN signals when the preamble part is missed. This method complements the

existing CCA method, effectively preventing packet collisions and improving the coexistence of 5G NR and Wi-Fi networks.

The first topic focuses on developing a SW/HW co-working platform to facilitate ML in wireless system application development. Given the increasing integration of ML in wireless system applications, it is crucial for researchers to have a development platform that seamlessly incorporates ML. Therefore, we have designed a Unified SW/HW Wireless-AI Platform (USHWAP) to develop diverse wireless AI applications. This platform combines multi-SW simulation tools, including MATLAB and Python, with HW platforms like FPGA and SDR. The platform serves as a SW/HW co-working platform for various wireless AI applications and multiple device connectivity. It enables researchers to select their preferred development SW for simulation and HW implementation.

The second topic discusses the development of MTL-CNN for packet format detection and modulation recognition of WLAN signals. It leverages the USHWAP to showcase the effectiveness of the proposed platform. In IEEE 802.11, STAs utilize CCA to assess channel availability. When CCA indicates the channel is busy, STAs maintain their back-off counters. Conversely, in an idle state, they begin decrementing the counter. Upon detecting a packet's preamble, the STA processes the header and data symbols. However, CCA's effectiveness decreases when the PHY preamble is absent and received power is below the threshold. This can lead to packet collisions. The study proposes MTL-CNN as the solution to mitigate this. MTL-CNN enables STAs to discern occupied channels, even without the preamble. It also classifies received signals for packet format and modulation type, allowing STAs to manage transmission timing effectively. Furthermore, MTL-CNN enhances the coexistence of 5G NR and Wi-Fi networks by determining packet frame information. MTL-CNN is developed and tested using the proposed USHWAP. MATLAB generates the datasets while training the MTL-CNN model is conducted in a Python environment. To assess this model, FPGA and SDR are employed to test using the actual signals.

1.3 Thesis Organization

The structure of this thesis is depicted in Fig. 1.3 The first chapter describes the motivations and objectives of the research tasks. The remaining chapters are organized as follows.

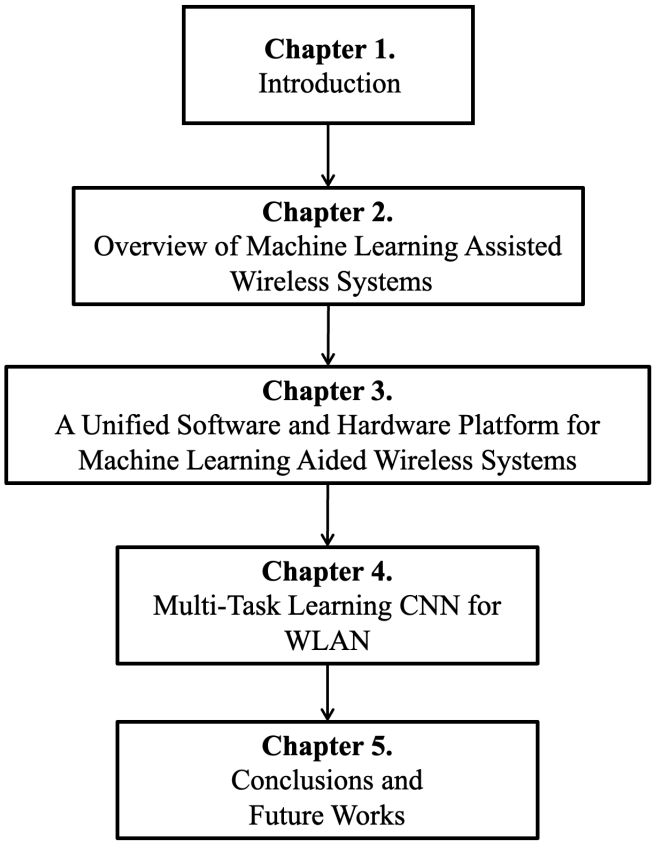


Figure 1.3: Thesis organization.

Chapter 2. Overview of Machine Learning Assisted Wireless Systems and the Wireless-AI Platform

This chapter presents an overview of ML-assisted wireless communication systems and introduces a wireless-AI platform to facilitate ML within these systems. The review aims to demonstrate the potential advantages of integrating ML into wireless communication systems and emphasizes its applications. The chapter also briefly reviews the related theory used in this thesis. Additionally, the critical features of the wireless-AI platform are assessed to offer insights and establish a foundation for further research endeavors.

Chapter 3. A Unified Software and Hardware Platform for Machine Learning Aided Wireless Systems

This chapter introduces a SW/HW co-design platform to develop diverse wireless-AI applications. It begins by defining the scope of the platform, followed by an overview of its general architecture, which combines SW simulation with HW real-time implementation and incorporates AI-based support. The proposed platform's performance is evaluated based on data throughput to demonstrate its capabilities. Additionally, we present three use cases to highlight the versatility and flexibility of this platform.

Chapter 4. Multi-Task Learning CNN for WLAN

This chapter introduces MTL-CNN for packet detection and modulation recognition in WLAN signals, complementing the CCA method used to evaluate the presence of WLAN signals and improve the coexistence of 5G NR and Wi-Fi networks. This chapter explains how the dataset is created. Following this, the architecture of the MTL-CNN model is detailed. Finally, simulation results and actual testing are presented using USHWAP to demonstrate the MTL-CNN capability to detect packets and recognize WLAN signal modulation.

Chapter 5. Conclusions and Future Works

This chapter presents the summary of the works and the achievable results. In addition, directions and suggestions on future research tasks are provided.

Chapter 2

Overview of Machine Learning Assisted Wireless Systems

This chapter provides an overview of ML-assisted wireless communication systems and the platform for a wireless-AI designed to facilitate ML within these systems. The review aims to show the potential benefits of integrating ML within the wireless communication system, highlighting the development platform and its applications. Additionally, an assessment of the critical features of the wireless-AI platform is provided to offer insights and establish a foundation for further research endeavors.

2.1 Machine Learning

2.1.1 Machine Learning Overview

Machine Learning (ML) involves the examination of data samples to draw fundamental conclusions through mathematical and statistical methods, enabling machines to learn without the need for explicit programming. The fundamental principle of ML is to acquire knowledge from data to predict or make decisions based on the given task [18]. The increasing computational power and data storage capacity have made it easier to train data-driven machine learning models, achieving predictions with nearly perfect accuracy. The ML algorithms are typically categorized into three groups: supervised, unsupervised, and

Figure 2.1: Various types of ML algorithms [1].

semi-supervised [18]. Nevertheless, ML algorithms can be further segmented into subgroups based on distinct learning approaches, as illustrated in Fig. 2.1.

Deep Learning (DL) constitutes a subset of ML that utilizes multiple layers to extract information, both at higher and lower levels, from input data (such as images, numerical values, and categorical values). DL deals with artificial neural networks with multiple layers (hence the term ”deep”). Modern DL models are constructed using Artificial Neural Networks (ANN), with Convolutional Neural Networks (CNN) being particularly prominent and often combined with other DL models, including generative models. DL can be

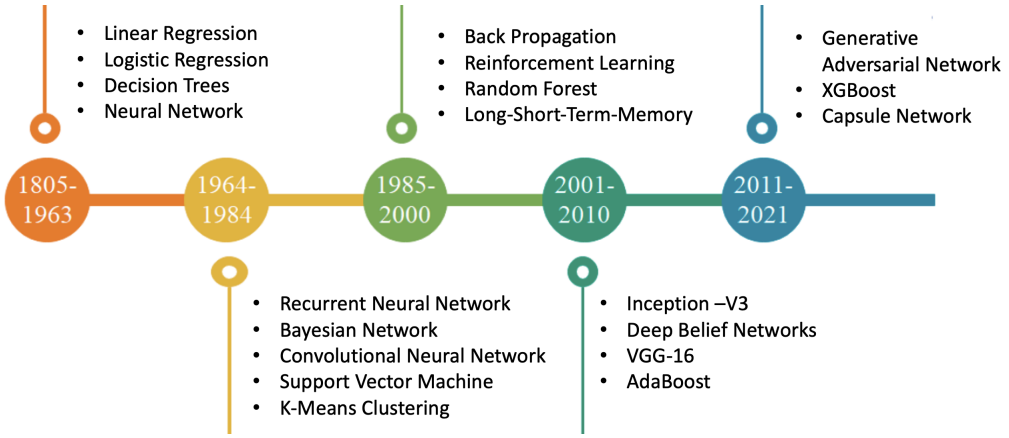


Figure 2.2: ML and DL algorithms development timeline.

categorized into three types: supervised, semi-supervised, and unsupervised. A schematic overview of ML and DL algorithms and their chronological development is presented in Fig. 2.2, a helpful reference for future researchers.

2.1.2 Machine Learning for Wireless Communication Systems

Artificial intelligence (AI), specifically ML and DL, has significantly advanced wireless communications. As the specialization of ML, DL has demonstrated its effectiveness in constructing signal-processing algorithms for demanding engineering tasks. Instead of relying on human-defined models, these algorithms excel by autonomously grasping the essential attributes of input signals.

Conventional wireless system design adheres to the ”model-based”, model-first-then-optimize approach. This entails creating a mathematical model of the design problem and estimating the model parameters. The optimal solution is obtained through mathematical programming [19]. In a conventional wireless communication system, the signal processing block comprises several consecutive processes, including modulation, channel encoding, channel estimation, and data detection. These processes are designed and optimized independently, which only sometimes ensures the system's overall optimal performance. Consequently, the disparity between theory and practical implementation has inspired numerous researchers to investigate intelligent communication methods, such as estimation,

detection, and optimization theory [20].

Conventional signal processing technologies have indeed excelled in practical wireless communication systems. However, they are limited, especially when confronted with the heightened computation complexity technology at the physical layer.

1. Inadequate precision in the model:

   Conventional signal processing relies on precise mathematical models to describe the physical layer's intricacies and signal impact. However, obtaining accurate models becomes challenging with growing system complexity [21], particularly in addressing hardware impairment effects influenced by various factors. In such cases, DL-based methods provide a compelling solution, excelling in extracting critical features from raw data and accurately approximating nonlinear functions. As a result, these methods have been integrated into the physical layer to enhance specific aspects of conventional communication systems.

2. Limited computational scalability:

   Conventional signal processing relies on mathematical optimization but struggles with scalability in large networks [22]. For example, massive multiple input, multiple output (MIMO) systems using numerous antennas pose computational challenges for traditional methods when optimizing antenna selection and beamforming. On the other hand, DL excels in handling large volumes of data and sequences by adding adaptable modules like convolutional and recurrent layers, mitigating scalability issues.

3. Limitation of global optimization:

   Conventional wireless communication systems face challenges in optimizing various functional elements, such as modulation, demodulation, coding, decoding, channel estimation, and signal detection. These components often have conflicting objectives, complicating overall optimization. For example, in orthogonal frequency division multiplexing (OFDM) systems, tasks like reducing peak-to-average-power ratio and enhancing signal modulation are crucial but conflicting [20]. Traditional optimization of these tasks separately is complex, and achieving global optimality

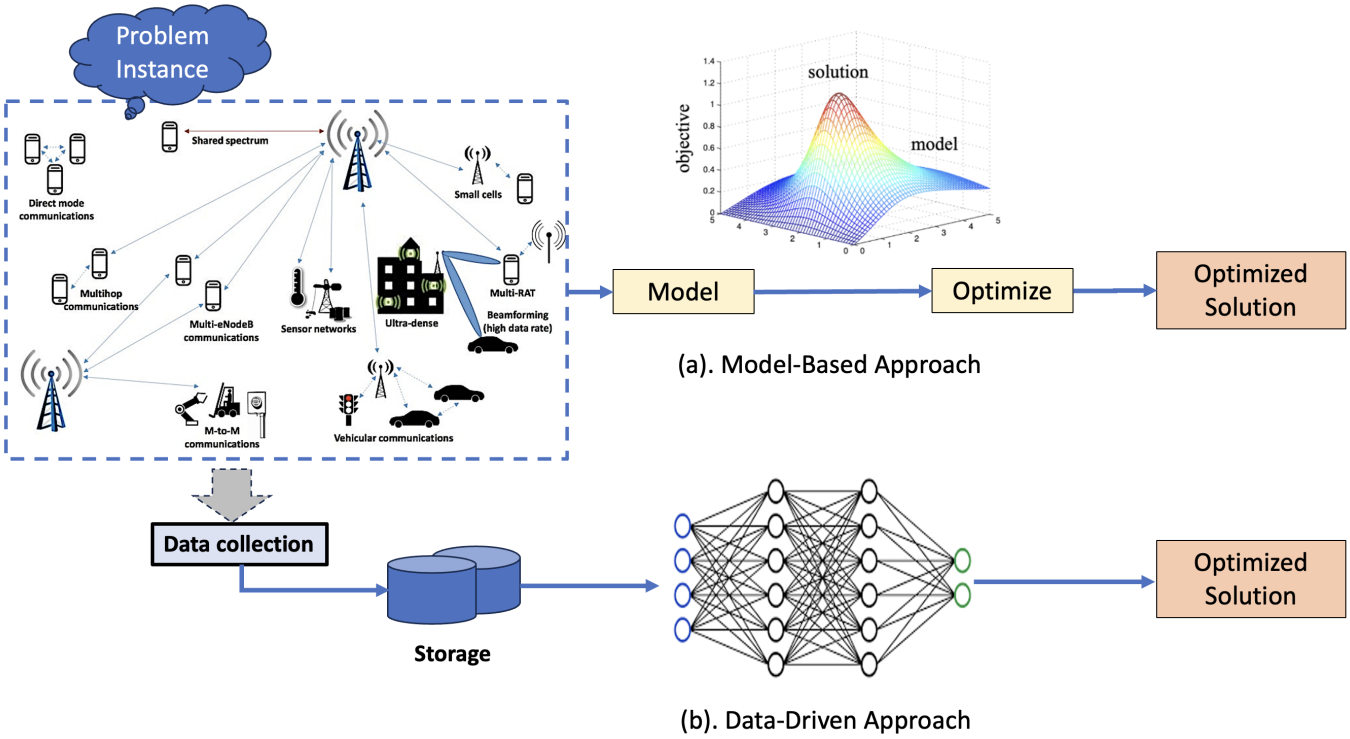


Figure 2.3: (a). Conventional model-based approach; (b). Data-driven based approach

is challenging. DL approaches can directly learn solutions to these optimization problems using training data, providing a more efficient method than conventional model-based strategies.

In response to these challenges, a cutting-edge solution has emerged in the form of ML as a pioneering data-driven signal processing technology [11]. The ML strategy seeks to acquire the best solution by representing the specific problem instance based on available data as ”data-driven-based”. Through extensive training on numerous instances of the problem, the neural network refines its weights based on the system's overarching objective, which is determined by the representation of the specific problem instances [20]. Fig. 2.3 shows the model-based approach and data-driven approach. As a specialized form of ML, DL can approximate complex mathematical functions, resulting in its exceptional effectiveness and efficiency in addressing intricate and extensive issues [12]. DL can be employed both

as conventional signal processing modules and for enhancing their performance by integrating domain-specific data insights [13] [14]. Furthermore, they possess the potential to replace the entire transceiver structure, thereby enabling comprehensive optimization from an End-to-End (E2E) standpoint [23]. Based on ML, signal processing modules leverage DL to supplant or augment conventional signal processing modules. DL addresses optimization challenges through pattern recognition. This starkly contrasts the conventional model-driven approach based on information theory in the design of communication systems. DL acts as an alternative solution or an additional computational component designed to optimize specific segments of the physical layer.

2.2 SW/HW Co-design Platform for Wireless-AI Design

This section will discuss various platforms for designing and testing wireless systems. The essential requirements for creating a platform for machine learning-assisted wireless applications will also be examined.

2.2.1 Platform for Wireless System Design

Based on the implementation platform, the platforms for wireless systems design can be classified into three primary categories: the full-software approach, the FPGA prototyping approach (hardware-based), and the hybrid software(SW)-hardware(HW) approach, which combines elements of both (SW/HW co-design).

1. Software-based Platform

The full SW-based platform for designing wireless systems includes implementing the system in SW on the host PC. This approach allows the entire design and testing process to occur in a SW without requiring specialized HW components. The main advantage of this approach is its flexibility. Researchers/engineers can easily modify developed algorithms and customize wireless system parameters without the limitations of dedicated HW. Furthermore, the full SW-based platform is more cost-effective, eliminating the need for expensive specialized HW.

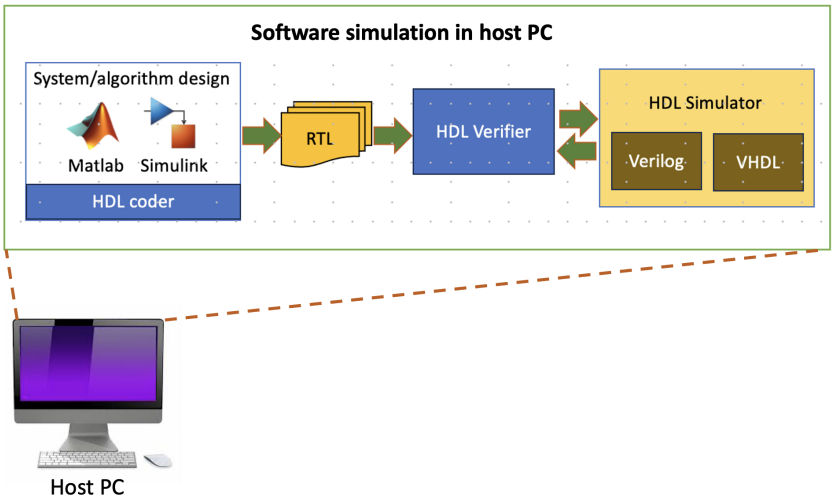


Figure 2.4: SW-based platform.

Additionally, the full software-based platforms enable rapid prototyping. Since no special HW is required, prototypes can be built and tested quickly with readily available computing resources [24]. This allows for faster development cycles, allowing researchers to iterate and improve repeatedly. These platforms often provide powerful simulation tools that facilitate modeling and analysis of system performance under various conditions.

However, this platform approach has limitations in real-time testing and cannot be applied to represent actual scenarios accurately. Examples of SW-based platforms for wireless system design and simulation environments include MATLAB/Simulink, as depicted in Fig. 2.4.

2. Hardware-based (FPGA-prototyping) Platform

The second approach involves utilizing an HW-based platform for wireless system design, employing an HW emulator like an FPGA prototype, as shown in Fig. 2.5. This method harnesses specialized integrated circuits that can be reconfigured to execute tasks relevant to wireless system functionality. It integrates a testbench and Design Under Test (DUT) into the HW platform.

A test bench serves as a simulation environment within electronic design automation

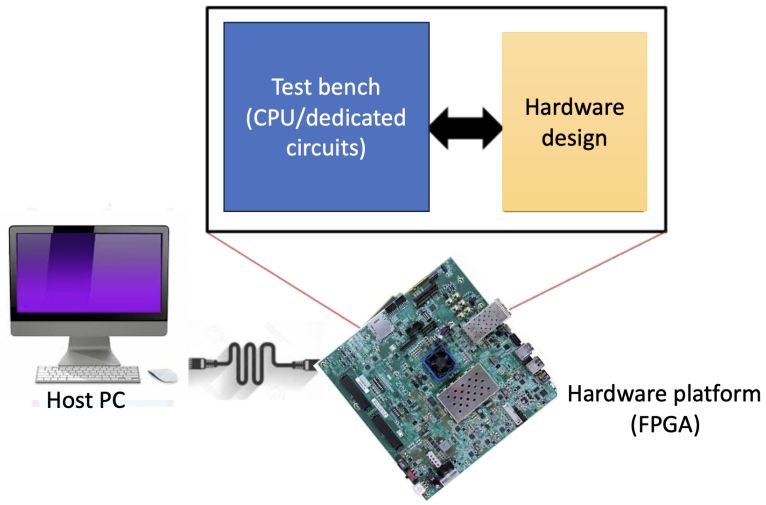


Figure 2.5: HW-based platform.

(EDA) that verifies the accuracy and functionality of a digital circuit or system. It comprises a suite of programs and scripts that generate input signals, apply them to the DUT, and subsequently monitor and analyze the resulting output responses. The testbench sequence may be implemented in a dedicated circuit or executed as code within an on-circuit CPU, while the DUT is entirely realized in an HW resource block. Connectivity to a host PC is only necessary for the initial setup of the system configuration, such as writing bit configurations into the hardware target [24].

FPGA prototyping has gained prominence as a tool for emulating complex systems due to its significant advantage in drastically improving verification time. The main advantage of FPGA is in real-time processing. It is important for applications where low latency and high-throughput performance are required. Moreover, FPGA-based verification allows for simulation at a cycle-accurate level. The FPGA prototype supports hardware-in-the-loop testing, enabling specific hardware components or subsystems to be tested within the broader context of the entire wireless system.

However, FPGA prototype implementations come with constraints, including limited debugging devices and a lack of flexibility for signal observability. To achieve high observability, researchers must invest considerable effort in developing and adding a dedicated circuit to capture various signals [25]. They use HW description languages like Verilog or

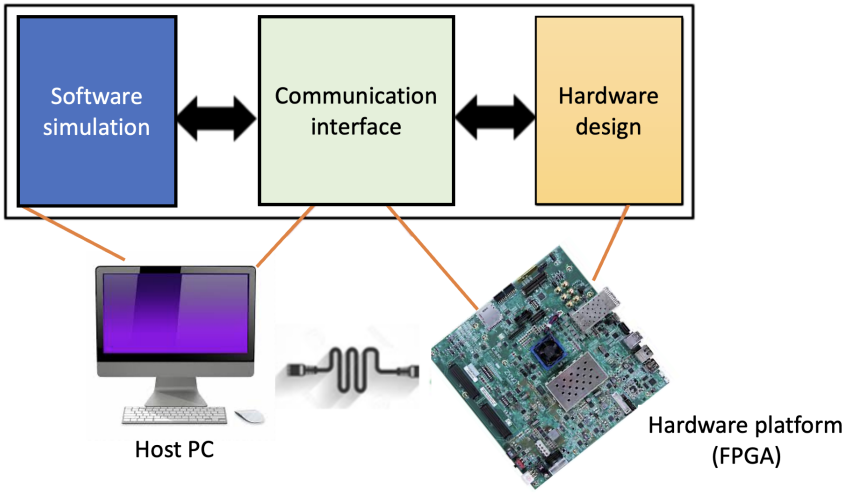


Figure 2.6: SW/HW co-design platform.

VHDL to program FPGAs, allowing for precise configuration of the HW logic. However, this may require significant effort from some researchers. Overcoming this challenge requires seasoned expertise and extensive time investment in system development in the lab until it reaches a state ready for verification. As a result, this verification platform is better suited for the later stages of system development, serving as a final product outcome rather than an integral component of the development process.

3. Software-Hardware Co-design Platform

With the increasing complexity of embedded system design, particularly in wireless systems, the utilization of purely SW-based or HW-based platforms for development and testing platforms has become less frequent [26]. This trend is particularly notable in systems involving intensive SW/HW co-design processes. Consequently, development platforms have evolved towards frameworks incorporating SW and HW verification, as shown in Fig. 2.6. This shift is prompted by the realization that task processing is no longer solely dependent on HW components but also involves significant contributions from SW elements. A SW/HW co-design platform for wireless system design combines both SW-based and HW-based approaches, resulting in a comprehensive environment for development and testing.

The characteristics of a SW/HW co-design platform include:

- **Flexibility and Rapid Prototyping:** Flexibility and Rapid Prototyping: The SW component allows rapid prototyping and is easy for algorithm modification and system parameters setting. This capability is essential in the early stages of system design when frequent iterations and experimentation are common.
- **Simulation Capabilities:** The platform provides powerful simulation tools that enable the modeling and analysis of system performance in various scenarios.
- **Real-Time Processing:** The HW component (FPGAs) is powerful for real-time processing tasks, essential for applications requiring low-latency and high-throughput performance.
- **Hardware-in-the-loop Testing:** The co-working platform facilitates hardware-in-the-loop testing, allowing specific HW components or subsystems to be tested within the context of the entire wireless system to validate hardware functionality.
- **Realistic Emulation:** The co-working platform aims to emulate the real-world behavior of wireless systems, allowing for thorough testing and validation.

HW/SW co-design has recently become a prominent strategy, focusing on integrating various SW simulations in a unified environment to enhance simulation speeds by minimizing communication overhead between SW and HW platforms. This approach improves conventional FPGA prototyping by offering greater flexibility, allowing test scenarios and extensive coverage tests to be modified. It excels in comprehensive system evaluations, especially for applications with diverse parameters and signaling types.

A notable benefit of SW/HW co-design is the accelerated deployment of HW (FPGA) prototyping, verification, and refinement of the system testbench. This streamlined approach is advantageous for researchers. Additionally, SW/HW co-design allows for flexible management of the trade-off between accuracy, speed, and run time. Tools like MATLAB provide a ”Hardware-in-the-loop” option for co-simulating HW designs within the Simulink Environment, while TCP/IP interfaces offer an effective alternative.

2.2.2 Limitations and Challenges of Wireless System Platforms

In exploring wireless system platforms, it becomes crucial to address the limitations and challenges that influence their development and utilization. In this section, we highlight the limitations of existing platforms and discuss the challenges of integrating AI within wireless systems.

1. **SW-Based Platform**
    - Advantages:
        - Rapid development and testing
        - High-degree of observability
        - Less effort of system setup
        - Flexible programming language and library support
    - Limitations:
        - Limited implementation for a realistic scenario
2. **HW-Based Platform**
    - Advantages:
        - Full system simulation
        - Realistic simulation for real-world wireless environments
    - Limitations:
        - High effort for the realization
        - Limited debugging facilities
        - Low flexibility
3. **SW/HW Co-Design Platform**
    - Advantages:
        - Flexibility and Rapid Prototyping
        - Simulation Capabilities

  - Real-Time Processing
  - Hardware-in-the-loop Testing
  - Realistic Emulation
- Limitations:
  - Complexity may lead to difficulties in seamless integration and communication between software and hardware components.
  - Platform scalability limitations may restrict its applicability to extensive network deployments or high-density scenarios.

2.2.3 SW/HW Wireless-AI Platform Requirements

Given the advantages and limitations of the existing platforms mentioned above, SW/HW co-design emerges as the optimal choice for developing a wireless-AI platform. A SW/HW Wireless-AI platform is essentially a SW/HW co-design platform designed to facilitate the integration of AI capabilities into developing wireless system applications. This platform provides comprehensive support for developing and testing various wireless-AI applications, leveraging both SW simulations and real-time HW testing for a robust and versatile solution.

To achieve an efficient and effective SW/HW wireless-AI platform, the following requirements need to be considered as a trade-off for the metrics of the framework [27] [28]:

- **Low Latency and Real-Time Processing:** To achieve rapid design exploration and support multiple test scenarios, critical requirements include effective interaction and data synchronization of SW/HW components.
- **Multi Software Development Environment:** When designing wireless-AI applications, researchers can choose they prefer SW to work with.
- **Scalability:** The platform must exhibit scalability features, accommodating various HW configurations and facilitating the integration of multiple devices.

- **Supporting Multiple Applications:** In the design of wireless-AI applications, diverse teams of designers may lack extensive expertise in each other's domains. Furthermore, simulations and verifications are frequently conducted using disparate frameworks. To address this practical challenge, offering an HW/SW co-design platform that various groups of researchers can utilize is highly advantageous.

Developing an effective SW/HW wireless-AI platform presents challenges, primarily related to the versatility required for various applications in SW simulation, HW real-time testing, and SW/HW connectivity. A critical challenge involves achieving a generic platform that accommodates diverse wireless AI applications, ensuring reusability and flexibility. Furthermore, scalability for multiple devices is crucial, ensuring researchers can experiment with various devices and choose their preferred SW for simulation.

In this thesis, we propose novel features for a unified SW/HW wireless-AI platform, including:

1. **Generic for various wireless-AI applications:** It can be reused for various applications.

2. **Flexible Software Selection (multi-software):** Researchers can choose their preferred SW, such as MATLAB or Python, or even use both.

3. **Dual Software (SW) and Hardware (HW) Modes:** Researchers can choose to use both SW and HW modes for simulating and implementing applications.

4. **Unified SW/HW Connectivity and Integration of Multiple Devices:** It can integrate and collaborate with multiple devices. This empowers researchers to seamlessly select the number of devices for their experiments, ensuring adaptability to diverse use cases and requirements.

2.3 IEEE 802.11 Physical Layer with Deep Learning

The IEEE 802.11 standard encompasses specifications for both medium access control (MAC) and physical layer (PHY) that facilitate the implementation of WLAN. The IEEE

Table 2.1: OFDM length and subcarrier number of WLAN for various standards.

	802.11a	**802.11n**	**802.11ac**	**802.11ax**
Number of subcarriers	52 (48)	56 (52)	56 (52)	242 (234)
OFDM length	4μs	4 μs	4 μs	16 μs

802.11 PHY serves as the intermediary between the wireless medium and the MAC layer, setting out the specifications for how radio waves are modulated for data transmission. The 802.11 standard encompasses four distinct PHY layer specifications: Infrared (IR), Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS), Complementary Code Keying (CCK), and Orthogonal Frequency-Division Multiplexing (OFDM). This thesis will focus on the more popular technique, OFDM.

2.3.1 Orthogonal Frequency-Division Multiplexing

OFDM operates by dividing a high-rate data stream into parallel, lower-rate streams transmitted simultaneously across multiple subcarriers. Each of these smaller streams is assigned to an individual subcarrier and modulated using techniques like Phase Shift Keying (e.g., Binary PSK or BPSK) or Quadrature Amplitude Modulation (e.g., QPSK, 16QAM, 64QAM). These subcarriers are carefully spaced to avoid interference. Utilizing lower-rate subcarriers reduces the impact of time dispersion from multipath delay spread. Additionally, introducing a guard time in each OFDM symbol almost eliminates intersymbol interference. Fig. 2.7 depicts the OFDM symbol with cyclic extension. The guard time cyclically extends the OFDM symbol, preventing intercarrier interference. This is achievable due to the orthogonality of frequencies (sub-carriers), where one sub-carrier's peak aligns with an adjacent sub-carrier's null. Table 2.1 shows OFDM length and subcarrier number of WLAN for various standards.

2.3.2 IEEE 802.11 Standards Family

The IEEE 802 series encompasses a range of IEEE standards governing both Local Area Networks (LANs) and Metropolitan Area Networks (MANs). Oversight of the IEEE 802

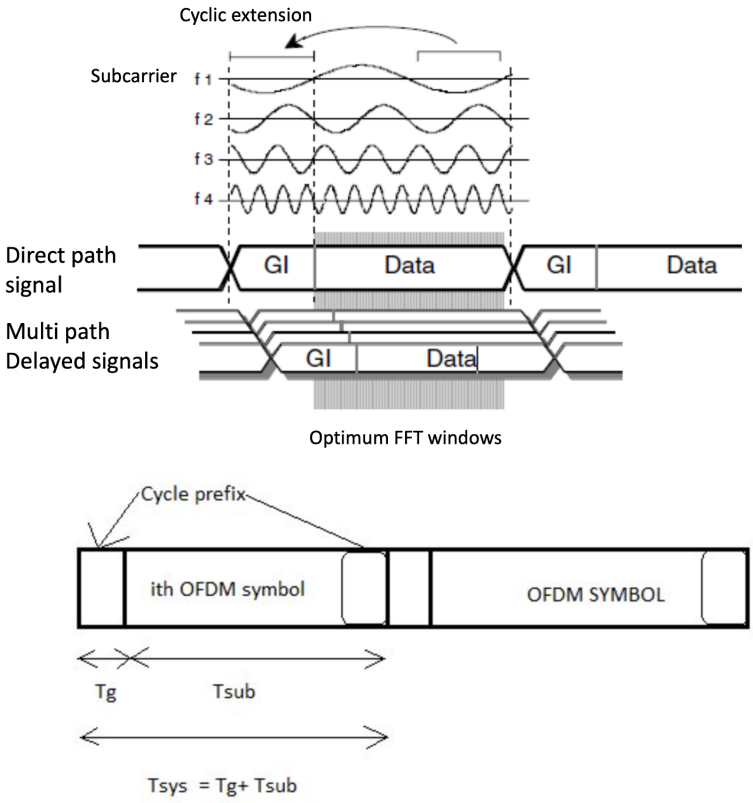


Figure 2.7: OFDM symbol with cyclic extension.

family of standards falls under the purview of the IEEE 802 LAN/MAN Standards Committee (LMSC), with specific Working Groups dedicated to individual domains. Among these, IEEE 802.11 stands out, comprising specifications governing the MAC and PHY for implementing WLAN communication. The 802.11 family encompasses a suite of modulation techniques for wireless communication, all adhering to a shared foundational protocol. These standards serve as the foundation for products in wireless networks bearing the Wi-Fi brand. The specific portion of the radio frequency spectrum allocated to 802.11 can vary from one country to another.

- IEEE 802.11a

The 802.11a standard, introduced in January 2003, experienced rapid consumer adoption even before official ratification, driven by a demand for higher speeds and manufacturing cost reductions. This standard transitioned most dual-band 802.11a/b products to dual-band/tri-mode by the summer of 2003, supporting 802.11a and b/g within a single adapter card or access point. Operating in the 2.4 GHz band like 802.11b, 802.11g utilizes the same OFDM-based transmission scheme as 802.11a, achieving a maximum physical layer bit rate of 54 Mbit/s.

- IEEE 802.11n

  The 802.11n amendment substantially improves WLAN performance, offering enhanced range, reliability, and throughput. These improvements result from advanced signal processing and modulation techniques in the PHY layer, utilizing multiple antennas and broader channels. High Throughput (HT) enhancements significantly boost data rates, reaching up to 600 Mbps, a notable advancement from the 54 Mbps of 802.11a/g. Operating in both the 2.4 GHz and 5 GHz bands, 802.11n introduces multiple-input multiple-output (MIMO) and 40 MHz channels at the PHY layer and frame aggregation at the MAC layer.

  A key feature of 802.11n is its capacity to receive and transmit data through multiple antennas simultaneously. The standard defines various antenna configurations, from $1x1$ to $4x4$. MIMO technology effectively utilizes multiple antennas, employing Spatial Division Multiplexing to transfer multiple independent data streams within a single spectral channel's bandwidth. Increasing the number of resolved spatial data streams through MIMO significantly enhances data throughput, with each stream requiring a distinct antenna at both ends.

- IEEE 802.11ac

  IEEE 802.11ac, or Very High Throughput (VHT), is an upcoming WLAN standard designed to deliver enhanced throughput in the 5 GHz band. Aimed at meeting the growing demand for higher throughput, 802.11ac ensures backward compatibility with 802.11n and 802.11a. The standard anticipates achieving a multi-station WLAN throughput of at least 1 Gbps and a single-link throughput of at least 500 Mbps. Building upon the concepts introduced by 802.11n, enhancements include

wider RF bandwidth (up to 160 MHz), support for more MIMO spatial streams (up to 8), multi-user MIMO, and high-density modulation (up to 256-QAM). Mandating support for 20, 40, and 80 MHz channels, 802.11ac also introduces optional features such as broader channel bandwidths, higher modulation, multi-user MIMO, short guard intervals, and the use of Space-Time Block Coding and Low-Density Parity Check.

- IEEE 802.11ax
  IEEE 802.11ax introduces a new PHY protocol with higher modulation and coding schemes, resulting in a nominal data rate increase of up to 9.6 Gbps. This marks a 37% improvement over 802.11ac, focusing on more efficient spectrum utilization rather than increasing MIMO spatial streams or widening the channel. A key feature is the adoption of OFDMA, commonly used in cellular networks but new to Wi-Fi, addressing challenges with frequency selective interference. OFDMA groups adjacent subcarriers into resource units, allowing optimal selection for each receiver. This leads to higher Signal-to-Interference-plus-Noise Ratio (SINR), Modulation and Coding Scheme (MCS), and overall higher throughput. OFDMA's efficiency is particularly notable for stations with limited data to transmit, offering six times higher throughput than legacy Distributed Coordination Functions (DCF). Unlike Long-Term Evolution (LTE), 802.11ax operates on top of the legacy DCF. It is coordinated by the access point (AP), providing flexibility for downlink (DL) transmission, DL Multiple User (MU) transmission, or allocating Resource Units (RUs) for uplink (UL) MU transmission after gaining channel access.

A WLAN packet consists of a preamble part and a data part. Over time, the WLAN preamble and PHY header have evolved, transitioning from a single symbol interference guard (SIG) field symbol to incorporating multiple SIG field symbols to align with the advancing 802.11 standard and its integration of new PLCP Protocol Data Unit (PPDU) formats. Fig. 2.8 visually depicts this progression, distinguishing SIG field symbols in a lighter shade from DATA symbols in a darker shade. Additionally, Table 2.2 presents the WLAN's packet preamble for various WLAN standards. Irrespective of the PPDU format, the SIG fields are preceded by the legacy-short training field (L-STF) and legacy-long

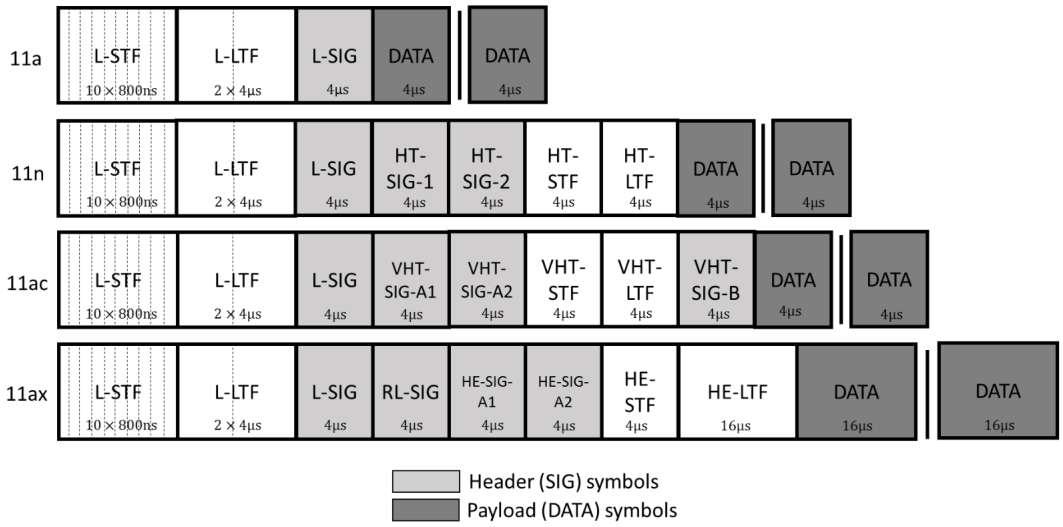


Figure 2.8: Evolution of 802.11 frame format [2]

training field (L-LTF) symbols, which are vital for initial frame synchronization and channel estimation. In contrast, the SIG fields contain crucial details, including duration and other pertinent information essential for the STA to decode the subsequent DATA symbols.

2.3.3 Deep Learning-based Approaches in the Physical Layer

Utilizing DL-based techniques in the PHY layer presents an opportunity to enhance performance compared to traditional algorithms. These DL algorithms serve as viable alternatives, offering a model-free approach. The effectiveness of DL-based methods is contingent on the transmitted signal's inherent characteristics rather than on mathematical models as in conventional PHY layers. Recent research has embraced this approach, including signal detection and modulation recognition tasks across various physical modules.

There are two approaches to applying DL in the PHY layer of communication systems. First, the system can be replaced entirely with a neural network to achieve DL-based end-to-end communication. Second, specific blocks of the existing block-structured communication can be enhanced or augmented using DL algorithms. A data-driven DL communication system treats the entire system or specific blocks as a black box that can

Table 2.2: WLAN's preamble part for various WLAN standards.

Packet format		**L-STF**	**L-LTF**	**L-SIG**	**SIG**	**STF**	**LTF**	**SIG**	**Sum**	**Sampling at freq =20MHz**
Legacy/ 802.11a	Symbol	2	2	1					5	
	OFDM length	4µs	4µs	4µs					20µs	400
HT/ 802.11n	Symbol	2	2	1	2	1	1		9	
	OFDM length	4µs	4µs	4µs	4µs	4µs	4µs	4µs	36µs	720
VHT/ 802.11ac	Symbol	2	2	1	2	1	1		9	
	OFDM length	4µs	4µs	4µs	4µs	4µs	4µs	4µs	40µs	800
HE/ 802.11ax	Symbol	2	2	1	3	1	1		10	
	OFDM length	4µs	4µs	4µs	4µs	4µs	16µs		52µs	1040

be tuned using labeled data. This approach demands substantial training datasets and sufficient computing resources for accurate training. Conversely, model-driven techniques leverage prior knowledge of the communication system to reduce the number of trainable parameters, consequently shortening the training time. However, data-driven techniques are more robust under varying circumstances due to their reduced assumptions. In contrast, model-driven methods may only work if the initial assumptions align with real-world scenarios.

Figure 2.9 illustrates these distinct categories of DL applications for communication systems, as suggested in [15]. The end-to-end DL-based communication system falls under the data-driven category, as it involves replacing the entire system with a DNN trained solely on labeled data without any presumptions. On the other hand, DL-based block-structured communication systems can be either data-driven or model-driven. A specific block is substituted with a neural network in the data-driven block-structured approach. In contrast, with the model-driven approach, the network is simplified by utilizing conventional mathematical models or prior knowledge, resulting in a reduction of trainable

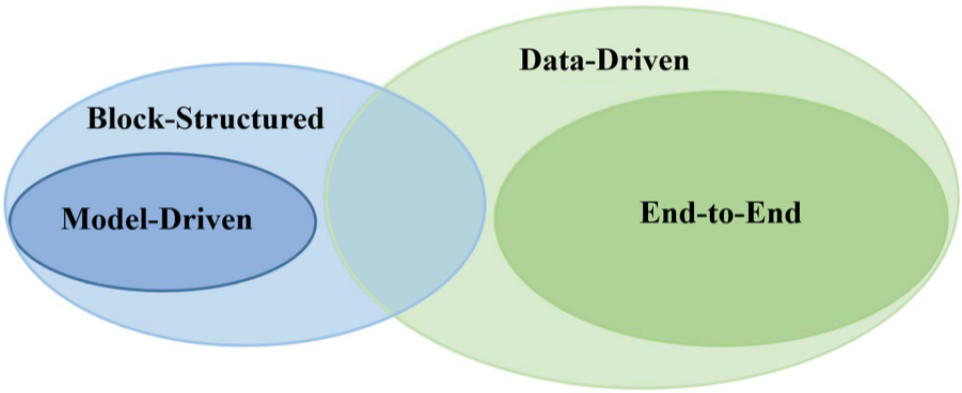


Figure 2.9: Various DL classifications for communication systems [3].

parameters. The block-structured model-driven DL method is particularly appealing for PHY layer communications, as mathematical models are typically available. This allows for modeling a rough and general solution for one or several cascade functions, enhancing system performance.

2.3.4 Application of Deep Learning in Physical Layer: Signal Detection and Modulation Classification

The capability of a wireless communication system to accurately perceive and adapt to its operating environment hinges on its proficiency in signal detection and classification. This encompasses tasks such as identifying and coexisting with other spectrum users. Conventionally, signal detectors are required for each waveform, designed based on its analytical properties. However, this approach can lead to systems that are challenging to develop and deploy effectively in practical wireless scenarios due to their excessive specificity, complexity, or suboptimal performance in real-world conditions [19].

Signal detection is critical in communication systems, extracting original information from noise-influenced signals. A crucial aspect involves signal detection, wherein the receiver identifies the transmitted information. The likelihood of successful detection is directly linked to the signal-to-noise ratio (SNR) at the receiver.

Conventional signal detection and classification techniques can be divided into two

categories [19]:

1. General Methods: These approaches do not rely on prior knowledge about the signal types. They can detect various signal types, but their performance in maintaining a constant false alarm rate (CFAR) is comparatively less efficient.

2. Specialized Methods: These methods are designed to provide susceptible detectors for specific types of signals. They involve developing detection and classification techniques based on specific features unique to the signal of interest. However, these methods often lack scalability, as a new type of classifier is necessary for each distinct waveform.

While optimal detectors, like the maximum likelihood detector, can be derived using statistical models with known distributions, they often come with a high computational burden that makes them impractical for real-world applications. As a result, suboptimal detectors, such as various linear detectors, are commonly employed to strike a balance between performance and complexity. In systems with multiple carriers and antennas, the challenge of designing a high-performance signal detector intensifies due to the expanded dimensions of the signal space. Suboptimal detectors based on techniques like approximate message passing and semidefinite relaxation have been explored in this context. However, these methods are highly contingent on the accuracy of channel estimations, and their computational demands can be burdensome. To address these limitations, recently, there has been a proposal to employ DL-based signal detectors.

Modulation recognition, a crucial technology in software-defined radio, empowers cognitive users to identify diverse transmission technologies for dynamic spectrum sharing without necessitating system-specific information. Conventional approaches to modulation recognition fall into two categories: likelihood-based decision-theoretic methods and feature-based statistical pattern recognition methods. Likelihood-based methods rely on stringent statistical assumptions and prior knowledge, often leading to high computational complexity. Feature-based methods employ learning-based classifiers like artificial neural networks, random forests, multilayer perceptrons, and support vector machines, demanding extensive manual effort for feature extraction. Moreover, these conventional techniques exhibit subpar performance in scenarios with low SNR and high interference [20]. To address

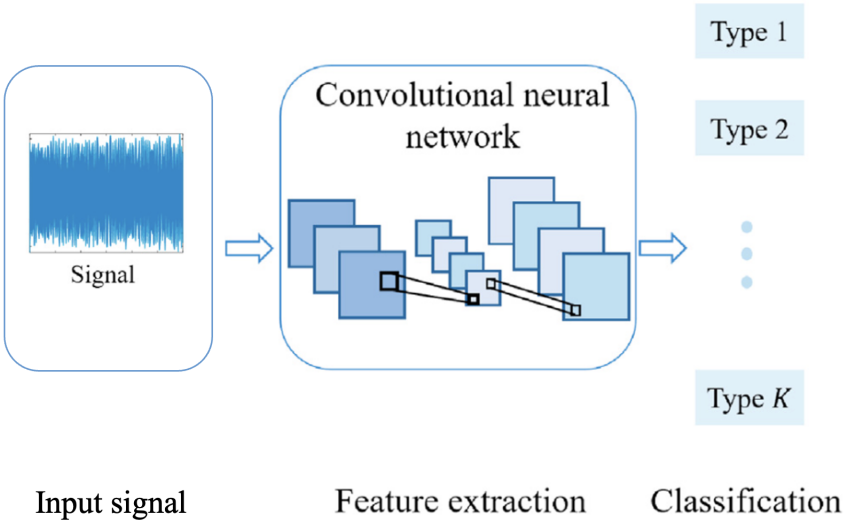


Figure 2.10: Signal classification using CNN

these limitations, various DL-based techniques have been introduced [21–23]. Fig. 2.10 shows the signal classification application using CNN. Compared to conventional methods, DL-based approaches bypass the need for manual feature extraction and deliver significantly enhanced accuracy and robust modulation recognition.

2.4 Summary

This chapter presents an overview of ML-assisted wireless communication systems and introduces a dedicated platform to facilitate seamless ML integration within these systems. We have outlined the advantages and limitations of current technology, underscoring the potential advantages of incorporating ML in wireless communication and emphasizing its applications. Additionally, we brought attention to an existing platform for the development of wireless systems, noting its limitations in terms of flexibility. In light of the limitations identified in the current platform, we thoroughly examined the prerequisites essential for a robust SW/HW co-design platform tailored for wireless design. Consequently, to meet these requisites, we introduce the unified SW/HW wireless-AI platform, which builds upon the existing SW/HW platform, enhancing its capabilities.

Chapter 3

A Unified Software and Hardware Platform for Machine Learning Aided Wireless Systems

3.1 Introduction

Artificial intelligence (AI), specifically machine learning (ML), has brought significant advancements to wireless communications in recent years. The integration of ML in wireless systems allows for the development of intelligent, self-adaptive, and self-reconfigurable systems [29]. ML-based approaches have the potential to replace conventional methods in developing wireless applications, protocols, and devices [30]. Consequently, the incorporation of ML in wireless systems is becoming increasingly widespread. ML-based wireless systems are expected to improve the performance of wireless networks by enabling new features such as self-organizing networks, proactive interference management, and dynamic resource allocation. As wireless networks become more complex, ML-based techniques are essential for designing, optimizing, and managing wireless systems. This leads to increasing demand for platforms that support the development, testing, and deployment of ML-based wireless systems.

Developing ML applications in a wireless system requires a platform including software (SW) and hardware (HW) components. It enables researchers to build and evaluate the

performance of their intelligent algorithms. Popular AI libraries such as TensorFlow, Keras, and PyTorch, integrated with Python, are often used to build and test intelligent algorithms quickly. Additionally, MATLAB Simulink toolboxes, such as WLAN and signal processing toolboxes, are utilized by some researchers to develop wireless system algorithms. During the testing phase, validating the algorithm's performance in a realistic scenario as a real-time application is crucial. This often involves deploying the application on a real-time HW testbed for development and implementation in a laboratory setting. Hence, utilizing multi-SW and HW platforms for development and testing is a widespread challenge in the ML-based wireless system development process.

To overcome this challenge, a unified SW and HW platform has become increasingly popular. It enables the flexible development of both SW and HW components on a single platform rather than using separate platforms for each. The versatility of a unified platform in implementing diverse research scenarios simplifies the process for researchers to create and evaluate new ML models.

Several references have described platforms for wireless systems featuring AI integration, [31], [32], and [33]. However, these platforms have limitations in terms of flexibility and applicability. For example, the platforms described in [31] and [32] are limited to use in the MATLAB environment. As a result, these platforms may not be easily adaptable for use in high-level SW development for a wide range of applications. The platform described in [33] is a virtualized testbed for a radio access network, which offers high flexibility for research scenarios but is not suitable for real-time or actual implementation.

This thesis introduces a Unified SW and HW Wireless-AI Platform (USHWAP) to facilitate machine learning within wireless systems. The proposed platform enables the design and verification of various ML applications, supporting both SW simulation and HW implementation, and can be applied to a wide range of off-the-shelf devices. The platform integrates SW/HW co-design, which unifies high-level SW such as MATLAB and Python, along with FPGA and SDR. USHWAP includes a unified connectivity for multiple devices, facilitating efficient data transfer between the host PC and the FPGA. This feature makes it highly suitable for device and edge computing applications.

The USHWAP platform has an advanced trigger mode function that allows precise control and synchronization of data transfer timing. This feature is beneficial for coordinating

multiple devices, especially in edge computing scenarios. For example, it ensures accurate coordination and synchronization among multiple devices acquiring Channel State Information (CSI) through Wi-Fi sensing or implementing full-duplex (cooperative Multiple-Input Multiple-Output (MIMO) systems). This capability improves the performance and reliability of edge computing applications that rely on multi-device cooperation.

3.2 Scope of the Platform

The proposed USHWAP is a flexible and versatile co-design platform. USHWAP serves as an integration of multi-SW and HW for the creation and advancement of diverse wireless AI applications. This platform integrates multi-SW simulations, including MATLAB and Python, with cutting-edge HW platforms such as FPGA and SDR, as depicted in Fig. 3.1. As a unified SW/HW co-design platform, USHWAP unfolds its utility through three distinct usage scenarios.

1. Researchers have the flexibility to select their preferred SW, either MATLAB or Python or even both, for the simulation of wireless AI applications (see Fig. 3.1, point (1)). MATLAB or Python is utilized to create AI models and perform simulations. Furthermore, MATLAB supports generating datasets using its Simulink and Toolbox, while Python offers extensive AI libraries for training datasets. A bridging software has been developed to convert a trained model from Python to MATLAB to facilitate the interface between Python and MATLAB for transferring a DL model.

2. Researchers can use the SW and HW modes to simulate and implement their wireless AI applications. AI models are developed and simulated using either MATLAB or Python. Additionally, MATLAB can be employed to receive the actual signal from the FPGA for dataset collection. In the SW mode, the transmitter (Tx) and receiver (Rx) are implemented in the SW part, allowing researchers to customize them easily according to their specific requirements. For HW mode, researchers deploy their simulated AI model on the HW platform (FPGA and SDR) to conduct real-time evaluation (refer to Fig. 3.1, point (2)). The Tx and Rx are implemented on the FPGA in the HW mode using hardware description language (HDL).

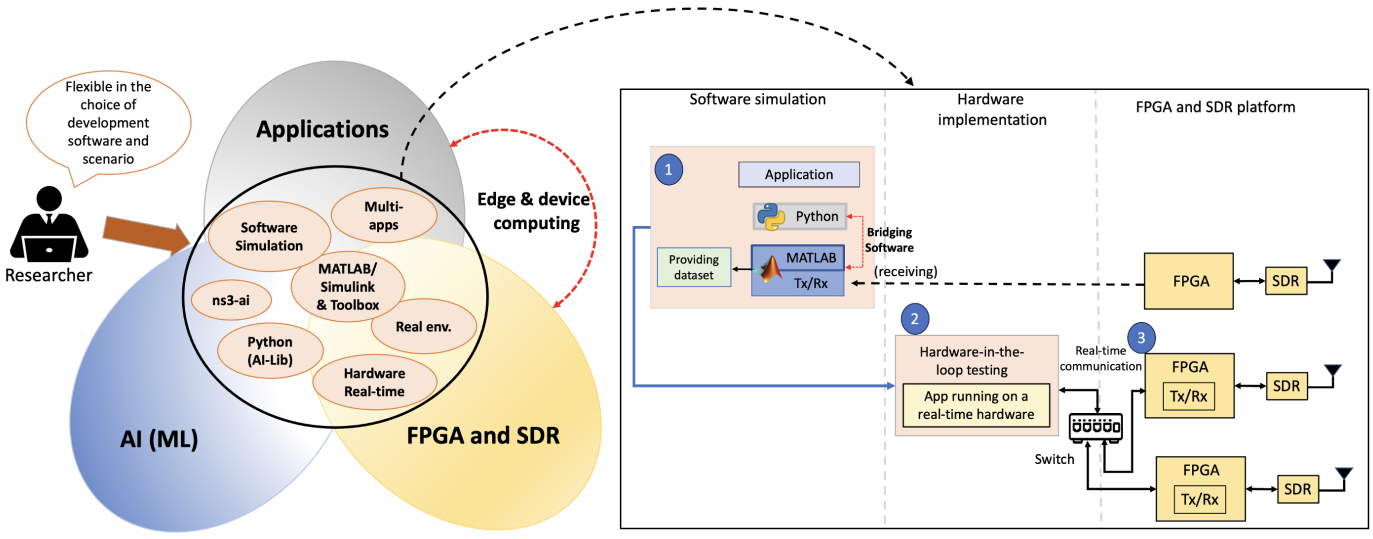


Figure 3.1: Proposed USHWAP system.

3. The platform facilitates unified SW/HW connectivity and integration of multiple devices (refer to Fig. 3.1, point (3)). It allows researchers to easily choose the number of devices for their experiments, such as edge computing applications. The unified connectivity and flexibility of designs make USHWAP highly suitable for various research scenarios.

Several platforms have been developed to integrate AI into the network. In [31], [32], and [34], the authors propose an SDR testbed platform for AI signal recognition and body movement using CSI. The platform demonstrates real-world performance but lacks software development flexibility across platforms; [31], and [32] focus on the development of MATLAB environments, while [34] proposes a system implemented using only Python. In [33] presents a testbed platform that integrates AI in a radio access network using virtualized network functions. It utilizes software-based RF emulation to test various network scenarios, providing flexibility for research. However, it is limited to software simulations and cannot be applied to actual scenarios.

Other works have proposed using testbed platforms featuring AI in cloud environments, as discussed in [35] and [27]. In [35], the authors developed a testbed to evaluate fog radio access networks and develop intelligent algorithms in a lab setting, using the open-air

interface and universal software radio peripheral (USRP). Meanwhile, in [27], the authors discuss a testbed that integrates AI into core-network, optical transport, and 5G radio access networks, using actual data for experimental and AI optimization. However, these platforms can be expensive to implement on a lab scale, which may limit accessibility for many researchers. In [36], the authors utilized the USRP platform for implementing an OFDM communication system. However, it does not consider the SW/HW connectivity aspect in its work.

The proposed platform addresses the above-mentioned limitation by offering a comprehensive solution for ML-aided wireless systems, including developing various wireless AI applications, software simulation, and real-time hardware implementations. It provides a flexible and unified SW/HW development and connectivity approach. This enhances researchers' flexibility in their research and development process. In contrast, the reference platforms lack this comprehensive approach and are limited in their scope, designed only for a specific type of high-level software development, restricting researchers' ability to choose their preferred method. The proposed platform offers unified connectivity in both SW/HW. This sets it apart from USRP, which requires using specific MATLAB toolboxes like HDL Coder for HW implementation [36].

3.3 Proposed Unified SW/HW Wireless-AI Platform

3.3.1 General Architecture of USHWAP

The USHWAP architecture employs a modular design for its SW and HW components, which enables easy integration of new features. Fig. 3.2 depicts the architecture of the USHWAP system, showcasing the distinct modules of SW and HW functions along with their components. A modular programmability approach that utilizes the strengths of both MATLAB and Python for simulation is adopted to enhance flexibility. MATLAB is recognized for its Simulink and Toolboxes, including signal processing and wireless toolboxes, which enable the development of algorithms and customization of wireless system parameters. Meanwhile, Python offers a wide range of open AI libraries, such as TensorFlow,

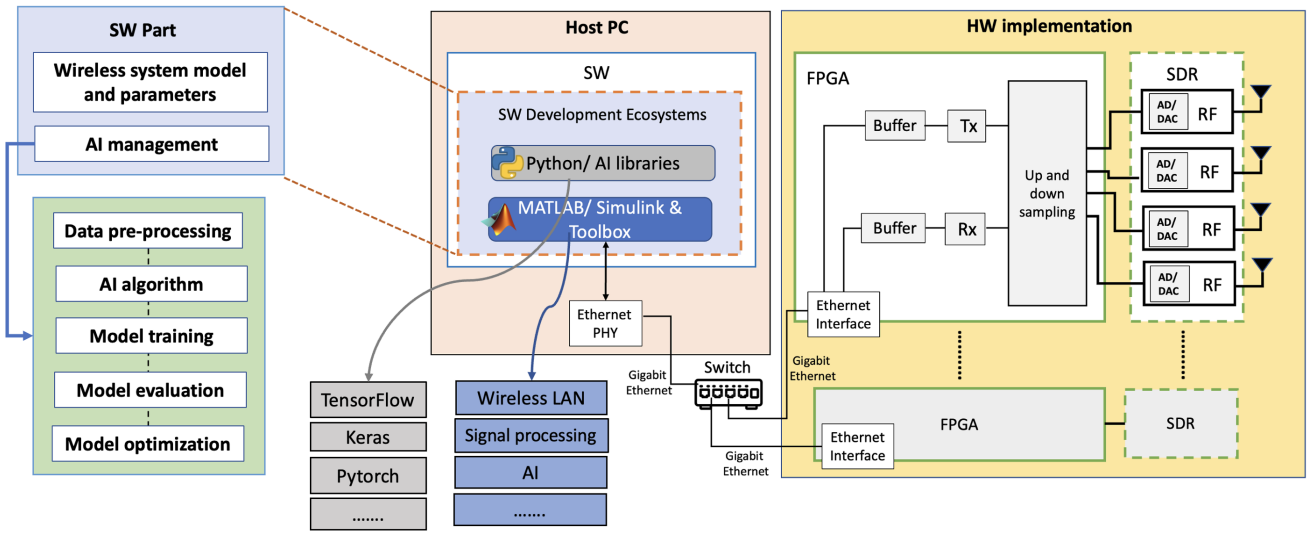


Figure 3.2: USHWAP system architecture.

Keras, and PyTorch, allowing the rapid creation and testing of intelligent algorithms. During application development, MATLAB and Python can operate independently or collaboratively (co-working). In co-working scenarios, MATLAB provides the dataset, which Python then employs to construct an AI model and conduct dataset training. Using multi-SW capabilities, researchers can leverage their individual strengths in their respective work. The modular programmability characteristic empowers the focused development of wireless AI applications, ensuring adaptability for diverse purposes.

The platform employs a task partitioning method to optimize the platform's design by dividing the functions between SW and HW, as shown in Fig. 3.2. The transmitter (Tx) and receiver (Rx) blocks are located within the HW component of the platform, which can be implemented across multiple devices (FPGA and SDR). On the other hand, functions such as the AI application module (simulation part), system parameters, and performance evaluation are implemented within the SW component. This approach enables us to allocate resources and effectively create an efficient design platform.

A wireless communication system comprises three signal communication blocks: the Tx, channel model, and Rx. The Tx and Rx blocks are placed in the HW part as an emulated platform, whereas several components are implemented in the SW part, i.e., the application

module, dataset generator, system parameters, and performance evaluation. The purpose of flexibility should be under consideration in SW/HW design.

The USHWAP operates as a master-slave system, where the host PC serves as the master, overseeing the entire process, and the FPGA functions as the slave. In this configuration, the host PC assumes control by transferring the generated bit file to the target FPGA. It then orchestrates the FPGA's operations through an application programming interface (API) program, governing data exchanges between the host PC and the FPGA. The API facilitates data transactions, ensuring the efficient movement of data between the MATLAB processes and the FPGA components. The bus interface module facilitates this data interchange, which facilitates data transactions via an Ethernet cable connecting to the FPGA. The SDR is employed to manage the system's radio frequency (RF) front-end.

From the HW standpoint, the system incorporates a flexible interface responsible for managing two transfer modes: single shot and cyclic shot. Moreover, it encompasses two transmission modes, namely the normal and trigger modes, enhancing the system's adaptability and functionality.

3.3.2 Hybrid of Software Simulation and Hardware Real-time Implementation

SW simulation offers researchers a user-friendly environment that delivers fast results. However, it may have limitations regarding real-time capabilities and performance compared to HW implementation. On the other hand, HW implementation is achieved by implementing and validating the system using FPGAs and SDRs, ensuring real-time performance. Nevertheless, it is crucial to acknowledge that developing HW-based platform can be time-consuming and incur higher costs.

The USHWAP provides an SW/HW co-design platform that empowers researchers to develop wireless AI applications through SW simulation and HW implementation. The platform supports MATLAB and Python, utilizing their Simulink and AI library capabilities for simulation. Additionally, USHWAP enhances the simulation by integrating AI libraries for network optimization using ns3-ai and Python. Researchers can deploy their

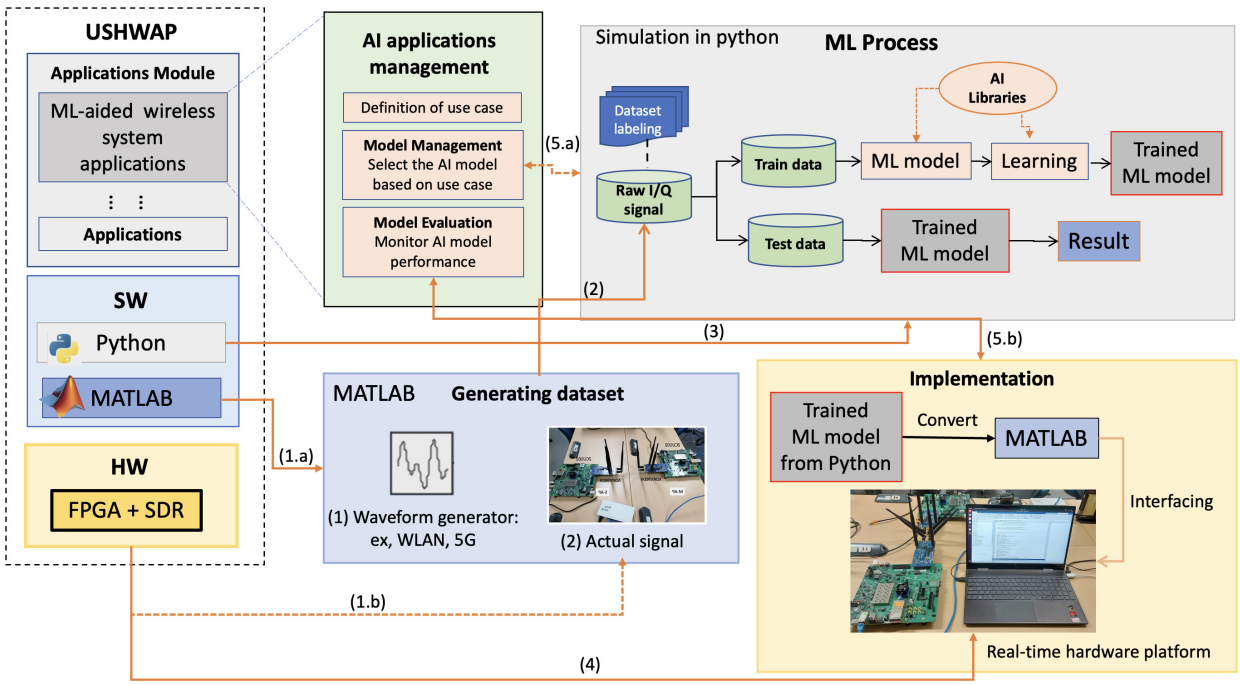


Figure 3.3: AI-enabling architecture on USHWAP.

applications on the HW platform to evaluate system performance under real-world conditions, accounting for factors like channel models and analog impairments. The USHWAP's HW capability can be extended to support edge computing applications by utilizing multiple FPGAs and SDRs. It offers unified connectivity for SW and HW components without requiring any specific MATLAB toolbox. The platform utilizes the trigger mode function to control and synchronize data transfer timing, especially for coordinating multiple devices. The SW/HW co-design platform approach simplifies and streamlines the development and verification of new AI algorithms, offering a more efficient and cost-effective solution than using separate platforms.

3.3.3 AI-Powered USHWAP

The USHWAP integrates the AI algorithm as an application module, enabling a focused evaluation of its input-output process and real-time performance. For online-training and online-testing algorithms, such as reinforcement learning algorithms, real-time performance

is achieved by directly obtaining real-time data from the FPGA and analyzing it on the host PC. In contrast, AI algorithms like deep learning, such as convolutional neural networks (CNNs), which require extensive training time, can be trained offline and implemented on the USHWAP. The USHWAP leverages the Nvidia GPU GeForce RTX 3090 for training purposes on the workstation.

The AI workflow within the USHWAP encompasses multiple stages, including dataset creation, AI algorithm development, model training, and real-time implementation. Fig. 3.3 demonstrates an example workflow, showcasing the second usage scenario for enabling AI on the USHWAP. The diagram depicts the sequential steps in this process, highlighting the progression from dataset creation to model training and, ultimately, real-time implementation. MATLAB is utilized for dataset generation through offline processes, employing its Simulink and Toolbox functionalities, particularly the signal generator (1a). Additionally, datasets can be generated through online processes by capturing real signals from FPGA and SDR (1b). Subsequently, the raw data is utilized as the dataset and labeled accordingly (2). In Python (3), AI models are created and trained using the dataset, leveraging AI libraries such as TensorFlow, Keras, and PyTorch. Upon completion of training and achieving high accuracy through convergence, the models are stored. For real-time implementation (4), the AI model is tested on the HW platform (FPGA and SDR) to conduct real environment evaluations. A developed bridge function facilitates the connection between Python and MATLAB, enabling the transfer of trained AI model parameters from Python to MATLAB. Finally, the AI model is evaluated (5a - 5b).

3.3.4 USHWAP SDR-based Transceiver

In the USHWAP architecture, the host PC controls the FPGA. The host PC is responsible for uploading files and collecting data from the FPGA through an application API program. The API allows for data transfer between MATLAB and the FPGA, and an Ethernet cable connects the FPGA to the host PC. The SDR handles the radio frequency (RF) front end. This architecture enables real-time control and data collection.

The FPGA and SDR serve as versatile wireless communication platforms, capable of

being configured across various frequencies, modulation methods, protocols, and standards. Within the system, signals generated on a host PC are converted into a digital format, subsequently processed through a digital-to-analog converter, and then propelled for transmission. In preparation for transmission, the SDR adeptly gathers the IQ samples, while a subsequent frequency conversion transforms the baseband signal into a broadcast waveform suitable for propagation through the air. At the Rx, the incoming signals undergo down-conversion, followed by an application of analog-to-digital conversion. The API facilitates the seamless transfer of received data from the FPGA to the host PC via the Ethernet cable. An interrupt signal initiates reading the HW's direct memory access (DMA) to ascertain signal reception status. The API function efficiently gathers the received signal, subsequently forwarding it to the host PC for analysis.

In platform design, it incorporates two distinct modes for data transmission:

1. Normal mode: This mode involves the host PC initiating the transmission of data packets, which are then relayed to the FPGA and subsequently transmitted.

2. Trigger mode: The trigger mode represents an advanced feature of USHWAP, designed specifically to facilitate synchronization processes required for data transmission for coordinating multiple devices. This functionality is pivotal in assisting researchers in developing cutting-edge WLAN algorithms within the realm of the 802.11b standard. It is instrumental in achieving precise time control over data transmission, catering to communication scenarios demanding synchronization processes like implementing full-duplex (cooperative Multiple-Input Multiple-Output (MIMO) systems).

In the trigger mode process, the packet data from the host PC is stored within the FPGA buffer. Subsequent transmission of the buffered data occurs only upon receiving a trigger signal. This trigger function addresses the emerging coordination challenge associated with multi-access point (multi-AP) scenarios.

Alongside the diverse data transmission modes, the USHWAP encompasses two distinct burst data streams:

1. Single-shot: This mode involves the transmission or reception of data occurring once.

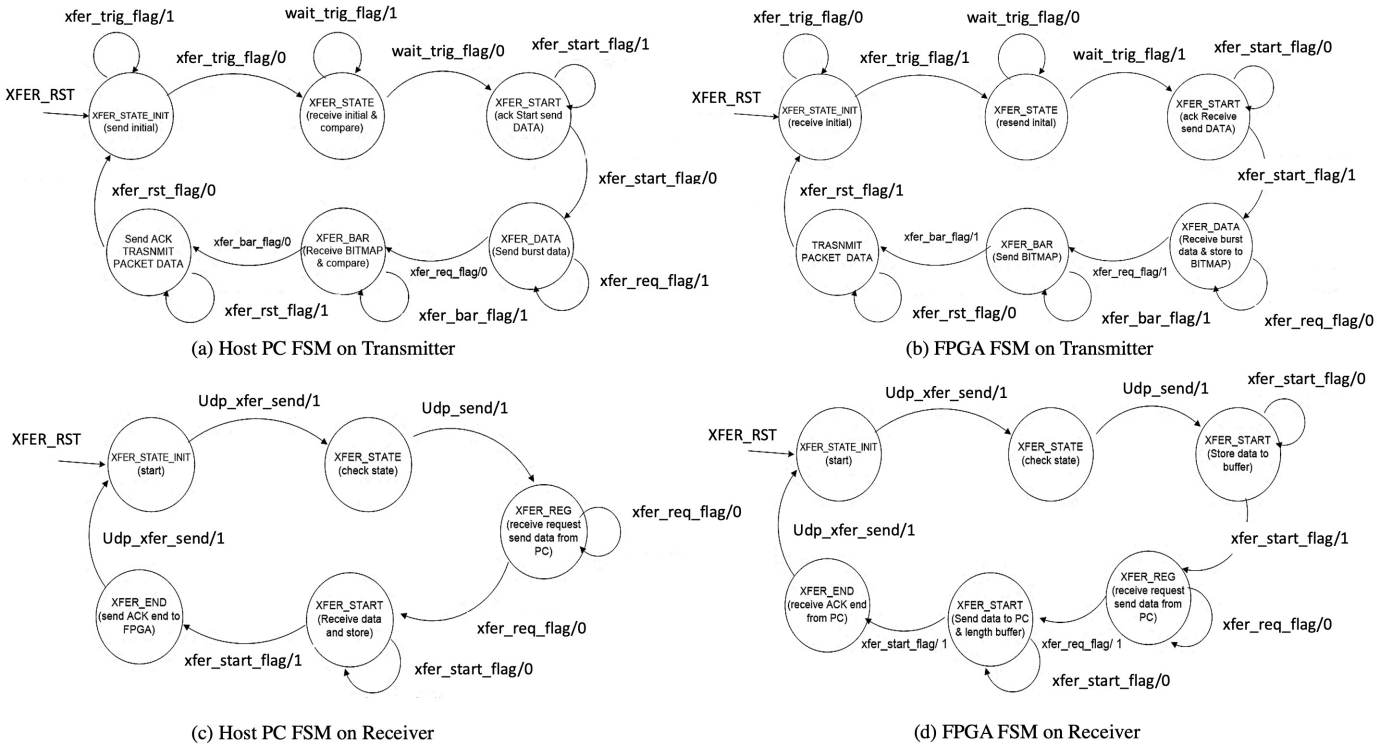


Figure 3.4: FSM design on transmitter and receiver

2. Cyclic-shot: In this mode, data transmission or reception is maintained in a continuous cycle.

Importantly, both normal and trigger mode transmission capabilities seamlessly integrate with these burst data stream modes, contributing to the platform's inherent flexibility and adaptability.

The USHWAP system employs a connectionless protocol with Internet Protocol and User Datagram Protocol (UDP) for efficient and reliable communication between the host PC and the FPGA. An enhanced HW interrupt-based interface method using DMA is proposed for high-speed wireless data transmission. This interface utilizes a single memory buffer for efficient data transfer, reducing the number of interrupts and memory copy operations to control data flow.

The finite state machine (FSM) is presented to define the transmitter and receiver transaction process. The FSM transmitter and receiver process is viewed from a host PC as a

master controller and FPGA as a slave. Fig. 3.4 shows the design of FSM. The interface's backend function *rx_buffer_status* is activated whenever a valid packet is detected and received, updating the buffer status and providing timestamps for the actual packet received. This process increases the overall speed of the data transfer and improves system efficiency.

The data from the host PC to the target HW is temporarily stored in a buffer until all data is ready for transmission. The entire data set is then encapsulated as a BITMAP file with a specified data length. Subsequently, the BITMAP is transmitted to the hardware. The backend function *rx_buffer_status* triggers the DMA, allowing the peripheral to access memory directly without relying on the processor. This enables the processor to perform other tasks concurrently while the DMA accesses memory, resulting in faster data transfers. The sequence of this process is outlined in the FSM diagram below:

1. Data Transmitting Process in host PC
    - Send the UDP packet with the specific data length as the initial signal to the slave.
    - Receive the initial signal back from the slave and compare the data length.
    - Set *XFER_STATE*=1 if slave ready to transmit data.
    - *XFER_START*=1 and sends a signal to the slave as the initial preparation to send data.
    - *XFER_DATA*=1 and send the burst data to the slave.
    - Receive the BITMAP from the slave and compare the BITMAP.
    - Send *ACK* signal to TRANSMIT DATA.
2. Data Transmitting Process in FPGA systems
    - Receive the initial signal and resend the initial to the host PC
    - Receive signal *XFER_STATE*
    - Receive signal *XFER_START* and prepare to receive the data from the master
    - Receive the burst data and store the data in the BITMAP
    - Set *XFER_BAR*=1 and send the BITMAP to the master

- Receive the *ACK* signal and TRANSMIT DATA

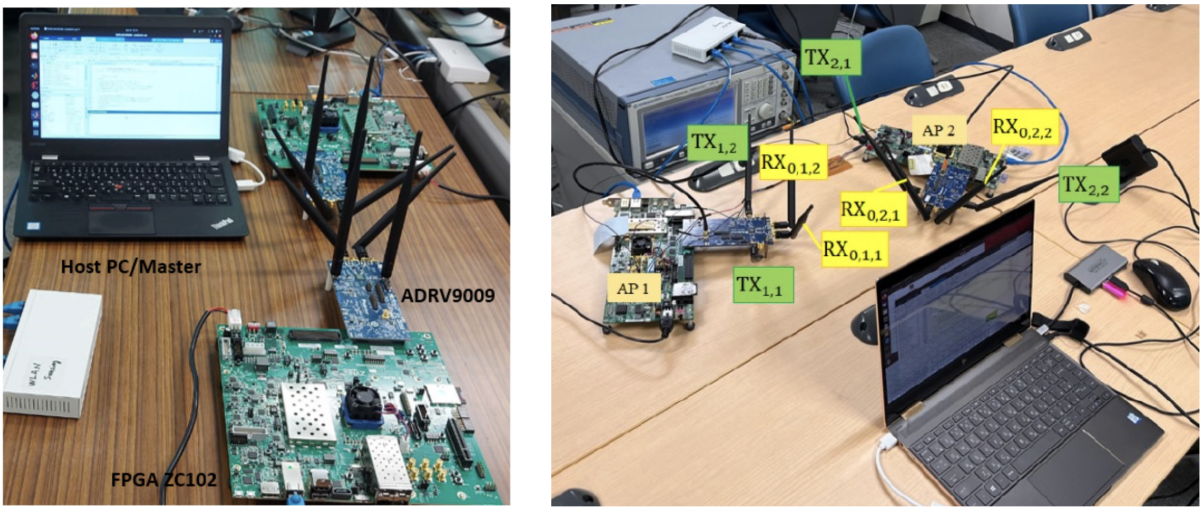
3. Data Receiving Process in a PC host

   - Send the UDP signal as the initial to the slave
   - Set *XFER_REG*=1 and send to the slave as the signal request to send the data
   - Receive the DATA and store it in the buffer, then compare the length of the DATA
   - Set *XFER_END*=1 and send it to the slave

4. Data Receiving Process in FPGA systems

   - Receive the DATA TRANSMIT and store it in the buffer
   - Receive the initial signal
   - Receive the signal *XFER_REG*
   - If DATA ready, set *XFER_START*=1, send burst data and the length of data to the master
   - Receive signal *XFER_END*

Fig. 3.5 depicts the HW configuration for implementing the USHWAP system. The system is implemented on a Ubuntu host PC and powered by an AMD Ryzen 7 series 4000 processor. Additionally, a Macbook Pro M1 is also utilized in the implementation. Data transmission is facilitated using the Xilinx FPGA ZC102 and ZC706 in conjunction with the SDR ADRV9009 and ADRV9371 RF front-end modules. The multigigabit-per-second serial transceiver FPGA Mezzanine Card (FMC) bridges the RF front-end, mixed-signal baseband, and frequency synthesizers. This transceiver supports two transmit and two receive channels, operating across a frequency range of 75 MHz to 6 GHz. The transmit synthesis bandwidth is set at 450 MHz, while the receive bandwidth is configured at 200 MHz. An ethernet cable establishes connectivity between the host PC and the FPGA. Furthermore, a workstation setup for training uses an Nvidia GeForce RTX 3090 GPU.



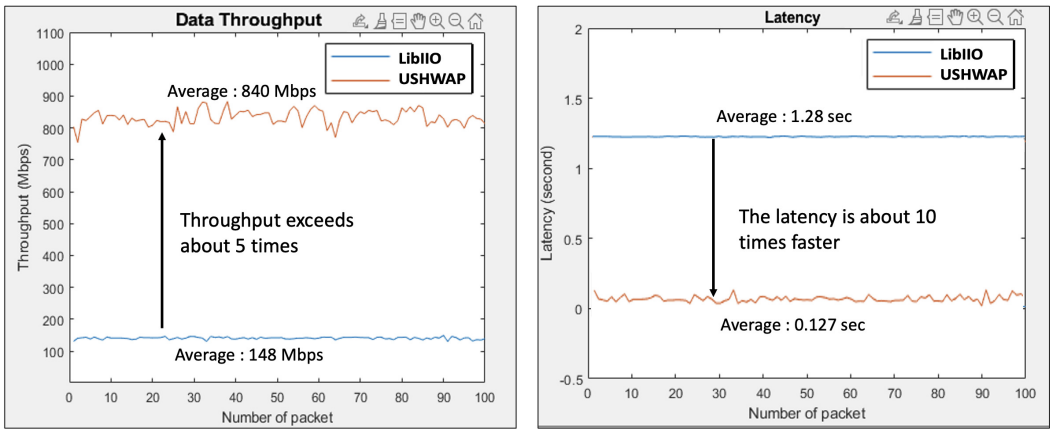
(a). USHWAP implementation using FPGA (b). WLAN experiments on USHWAP

Figure 3.5: USHWAP implementation and experimental testing.

3.3.5 Multi-purpose and Flexible Development Platform

The USHWAP is a versatile solution for developing applications, offering a comprehensive multi-SW environment with support for MATLAB and Python for simulation and FPGA for real-time implementation. This allows for efficient time management by using system simulation for development and real-time implementation without the need for extensive HW design, making it easy for researchers to develop and implement their applications quickly.

The USHWAP's HW flexibility enables multiple FPGA and SDRs, and its modular programmability design allows for easy updating or swapping of applications. This makes the USHWAP highly adaptable and suitable for various development needs. The platform's versatility and adaptability are demonstrated through three use cases: wireless signal classification, real-time WLAN sensing, and rate adaptation, showcasing its ability to scale and customize to fit specific development needs. The USHWAP uses a range of open-source and proprietary tools for AI development integrated into Python libraries and the benefit of the MATLAB toolbox and Simulink, which are not open-source but offer valuable capabilities for wireless system development. Overall, the platform provides a versatile and robust environment for AI development in wireless systems.



(a) (b)

Figure 3.6: Comparison performance evaluation: (a) Throughput, (b) Latency.

3.3.6 Evaluating Performance

The performance of USHWAP is evaluated by measuring its data throughput and latency, focusing on determining its SW's effectiveness. The evaluation system is configured with two spatial streams, a sampling rate of 122.88 MSamples/s, and a channel bandwidth of 20 MHz. The experiments are conducted indoors, using two pairs of FPGA and SDR as the transmitter and receiver at a distance of 1 meter. Packet data is generated using MATLAB. Each transmitted packet contains 2^{19} IQ samples, and the evaluation consists of 100 frames of packets.

The performance of USHWAP is compared to a SW-based open interface library (LibIIO) v0.21 as a reference. LibIIO is a SW interface for industrial I/O devices based on the Linux interface for generic interfacings, such as RF transceivers [37].

Since the previously mentioned platforms [31]-[35] did not provide reported data throughput performance in the literature, we used LibIIO as a comparison point. We conducted our assessments using the same HW device/platform and system parameters to ensure a fair evaluation. This approach allowed us to compare the performance of the data transmission speed for interfacing SW in USHWAP to a known reference, LibIIO, and determine the effectiveness of USHWAP in comparison. The results demonstrate that USHWAP achieves

a five times higher throughput than LibIIO, reaching a maximum of 800 Mbps. This improvement is accomplished by utilizing the UDP protocol, which enables efficient communication between the host PC and the FPGA. Furthermore, the latency of USHWAP is found to be ten times faster compared to that of LibIIO. The platform's effectiveness is further demonstrated by the fact that the latency of USHWAP is ten times faster than that of LibIIO. The results are presented in Fig. 3.6, which compares the throughput and latency of the proposed platform and LibIIO.

3.3.7 Comparison of Platform

This unified SW/HW wireless AI platform, USHWAP, aims to provide the SW/HW coworking platform for the flexible development of wireless AI applications. The USHWAP offers a comprehensive solution as it unifies SW/HW for developing various wireless AI applications, enabling multi-SW simulation (MATLAB and Python) and real-time HW implementation (multiple devices). This sets it apart from other existing platforms in the literature [31]-[35], which are typically designed for specific types of SW, HW, or use cases.

One of the critical advantages of USHWAP over platforms like USRP is its unified connectivity. While USRP offers unified connectivity for SW cases, it requires specific connections using MATLAB toolbox related to HDL coder for HW cases[*1], leading to increased costs[*2]. In contrast, USHWAP provides unified connectivity for both SW and HW without needing any special MATLAB toolbox. This unified connectivity eliminates the need for additional MATLAB toolboxes and simplifies the development process for both SW and HW applications. Unlike platforms restricted to specific SW or HW configurations, USHWAP allows researchers to leverage their preferred SW, such as MATLAB or Python, for application development and seamlessly integrate multiple HW components with middle cost into their workflow. Additionally, while USHWAP involves a middle cost due to the utilization of multiple FPGA and SDRs, it offers the versatility required for device and edge computing applications. The proposed platform is unique in its ability to support the design and optimization of wireless systems and is compared to existing platforms in Table 3.1.

Table 3.1: Platform's comparison.

Ref	Target	Use case	Software	Hardware	Trigger mode	Connectivity	Wirelessly	Simulation	Cost
[31]	Specific	Signal modulation	MATLAB	Device computing	No	Single	Yes	No	Low
[32]	Specific	CSI sensing	MATLAB	Device computing	No	Single	Yes	No	Low
[33]	Specific	5G network	Embedded	Virtualized	No	Single	No	Yes	Low
[34]	Specific	Signal modulation	Python	Device computing	No	Single	Yes	Yes	Low
[35] [27]	Specific	5G network	Embedded	Cloud & edge computing	No	Single	Yes	Yes	High
[36]	Unified	Multi-purpose	Multiple programming	Device & edge computing (multiple devices)	No	Multiple	Yes	Yes	Middle
Proposed: USHWAP	**Unified**	**Multi-purpose: Ex, Packet detection (including WLAN and 5G signals); WLAN sensing; Rate adaptation**	**MATLAB & Python**	**Device & edge computing (multiple devices)**	**Yes**	**Multiple**	**Yes**	**Yes**	**Middle**

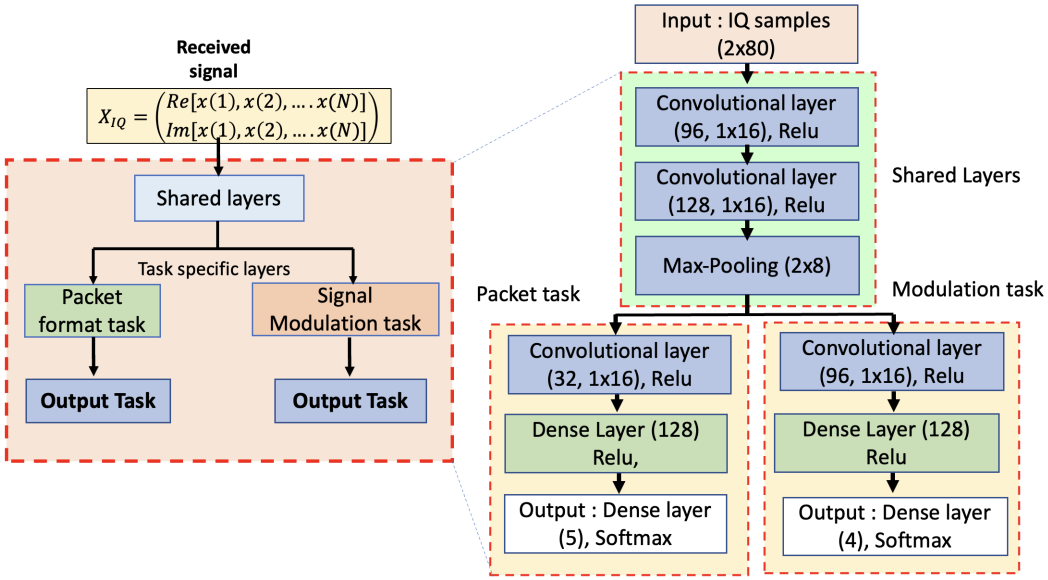


Figure 3.7: MTL-CNN structure.

3.4 Unified SW/HW Wireless-AI Platform Use Case

This section presents various use cases of USHWAP, including SW simulation, real-time HW experiments, and a hybrid approach combining SW simulation with HW implementation. These use cases showcase the versatility and flexibility of the platform, its ability to be applied to a wide range of applications, and its potential to improve the performance of wireless systems.

3.4.1 Wireless Signal Classification for Packet Format Detection

The first use case of evaluating USHWAP involves the SW simulation is wireless LAN classification using Multi-task Learning (MTL) with a convolutional neural network (CNN) for packet format detection. The next generation of Wi-Fi, known as Wi-Fi 6E, will operate in the 6 GHz frequency range, also shared by 5G New Radio Unlicensed (NR-U) networks. This may create interference issues that rely on Wi-Fi, especially in 6 GHz ranges, leading to reduced performance for both technologies. We have developed a packet format detection and modulation classification solution to address this challenge using the MTL-CNN

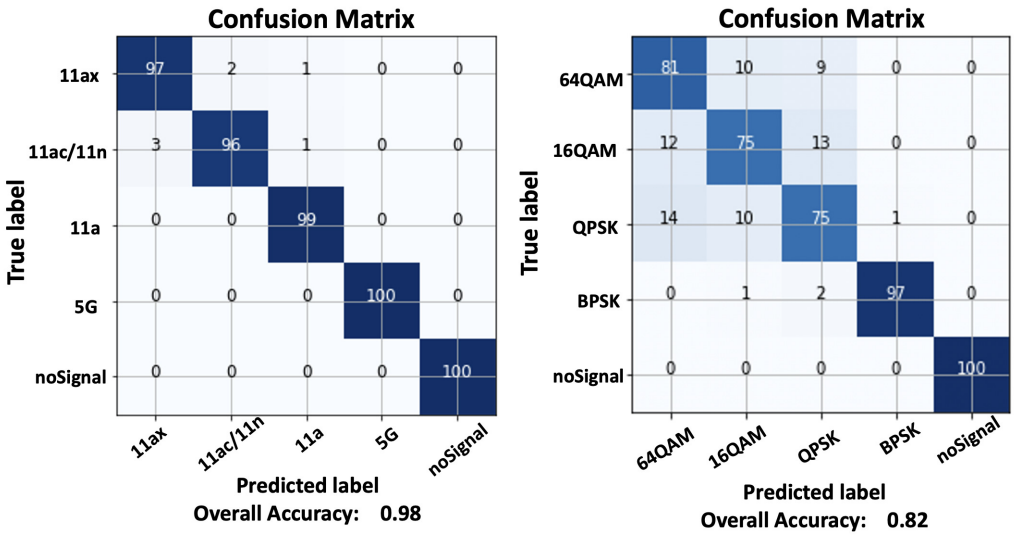


Figure 3.8: Confusion matrixes of MTL-CNN model.

model [17]. This model can identify various packet formats based on communication standards, including IEEE 802.11ax, IEEE 802.11ac, IEEE 802.11n, and IEEE 802.11a, as well as 5G NR frame packets and their respective modulation schemes.

Dataset Creation

The dataset used in this study consists of in-phase and quadrature parts of the orthogonal frequency division multiplexing (OFDM) symbol for various packet formats: IEEE 802.11ax, IEEE 802.11ac, IEEE 802.11n, and IEEE 802.11a, as well as 5G (DL-FRC-FR1-QPSK and DL-FRC-FR1-64QAM). The packets were generated using MATLAB waveform generator and subjected to additive white Gaussian noise (AWGN) with a signal-to-noise ratio (SNR) ranging from 4 to 24 dB. The same raw I/Q data is used for the training process, but different labels are assigned based on the packet format (11ax, 11ac/11n, 11a, 5G) and the signal modulation type (BPSK, QPSK, 16QAM, 64QAM). Only the data field part is retained, and the waveform preamble part is removed. Each data *block* consists of 80 samples, as the length of a *block* in the data field is 4μs, and the sampling frequency is 20 MSamples/s. The dataset is enhanced by incorporating the No-Signal scenarios at varying SNRs to enable the system to detect the absence of a signal accurately.

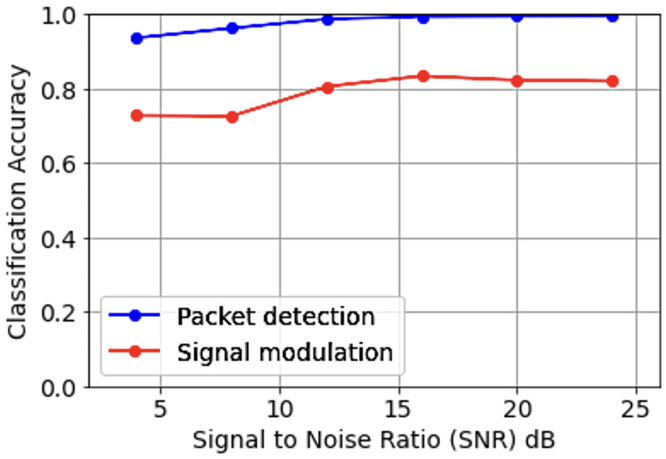


Figure 3.9: Classification performance for different SNR conditions.

MTL-CNN Model and Simulation Result

Using the same input, the MT-CNN model performs two tasks: packet detection and signal modulation. The MTL-CNN model improves performance by allowing the model to learn both tasks in parallel, utilizing shared knowledge such as model weights or gradients between tasks. In the MTL-CNN model, we share hard parameters in the hidden layers and separate task-specific layers. The proposed MTL-CNN model is shown in Fig. 3.7. The MTL-CNN model is developed and simulated in Python, utilizing TensorFlow and Keras on the USHWAP workstation. The training results show an accuracy rate of 0.98 for packet detection and 0.82 for modulation classification. The normalized confusion matrices for the MTL-CNN models are shown in Fig. 3.8. The MTL-CNN model is verified at various SNRs to demonstrate the ability of the model to classify the wireless signal. Figure 3.9 shows the performance of MTL-CNN under SNR = 4 to 24 dB. The results show the model performs well in classifying the WLAN signal for packet format detection and modulation classification. To implement the model on the HW platform, we convert the model-trained parameters from Python to MATLAB and then test it using actual signals from FPGA and SDR.

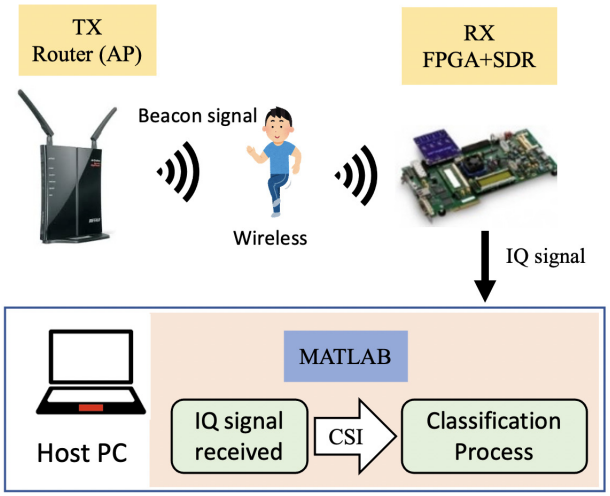


Figure 3.10: WLAN sensing system.

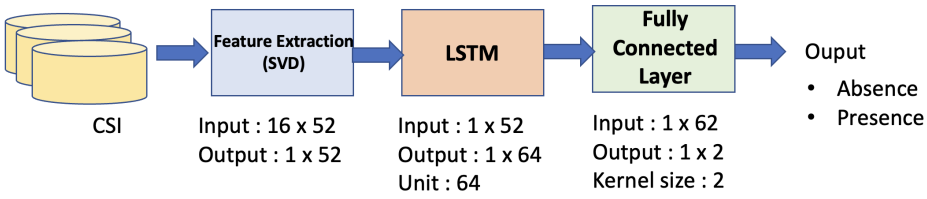


Figure 3.11: WLAN sensing network architecture.

3.4.2 Real-time Wireless LAN Sensing for Indoor Human Location

The second use case of USHWAP employed a real-time experiment of WLAN sensing for indoor human location detection application by analyzing variations of Channel State Information (CSI). The study focused on analyzing the CSI in an indoor setting by constantly monitoring beacon frames transmitted by WLAN access points (AP).

CSI is a metric used to measure the quality of a received signal in communication systems. It measures the channel frequency response for each antenna and subcarrier in OFDM systems such as WLAN. It is acquired through a channel estimation process and is composed of entries that reflect the Channel Frequency Response (CFR) defined as follows

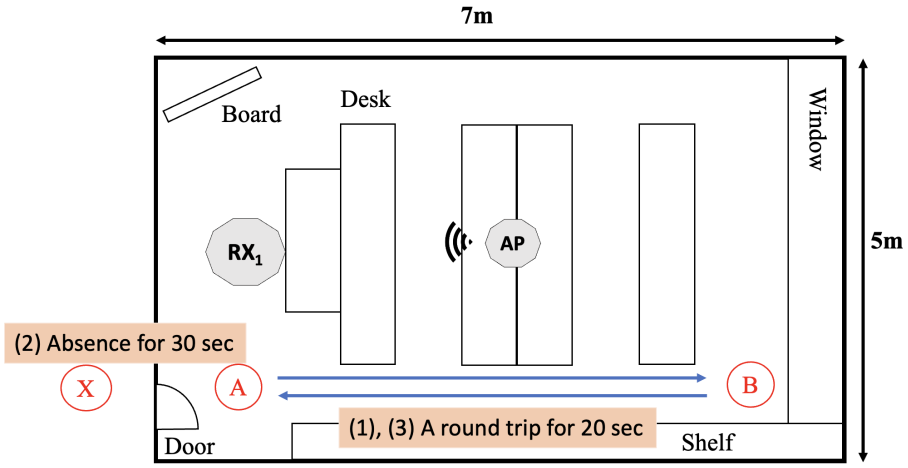


Figure 3.12: Experiment room setting.

Table 3.2: Hardware used

Device	Model	Configuration
AP (TX)	WZR-HP-AG300H (BUFFALO)	Sending beacon frame at interval 0.5s
RX	FPGA ZC706 (Xilinx. inc) SDR ADRV9371-W/PCR (Analog device. inc)	Receiving beacon frame at channel: 144 Ch (fc: 5720 MHz); bandwidth: 20 MHz

:

$$H(f;t) = \sum_{n}^{N} a_n(t) e^{-j2\pi f \tau_n(t)} \tag{3.1}$$

where CSI amplitude and phase, represented by $|H|$ and $\angle H$, respectively, are influenced by the movement and position of the transmitter, receiver, and any surrounding objects or humans. The amplitude attenuation factor, $a_n(t)$, and the propagation delay, $\tau_n(t)$, are also impacted by these factors and are dependent on the carrier frequency, f.

The CSI is presented as a matrix H for analysis. It comprises 52 subcarriers in the row direction, excluding null subcarriers used in OFDM communication (IEEE802.11a). In the context of the matrix H, the columns represent the CSI measurements for the total received packets (num_packets). Therefore, H is a singular complex matrix with dimensions

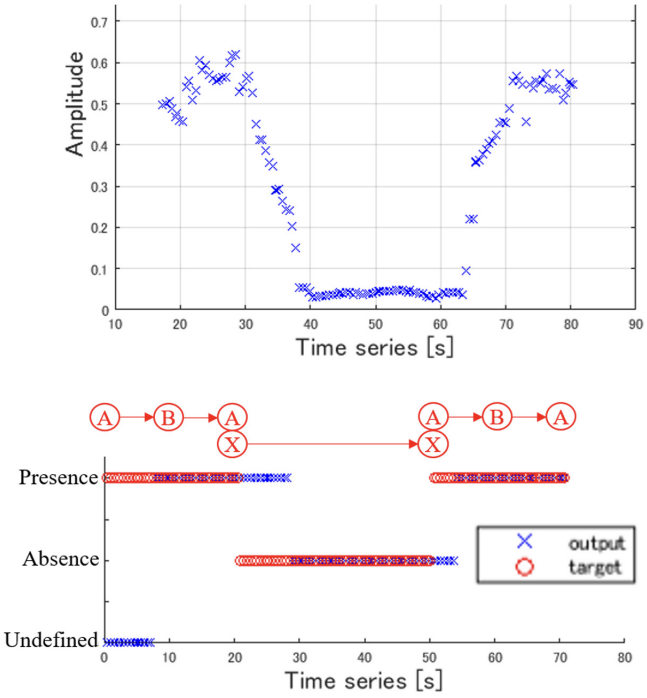


Figure 3.13: System output.

(num_packets) x 52.

Real-time Experiment on USHWAP

The experiment aimed to implement real-time WLAN sensing using USHWAP. The WLAN sensing system is illustrated in Fig. 3.10. In the experiment, the AP transmitted beacon frames at a cycle of 0.5 seconds. The Rx, consisting of an FPGA and an SDR with a 20 MHz bandwidth on channel 144 (frequency of 5720 MHz), received the beacon frames. The Rx captured the IQ signal comprising 52 subcarriers to obtain CSI. The HW used in the experiment is presented in Table 3.2.

Implementing the WLAN sensing network model, as depicted in Fig. 3.11, involves several key steps. Firstly, we apply singular value decomposition (SVD) to extract the CSI feature from the IQ signal of a WLAN. This extracted feature is then utilized as input for a long short-term memory (LSTM) model, which is trained to estimate the location of humans. The data analysis and implementation of the LSTM model are performed using

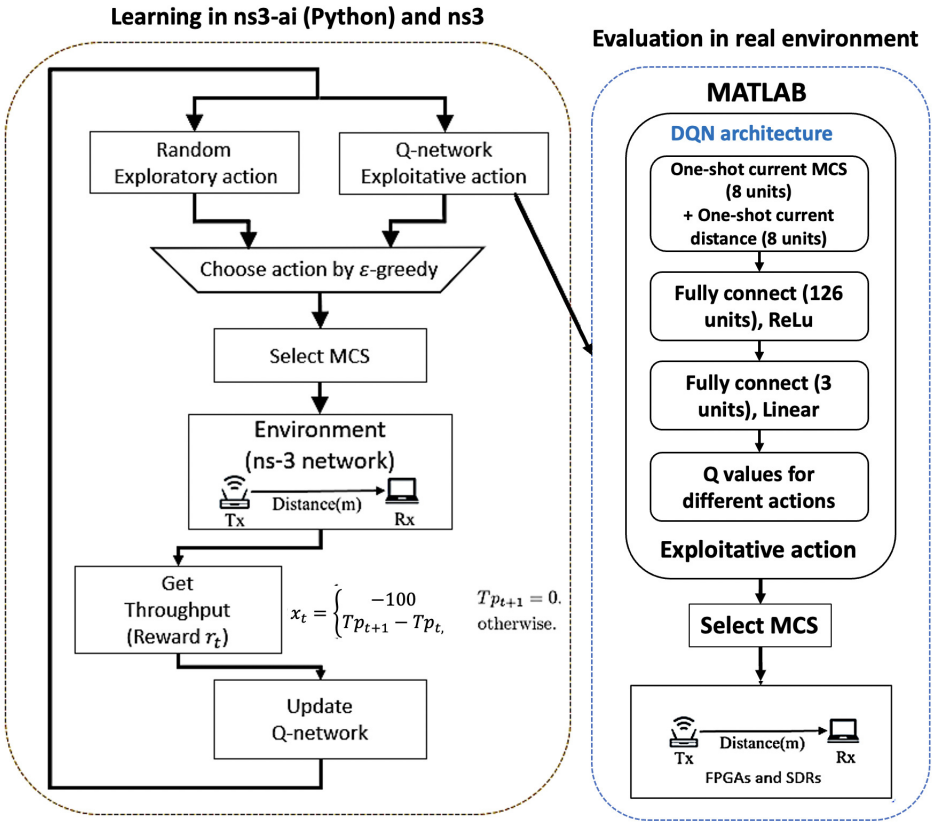


Figure 3.14: Simplified platform for rate adaptation.

MATLAB. To capture real-time data of the IQ signal, an FPGA is connected to MATLAB, serving as the real-time receiver signal.

The proposed system is tested by having human subjects walk from point A to point B and turn around, as depicted in Fig. 3.12. In these simulating real-world scenarios, human location needs to be estimated using WLAN sensing. The results are shown in Fig. 3.13. The system output is compared to the correct labels, and it is found to accurately estimate the locations of humans, with a delay of 8 seconds. This delay arises from the time required for data collection and storage of 16 beacons (500 ms x 16) packets, subsequent processing involving the SVD module, and the LSTM. The experimental results show that using WLAN sensing for human location classification is feasible, and the proposed system is an effective solution for this task.

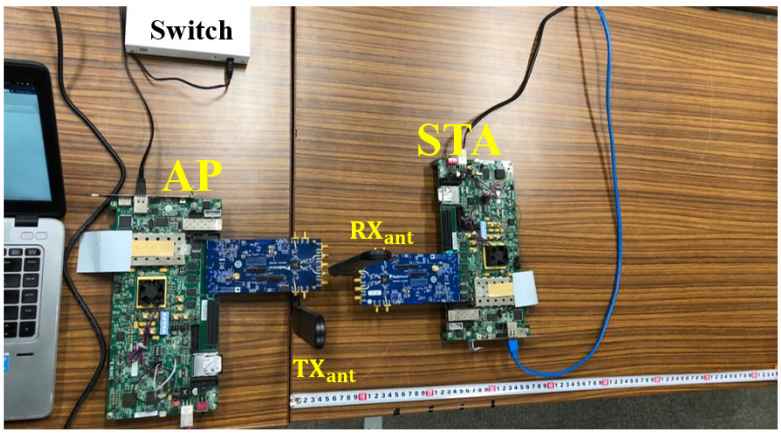


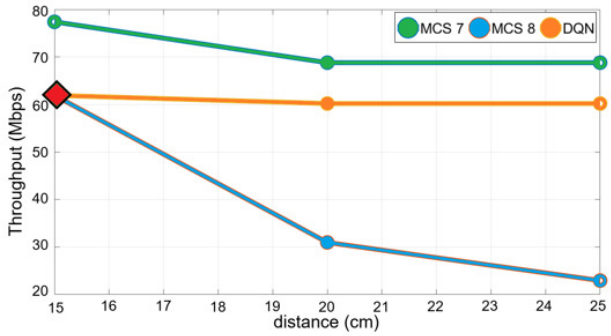
Figure 3.15: Experiment setup.

3.4.3 Rate Adaptation for Network Optimization

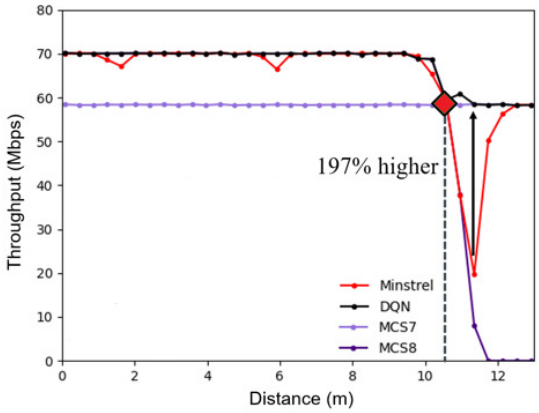
The third case implemented on USHWAP is a rate adaptation for network optimization. Rate adaptation is a technique used to enhance network performance by dynamically adjusting the data transmission rate. This is achieved by monitoring network conditions and adjusting the rate based on available bandwidth, packet loss, and delay. The goal of rate adaptation is to ensure that nodes transmit at a suitable rate that minimizes packet errors on the receiving side. To achieve this, we have developed a deep Q-network (DQN) model that adaptively changes the transmission rate for a moving node [38]. The effectiveness of this model has been assessed through real-time experimentation and compared to simulation using the Minstrel Rate Adaptation algorithm. Minstrel is a rate control algorithm widely employed in the Linux kernel for Wi-Fi transmission. It operates by selecting an MCS based on a sampling process and collects statistics on transmissions for each MCS.

Implementation and Experiment Result

Our research focuses on the scenario where downlink transmissions occur while a station (STA) moves away. Figure 3.14 shows the simplified platform to develop the proposed method. We first trained the proposed DQN model in a platform called ns3-ai. After that, we converted the DQN model to be used in MATLAB. We conduct experiments on the HW platform with the same moving scenario to evaluate the rate adaptation method. Fig. 3.15



(a)



(b)

Figure 3.16: Comparison performance evaluation: (a) Experiment, (b) Simulation.

shows the experiment setup. The experiments include 1) normal downlink transmission using MCS 8, 2) normal downlink transmission using MCS 7, and 3) downlink transmission with rate adaptation applied.

The results of the experiment are presented in Fig. 3.16. The AP in the experiment was configured with a transmit power restriction of 0 dBm, while in the simulation, a transmit power of 20 dBm was used. This deliberate difference in transmit power settings was chosen to showcase the flexibility and adaptability of the proposed platform. In Fig. 3.16(a), at a distance of 15 cm, which roughly corresponds to 10.5 m in the simulation, there is a reduction in throughput compared to Fig. 3.16(b). This difference can be attributed to several factors, including a potential 37 dB discrepancy between the simulation and experiment due to factors like noise figure (NF), Tx/Rx gain, and implementation margin (IM). When MCS 8 was utilized, a decrease in throughput was observed as the distance increased. The

variation in throughput between Fig. 3.16(a) and Fig. 3.16(b) with the application of MCS 8 can be attributed to different channel conditions. The simulation in ns-3 employed a simplified channel model, which resulted in higher throughput. However, when rate adaptation with DQN was applied to the AP, the throughput remained relatively consistent, with minimal degradation observed.

3.5 Summary

This chapter has presented the conception of a unified software and hardware wireless AI platform (USHWAP) that offers a versatile and inclusive solution for enhancing wireless systems through machine learning. USHWAP establishes an environment where software (SW) and hardware (HW) are unified, fostering the creation of wireless AI applications encompassing both software simulations and real-time hardware implementations. Its flexibility and unified approach towards software and hardware development streamline integration and ensure scalability, rendering it adaptable for diverse AI-driven wireless systems applications.

We have demonstrated three distinct use cases: wireless signal classification for software simulation, real-time wireless LAN sensing for real-time experiments, and rate adaptation application employing a hybrid approach that combines software simulation with hardware implementation. The results affirm the capability of the proposed platform for comprehensive development and evaluation across various scenarios, encompassing both software simulation and real hardware testing, as demonstrated by the use case performance.

Chapter 4

Multi-Task Learning CNN for WLAN

4.1 Introduction

The IEEE 802.11 WLAN (Wi-Fi) dominates in facilitating the wireless network in the upcoming decade [39]. In wireless networks, PHY layer technologies govern the transmission of information from the transmitter to the receiver over the channel. Signal detection and modulation recognition are crucial in identifying the transmitted information from noisy signals. Optimal detectors, such as the maximum likelihood detector, can be formulated using statistical models with defined distributions. However, they often entail a computational complexity that could not be practical for real-world applications [40]. DL-based signal detectors have been proposed recently to address these drawbacks.

The WLAN signal consists of two main parts: the preamble and the data field. The conventional method uses the preamble for detecting the frame format of the incoming WLAN signal. This process primarily relies on symbol synchronization and channel estimation (CE). Figure 4.1 illustrates the frame formats of IEEE 802.11a/n/ac/ax: legacy, High Throughput - Mixed Format (HT-MF), Very High Throughput (VHT), and High Efficiency (HE). The legacy-short training field (L-STF) and legacy-long training field (L-LTF) are the training symbols. Initially, the incoming WLAN signal undergoes symbol synchronization, L-LTF extraction, L-LTF demodulation, and CE/channel compensation. Following this, the three symbols after the L-LTF are employed for frame format detection. These three symbols utilize binary phase shift keying (BPSK) and quadrature BPSK (QBPSK)

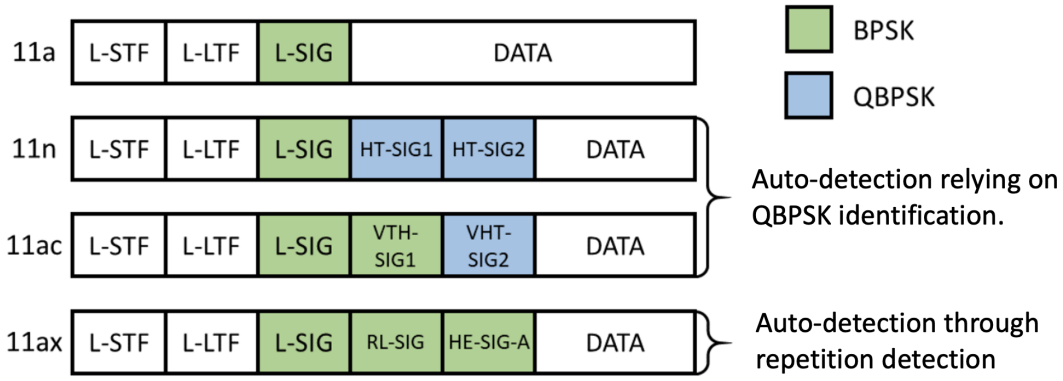


Figure 4.1: Preamble auto-detection for WLAN packet.

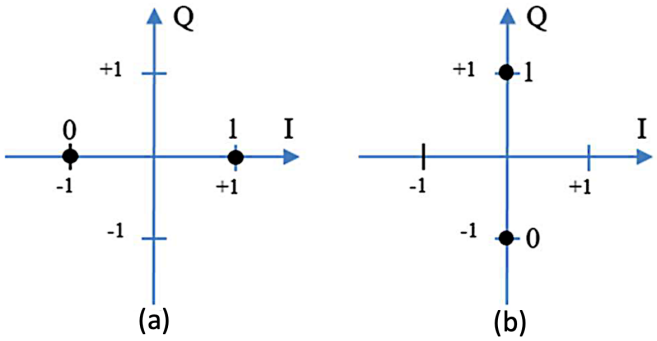


Figure 4.2: Constellation diagram of (a) BPSK, and (b) QBPSK.

with different constellation diagrams, as depicted in Fig. 4.1. Figure 4.2 shows the constellation diagram of BPSK and QBPSK. This allows the receiver to distinguish between different frame formats. In this conventional, the problem arises when the receiver misses the preamble part of the incoming packet, resulting in the inability to identify the packet frame format.

In addition, Automatic Modulation Classification (AMC) is crucial for noncooperative communication systems, as it automatically identifies the modulation type of the transmitted signal. In a basic communication system model (see Fig. 4.3), AMC methods are categorized into two groups: likelihood-based (LB) and feature-based (FB) [41]. LB methods focus on the likelihood function of the received signal, while FB methods rely on feature

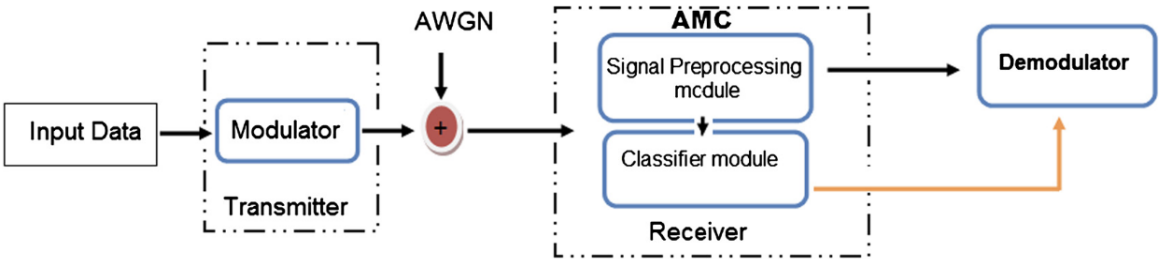


Figure 4.3: AMC on communication system model [4].

extraction and classifier design. LB methods aim for optimal solutions but are computationally complex and require prior information from transmitters. FB methods provide suboptimal solutions with lower computational complexity and do not rely on prior information. However, they depend on a manually designed feature set, which may not be practical for all scenarios. In contrast, DL methods can automatically extract features that have been adopted.

DL using CNN models has been proposed for WLAN packet format detection [42]. Moreover, CNNs employing single-task learning have been investigated for AMC in signal modulation detection [43] [44] [45] [46]. However, previous studies have treated signal detection for packet frame format and modulation recognition as separate tasks using single-task learning. These approaches are inefficient, particularly regarding the number of weight parameters in the neural network model.

Multi-task learning (MTL) trains the DL model to handle multiple tasks simultaneously. Instead of learning each task separately, the model learns them together, which can lead to better performance overall [47]. This chapter proposes the MTL-CNN method for WLAN's packet format detection and modulation classification. The objective is to identify the presence of 802.11 WLAN signals when the preamble part is missed.

The motivation for employing MTL over single-task learning arises to enhance model performance and generalization by leveraging shared knowledge across multiple related tasks. In single-task learning, a model is trained to perform a specific task, optimizing its parameters solely for that objective. While effective for that particular task, single-task models might struggle when faced with diverse or evolving data.

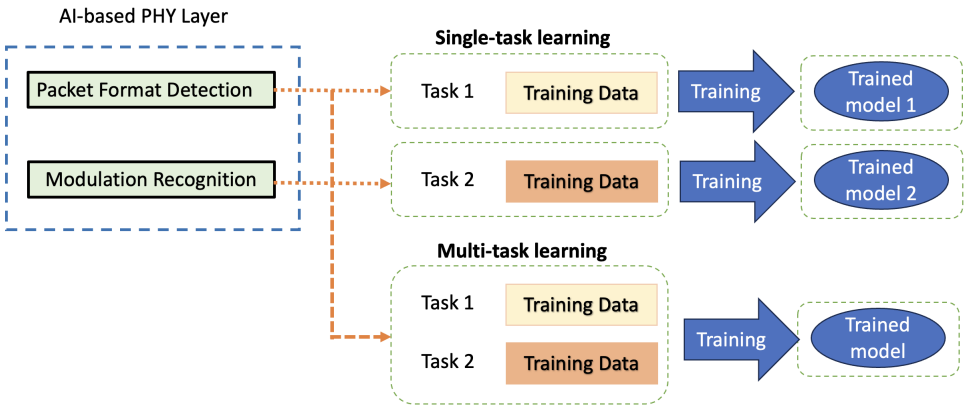


Figure 4.4: MTL for packet format detection and modulation recognition compared to single-task learning.

MTL, on the other hand, involves training a model on multiple tasks simultaneously. The intuition is that the shared representation learned across tasks can more robustly capture underlying patterns and relationships. This shared knowledge can act as a form of regularization, preventing overfitting and improving the model's ability to generalize to new, unseen data. Additionally, MTL can be particularly advantageous when tasks have shared features or when data for one task is limited, as knowledge transfer from other tasks can enhance performance.

Figure. 4.4 illustrates MTL for packet format detection and modulation recognition compared to single-task learning. The MTL architecture utilizes shared representation layers. By jointly modeling packet format and modulation classification, the model leverages the shared knowledge in these layers to enhance precision. This approach not only improves the efficiency of the learning process by training both tasks simultaneously but also results in a more lightweight neural network compared to separate tasks using single-task learning.

4.2 Proposed MTL-CNN for WLAN Applications

In this section, we explore the potential of the proposed MTL-CNN to address several challenges in WLAN. This includes improving Clear Channel Assessment (CCA) signal

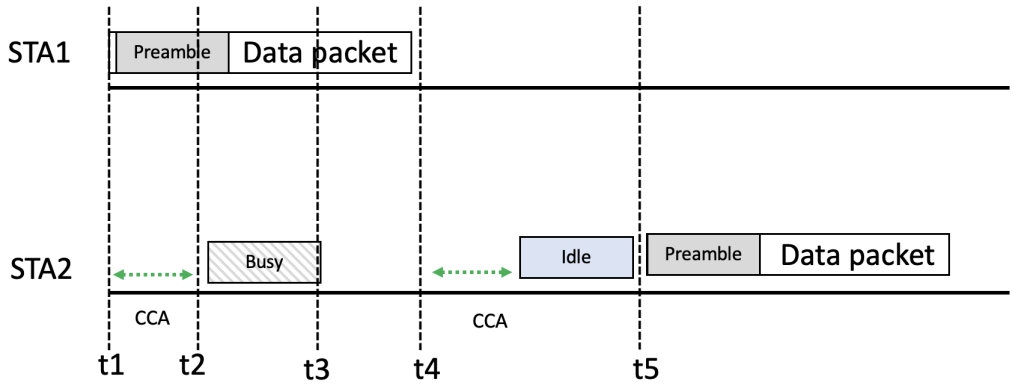


Figure 4.5: Conventional CCA.

detection to prevent packet collisions and enhancing the coexistence of Wi-Fi and 5G NR. The primary focus of this chapter is on developing and evaluating the MTL-CNN model.

4.2.1 MTL-CNN to Enhance CCA Sensitivity

The IEEE 802.11 uses CCA to monitor the channel and distinguish between (*busy*) and (*idle*) states. CCA aims to optimize WLAN performance by reducing signal interference. There are two distinct CCA methods. The first, CCA-SD, primarily focuses on the preamble part of 802.11 signals. The second method, CCA-ED, is for non-802.11 transmissions. To detect the presence of a signal at any STA, the standard requires a minimum received power of -82 dBm for CCA-SD and -62 dBm for non-802.11 signals using CCA-ED. This corresponds to signal-to-noise ratios (SNR) ranging from 4 dB to 24 dB. This power threshold is standardized for signals with a bandwidth (BW) of 20 MHz, with adjustments made for signals exceeding this BW.

In CCA-ED, the STA assesses the ambient power level in the channel and uses it along with the threshold to determine channel occupancy. However, in cases where the preamble segment is missed or the received power falls below the threshold, the STA cannot extract information from the ongoing WLAN packet [2]. Therefore, further research is needed to explore methods for detecting WLAN packets in scenarios where the preamble segment is absent [48].

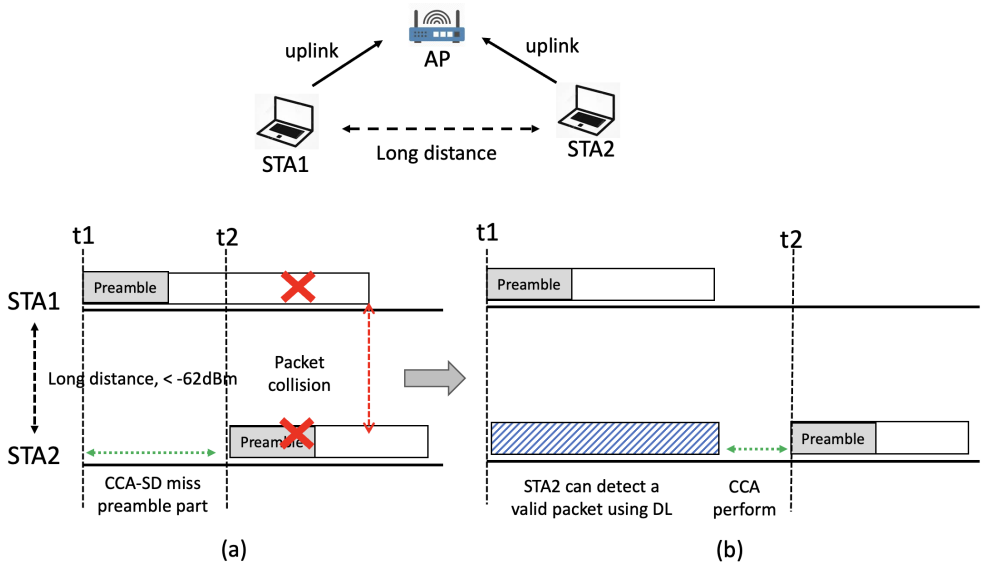


Figure 4.6: (a). Packet collision occurs when CCA-ED and CCA-SD fail the preamble part (b). Proposed MTL-CNN solution.

1. Packet Collision Avoidance

In the conventional CCA, STAs utilize the CCA mechanism to determine whether the channel is currently in use (*busy*) or available (*idle*). Fig. 4.5 illustrates the conventional CCA process. In the *busy* state, the STA must maintain its back-off counter. Conversely, in the *idle* state, the STA begins decrementing its back-off counter until it reaches 0, at which point it can transmit its packet, as seen in Fig. 4.5. Upon detecting the preamble of a packet (at t2 in Fig. 4.5), the STA gains information about the packet's frame format and proceeds to process the data symbols in the received packet. The CCA boasts a 90% probability of successfully detecting a valid OFDM signal within a 4μs timeframe [49].

The issue arises when CCA-SD lacks the 802.11 PHY preamble and the received power falls below the CCS-ED threshold. This scenario is depicted in the timing diagram in Fig. 4.6. STA-1 and STA-2 are going to send packets (uplink) to the access point (AP). STA-1 has already initiated a packet transmission, while STA-2 intends to commence a new transmission. STA-2 performs a CCA check before transmission, but it fails to detect the preamble part of the 802.11 PHY. During this period, STA-1 and STA-2 are significantly distant from each other, as the received signal power from STA-1 falls below the CCA-ED

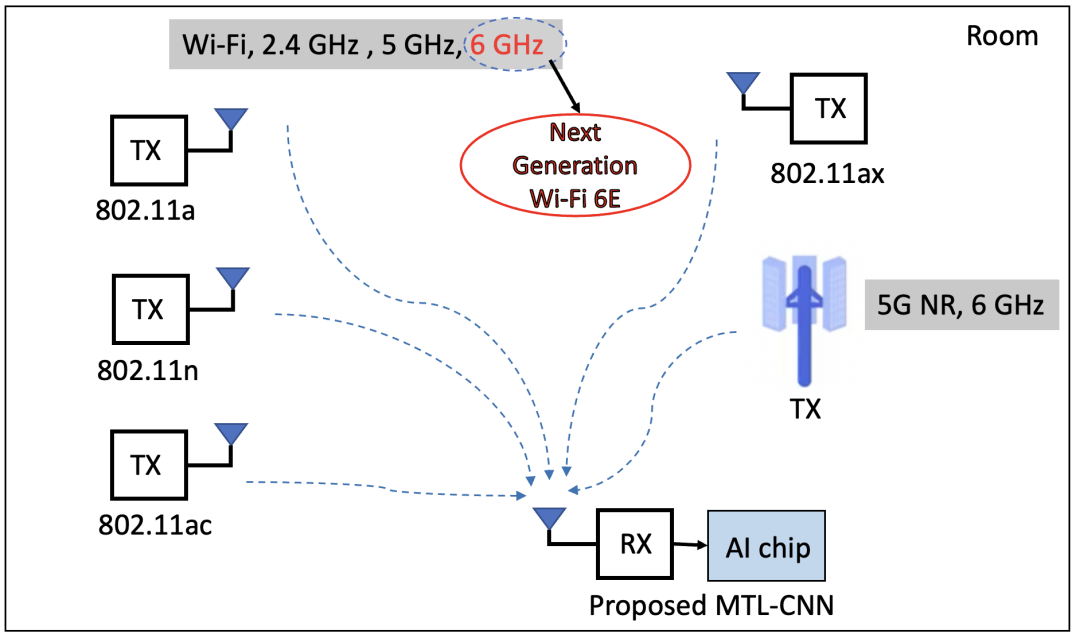


Figure 4.7: Wi-Fi and 5G Network Coexistence.

threshold. Assuming the channel is idle, STA-2 initiates its back-off counter and proceeds to transmit a packet, resulting in packet collisions. This situation is illustrated in Fig. 4.6(a).

We propose a novel solution using MTL-CNN to address this challenge. MTL-CNN empowers STAs to discern signals occupying the channel, even when the preamble segment is absent. Furthermore, MTL-CNN performs classification tasks to determine the received signal's packet format and modulation type. STAs temporarily suspend their back-off counters while the channel is engaged, subsequently reinitiating the CCA before transmitting their packets. The implementation of the proposed MTL-CNN approach is depicted in Figure 4.6(b).

2. Coexistence of Wi-Fi and 5G Networks

The increasing number of communication devices poses a challenge for wireless technologies, particularly those relying on license-free spectrums like 802.11 or Wi-Fi. Multiple Wi-Fi networks can occupy the same spectrum in densely populated areas, resulting in packet collisions and reduced network throughput. The recent introduction of the 6 GHz

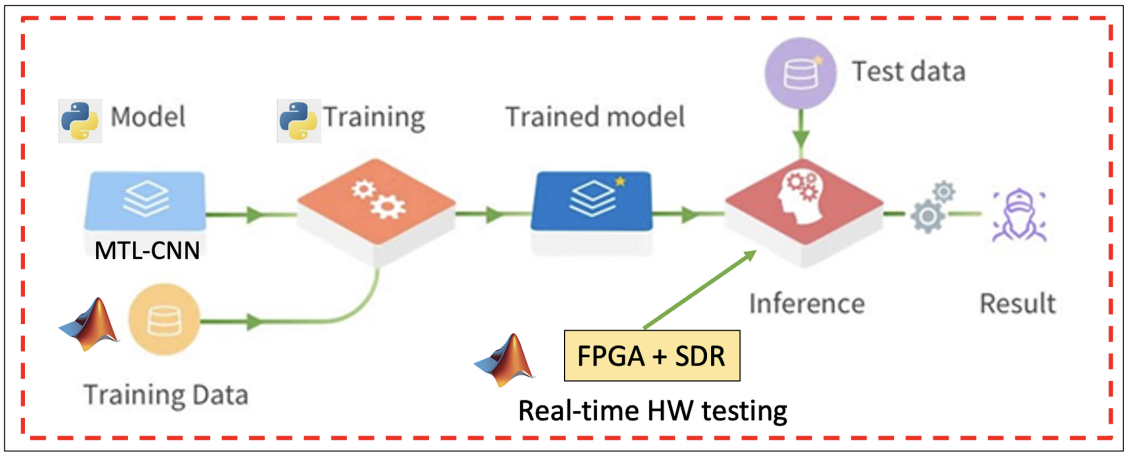


Figure 4.8: Design Workflow of MTL-CNN on USHWAP

spectrum aims to alleviate this issue by providing additional bandwidth.

Wi-Fi devices now have the option to operate in three different bands: 2.4GHz, 5GHz, and 6GHz. The next generation of Wi-Fi, known as Wi-Fi 6E, will operate in the 6 GHz frequency range, also shared by 5G NR networks. This may create interference issues that rely on Wi-Fi, especially in 6 GHz ranges, leading to reduced performance for both technologies. Fig. 4.7 illustrates the coexistence of Wi-Fi and 5G NR networks.

The proposed MTL-CNN approach addresses this issue by improving the coexistence of 5G and Wi-Fi networks. It identifies and classifies packets transmitted over the networks according to their respective communication standards. The MTL-CNN model is capable of recognizing a range of packet formats according to communication standards, encompassing IEEE 802.11ax, IEEE 802.11ac, IEEE 802.11n, and IEEE 802.11a, in addition to 5G NR frame packets along with their associated modulation schemes. This approach can facilitate more efficient use of the wireless spectrum, enhancing the performance of wireless communication systems.

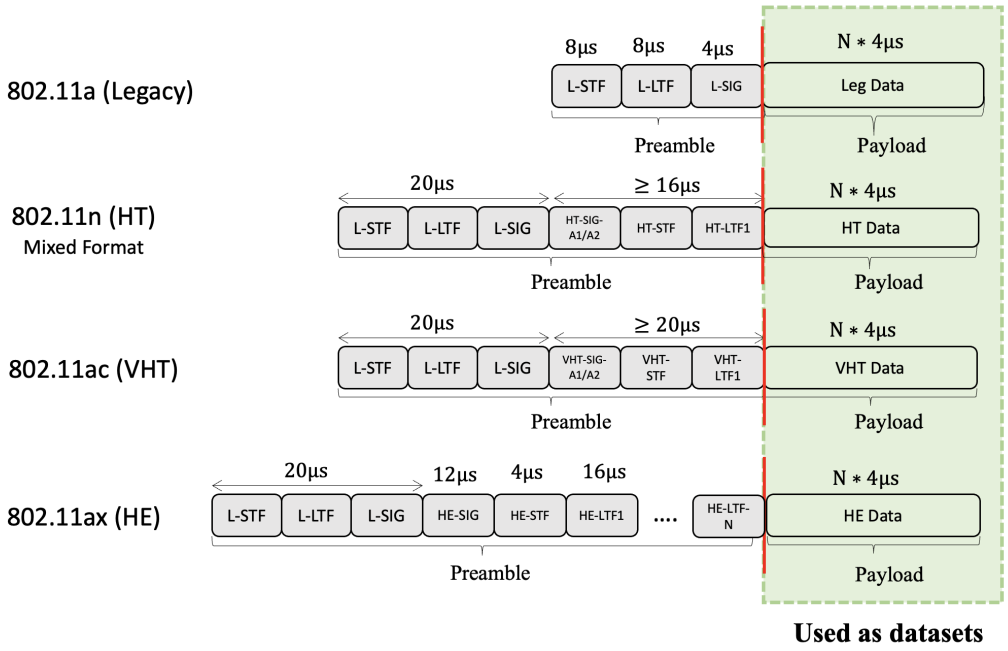


Figure 4.9: Wi-Fi 802.11 Sample Dataset

4.2.2 MTL-CNN Workflow on USHWAP

MTL-CNN is implemented on USHWAP for development and testing, although alternative platforms are also viable. Leveraging USHWAP offers distinct advantages, primarily in increased efficiency and flexibility. USHWAP provides unified connectivity, accommodating multi-SW (MATLAB and Python) and HW modes. This flexibility empowers researchers to choose their preferred SW for simulations while also enabling real-time HW testing. Fig. 4.8 depicts the design workflow of MTL-CNN on USHWAP. In the MTL-CNN development stage, datasets are initially created using the MATLAB waveform generator. MTL-CNN's algorithms are developed and trained in the Python environment. The algorithm evaluation in a real testing scenario uses FPGA and SDR. The integrated approach of USHWAP enhances the overall efficiency of the implementation, making it a valuable tool for developing and testing MTL-CNN models for wireless applications.

4.2.3 Dataset Creation

The dataset is compiled using OFDM waveforms from four WLAN standards, covering in-phase and quadrature (IQ) samples. Specifically, it encompasses 802.11 WLAN waveforms for the following standards: HE, VHT, HT, and legacy, corresponding to IEEE 802.11ax, IEEE 802.11ac, IEEE 802.11n, and IEEE 802.11a standards, respectively. The dataset includes 3GPP-based 5G NR frame packets and their respective modulation schemes (DL-FRC-FR1-QPSK and DL-FRC-FR1-64QAM). The 5G NR signals are included due to next-generation Wi-Fi (Wi-Fi 6E), which will operate in the same frequency band as the 5G NR in the 6GHz spectrum. We have added a 'No-Signal' category to the datasets to account for situations where the signal may be absent. The signals are generated using the MATLAB-waveform generator according to specific standards for training and testing.

The generated OFDM symbol in the dataset is represented as a complex (IQ) signal sequence denoted by $X = \{x(n)\}_{n=1}^{N}$, where x is an OFDM symbol. To prevent scaling issues, we normalize the power of X, resulting in the normalized received sequence $\hat{X} = \frac{X}{\sum_{n=1}^{N} |x(n)|^2}$.

The real (x_I) and imaginary (x_Q) components of an OFDM symbol are defined as $x = x_I + ix_Q$. We represent the components of the sequence of OFDM symbols as matrices function:

$$f\{x_n\} = \begin{bmatrix} x_I^n \\ x_Q^n \end{bmatrix} \in \mathbb{C}^N \tag{4.1}$$

Here, N denotes the symbol length, and n signifies the index used to extract the signal for inclusion in the datasets. Subsequently, the real ($\Re(x)$) and imaginary ($\Im(x)$) parts are combined into a matrix with dimensions $2 \times N$, which is treated as a single sample for training or testing purposes.

The 802.11 waveform comprises two main components: the preamble and the data field. The length and structure of the preamble vary for each standard, depending on the specific standard. The length of the preamble for each standard is shown in Table 2.2 in Chapter 2. In this study, the preamble is omitted, and only the data field segment is retained to represent a data symbol (referred to as a *block*). We utilize this data field segment as the datasets, as depicted in Fig. 4.9.

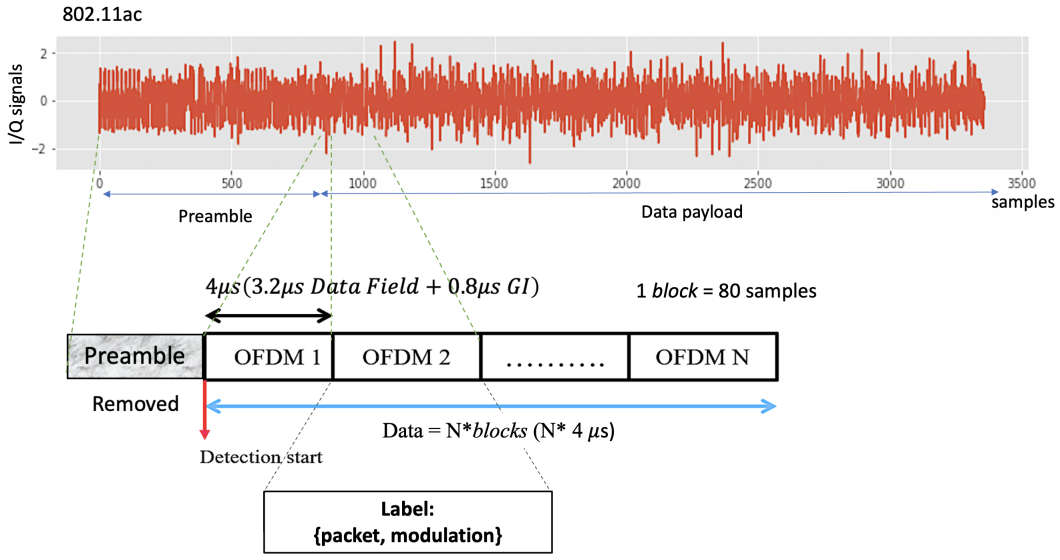


Figure 4.10: 802.11ac waveform sample datasets.

Each data *block* has a duration of 4µs, comprising 3.2µs of OFDM symbols and 0.8µs of guard intervals (GI), sampled at a frequency of 20 MSample/s. Consequently, each *block* is composed of 80 samples. The WLAN waveform within the datasets is associated with packet format frame and modulation coding schemes (MCS) ranging from MCS 0 to MCS 7, employing modulation types such as BPSK, QPSK, 16QAM, and 64QAM. The MATLAB waveform generator facilitates the generation of actual WLAN and 5G NR waveforms. This process involves applying the channel model delay-B. Then, we added Additive White Gaussian Noise (AWGN) with SNR spanning from 4 dB to 24 dB, corresponding to the CCA-SD and CCA-ED thresholds. Fig. 4.10 illustrates the dataset creation process for the 802.11ac waveform.

The dataset includes integrated timing offsets as data augmentation. In previous studies [43–46, 50–53], the influence of timing offsets was not investigated, as it was assumed that packet detection consistently occurred at the beginning of symbols. However, in practical scenarios, received packet detection within the 4µs window may not always align precisely with the symbol's commencement, and there can be some shifting during this interval. Consequently, assessing the effect of timing offsets on the proposed model becomes crucial.

To introduce timing offsets into the data, the OFDM symbol is shifted from 0% to 90%

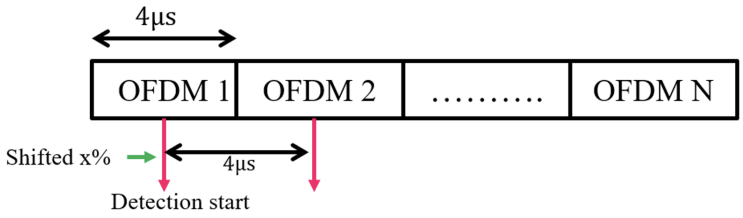


Figure 4.11: Timing offset of the augmented dataset

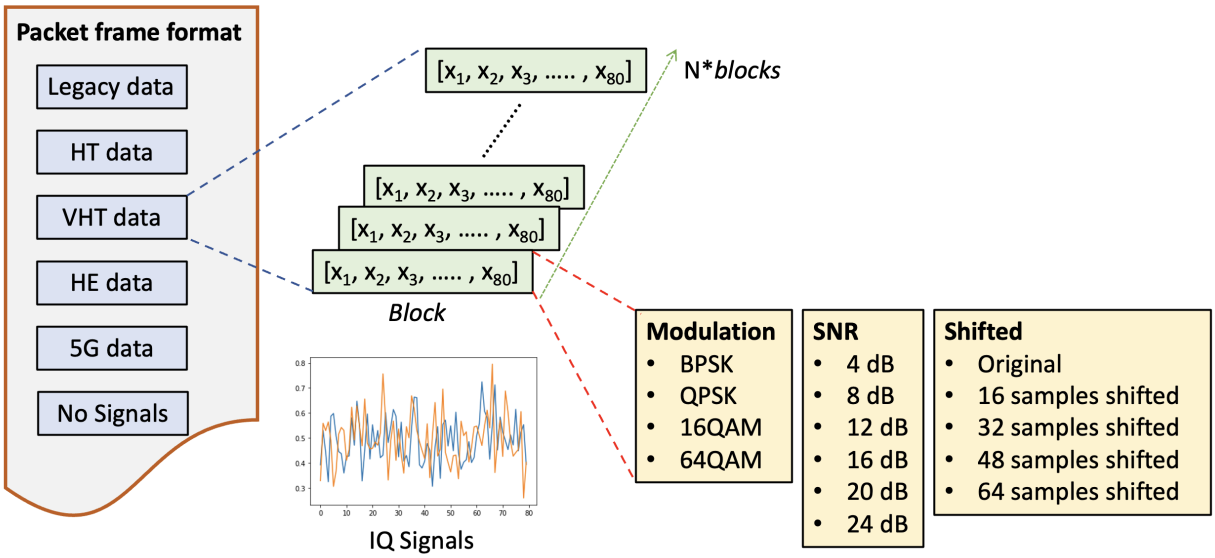


Figure 4.12: Dataset structure.

relative to the original data. The reason for considering a maximum shift of 90% is the uncertainty associated with the shifting percentage, influenced by environmental factors when implemented in real-world applications. The shifted data packets are depicted in Fig. 4.11.

In each *block*, two labels as the class are embedded: packet frame format (Legacy, HT/VHT, HE, 5G, No-Signal) and modulation (BPSK, QPSK, 16QAM, 64QAM). We combine the labels for HT and VHT into HT/VHT, as they share the same number of subcarriers, see Table 2.1 in Chapter 2. This distinct subcarrier count is a unique feature, enabling CNN to distinguish between different standards. The structure of datasets can be seen in Fig. 4.12.

4.2.4 MTL-CNN Architecture Model

MTL is a neural network paradigm for inductive knowledge transfer, enhancing generalization by learning shared representations between related tasks. MTL improves learning efficiency and prediction accuracy for each task instead of training a single-task learning model for each task. The underlying concept is that knowledge acquired for each task can benefit learning other tasks. In the MTL approach, shared layers transfer the learned features between distinct tasks.

The MTL-CNN model is designed to perform two tasks: WLAN packet format detection (including legacy, HT/VHT, HE, 5G signals, and No-Signal) and signal modulation classification (including BPSK, QPSK, 16QAM, and 64QAM). The MTL model enhances performance by concurrently learning tasks and sharing knowledge, such as model weights or gradients, between them. This is achieved by employing shared parameters in the hidden layers of the MTL model while retaining task-specific layers for signal classification. In the proposed MTL-CNN architecture, the shared hidden layers consist of five convolutional layers, one max-pooling layer, and one dense layer, with a total of 767,494 parameters. We optimized the shared layer to minimize the number of weight parameters. Subsequently, each task-specific portion incorporates output softmax classification layers. The proposed MTL-CNN model, employing hard parameter sharing, is depicted in Figure 4.13.

In the implemented MTL-CNN, the objective is to jointly optimize the model for two distinct tasks: packet format identification (L_p) and modulation recognition (L_m). The categorical cross-entropy loss function is employed to quantify the model's performance in each task. The overall loss function (L), when simultaneously performing both tasks, is expressed as a weighted sum of the losses for packet format detection and modulation classification. The weights (w_p and w_m) assigned to each task determine their relative importance in the optimization process.

The model's loss is defined mathematically as:

$$L(\theta_{shd}, \theta_p, \theta_m) = w_p L_p(\theta_{shd}, \theta_p) + w_m L_m(\theta_{shd}, \theta_m) \tag{4.2}$$

Here, θ_{shd} represents the parameters of the shared layer, while θ_p and θ_m denote the

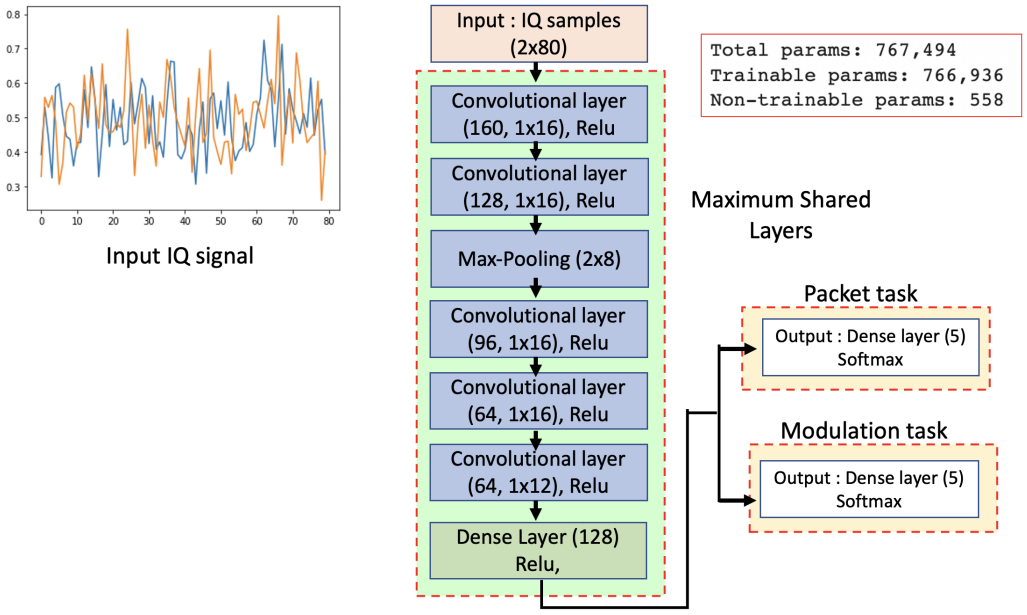


Figure 4.13: The proposed MTL-CNN model.

parameters specific to the packet format detection and modulation recognition tasks, respectively.

The optimization process aims to minimize the total loss in the MTL-CNN, and it is denoted as:

$$\theta^* = \arg \min_{\theta_{shd}, \theta_p, \theta_m} L(\theta_{shd}, \theta_p, \theta_m) \tag{4.3}$$

This optimization involves adjusting the parameters (θ_{shd}, θ_p, θ_m) to find the optimal configuration that minimizes the combined loss across both tasks. By sharing certain layers (θ_{shd}) and having task-specific layers (θ_p, θ_m), the model can effectively learn and extract features relevant to each task, leveraging shared information while adapting to the specific requirements of individual tasks. This approach encourages the model to discover representations that benefit both tasks, enhancing overall performance and efficiency in learning from multi-modal data.

The generated one OFDM *block* is a complex (IQ) signal with a length of 80 samples. Since the neural network deals with real numbers, we represent the IQ signal as real-valued.

Mathematically, the transformation of the IQ signal to real-valued can be shown with the relation:

$$f_{\text{trans}} : \mathbb{C}^{80} \rightarrow \mathbb{R}^{2\times 80} \tag{4.4}$$

The signal with size 2×80 samples as the input layer feeds into the network. The convolution in the CNN is the filtering process of the complex signal with a kernel filter feature. We define the kernel filter feature as the impulse response $h(t)$. The kernel filter is the real-valued impulse response: $h \in \mathbb{R}^{k\times l}$, where $k = 1$, l is the length of the kernel. The convolutional process is defined as:

$$z(t) = (x_R(t) + x_I(t)) * h_R(t) \tag{4.5}$$

$$z(t) = (x_R(t) * h_R(t)) + (x_I(t) * h_R(t)) \tag{4.6}$$

The real and imaginary parts are processed with the same impulse response, resulting in a complex value. In CNN, 2-D convolution is computed as a cross-correlation process. The convolution is performed on the input signal using a filter to produce a feature map. The complex signal $x(t)$ can be represented as $x \in \mathbb{R}^{m\times n}$, where $m = 2$ is the IQ representation, and n is the time index. The process of 2-D convolution is computed as:

$$z(m, n) = \sum_k \sum_l x(m + k, n + l) \cdot h(k, l) \tag{4.7}$$

The size of the filter h determines the size of the output value. The 2-D convolution output is $z \in \mathbb{R}^{m\times n}$. The max-pooling captures the highest value of x_j in N and gives important features to the next layer because the highest value of x_j is unique for different modulations. The max-pooling layer is applied after the convolutional layer.

We utilize the rectified linear unit (ReLU) as the non-linear activation function, defined as:

$$\sigma_{\text{ReLU}}(x) = \max\{0, x\} \tag{4.8}$$

Here, x is the previous input of the network. Dropout is employed to prevent overfitting

and weight updating for specific nodes. A dropout rate of 0.2 is set for the shared layer. The task-specific output involves softmax classification to predict the probability for five packet formats and four signal modulation classes. The softmax in this process is defined as:

$$\sigma_{\text{Softmax}}(x_j) = \frac{e^{x_j}}{\sum_{n=0}^{K-1} e^{x_k}} \tag{4.9}$$

The softmax function for the j-th element of the input vector (x_j) is calculated by taking the exponential of x_j and dividing it by the sum of the exponentials of all elements in the vector. This normalizes the values, ensuring that the resulting vector is a valid probability distribution, with each element representing the likelihood of the corresponding class. The softmax function is commonly used as the activation function in the output layer of a neural network for multi-class classification problems.

4.2.5 Loss Comparison of MTL versus Single-Task Learning

We elaborate on the benefits of MTL over single-task learning (STL). The key advantage of MTL is the efficient utilization of shared information, potentially resulting in improved generalization and feature extraction across tasks.

Let $L_p^{(\text{STL})}$ be the loss function for packet format identification in STL, and θ_p represents the optimized parameters for this task:

$$L_p^{(\text{STL})}(\theta_p) = -\sum_j x_{p,j} \log(\hat{x}_{p,j})$$

In MTL, the loss function $L_p^{(\text{MTL})}$ for packet format identification is incorporated with the modulation recognition task. Let θ_shd represent shared parameters, θ_p represent parameters specific to packet format identification, and θ_m represent parameters specific to modulation recognition:

$$L_p^{(\text{MTL})}(\theta_{shd}, \theta_p, \theta_m) = w_p L_p^{(\text{STL})}(\theta_p) + w_m L_m^{(\text{STL})}(\theta_m)$$

Here,

- $L_m^{(\mathrm{STL})}(\theta_m)$ is the loss for modulation recognition in STL.
- w_p and w_m are weights determining the importance of each task.

Combined MTL Loss: The overall MTL loss ($L^{(\mathrm{MTL})}$) is the sum of the losses for packet format identification and modulation recognition:

$$L^{(\mathrm{MTL})}(\theta_{shd}, \theta_p, \theta_m) = L_p^{(\mathrm{MTL})}(\theta_{shd}, \theta_p, \theta_m) + L_m^{(\mathrm{MTL})}(\theta_{shd}, \theta_p, \theta_m)$$

Optimization for MTL: The optimization process aims to find parameters (θ_{shd}^*, θ_p^*, θ_m^*) that minimize the combined MTL loss:

$$\theta^* = \arg \min_{\theta_{shd}, \theta_p, \theta_m} L^{(\mathrm{MTL})}(\theta_{shd}, \theta_p, \theta_m)$$

MTL's benefit lies in its ability to jointly optimize tasks, efficiently using shared information, and potentially improving the model's overall performance compared to optimizing tasks independently in STL.

4.3 Simulation and Evaluation

4.3.1 Datasets Condition

The dataset consists of 4,319,655 blocks, each containing two labels (class) in every *block* sample: a packet format and a modulation type label, as depicted in Fig. 4.12. The packet label includes Legacy, HT/VHT, HE, 5G, and No-Signal, while the modulation label encompasses BPSK, QPSK, 16QAM, and 64QAM. To simplify, HT and VHT labels is combined into the same class due to their identical data field format structure. The dataset is divided into 90% for training and 10% for validation. We employ the proposed USHWAP platform for model development, training, and testing. The model is trained using the Keras framework and TensorFlow on an Nvidia GeForce RTX 3090 GPU in the Python environment. The Adam optimizer is applied with a learning rate of $\alpha = 0.001$ for 100 epochs, employing a batch size of 100 for mini-batch learning, with a patience value of 15. The

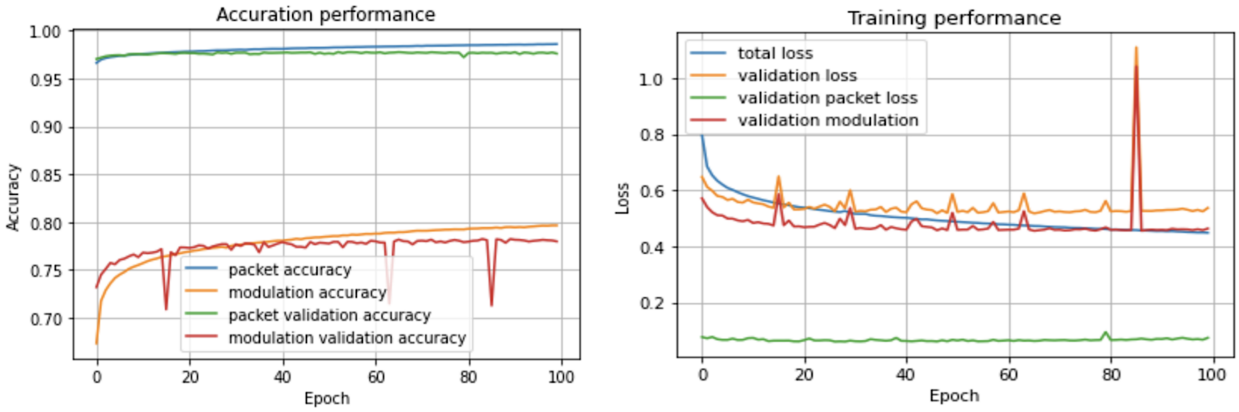


Figure 4.14: MTL-CNN training performance.

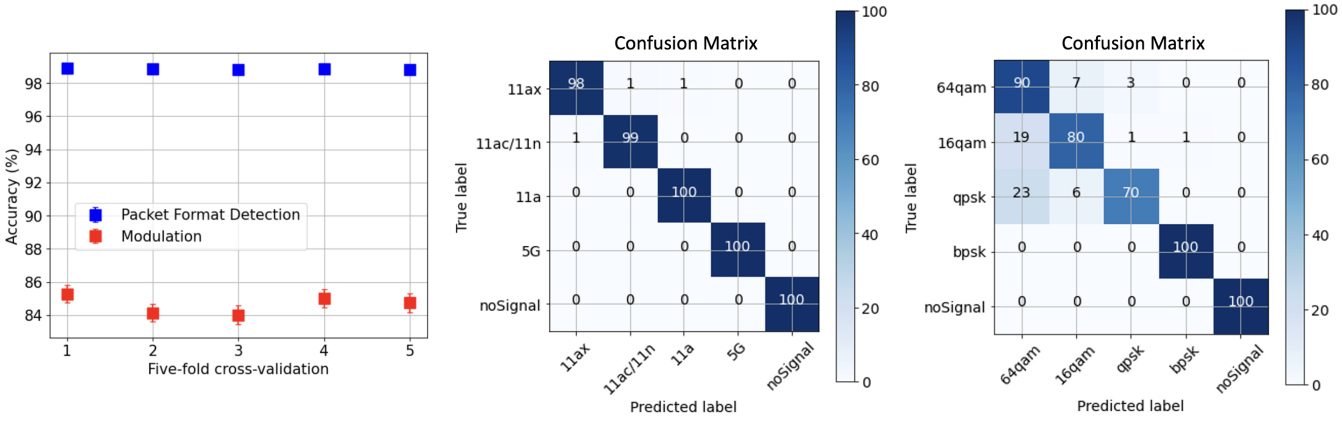
five-fold cross-validation is implemented with five iterations for training, and the uncertainty interval for each performance metric is calculated. The dataset is equally partitioned into five parts, with four parts used as training sets and the remaining part as the test set in each iteration.

4.3.2 Training Result

The MTL-CNN training performance is shown in Fig. 4.14. The five-fold cross-validation results for training and confusion matrix are presented in Fig. 4.15, showcasing the model accuracy over five iterations. As depicted in Fig. 4.15(a), the training results using five-fold cross-validation demonstrate consistent performance throughout the five training processes. The proposed MTL-CNN achieved an overall accuracy of 99% for packet format detection and 84.7% for signal modulation classification, as depicted in Fig. 4.15(b).

The normalized confusion matrix for packet format detection, ranging from SNR = 24 to 4 dB, is depicted in Figure 4.16, highlighting the MTL model's effectiveness in classifying each output class. The result demonstrates the robustness of the CNN model.

We verified the MTL-CNN model at various SNRs to demonstrate the ability of the proposed model to detect and classify the WLAN signal. Figure 4.17 shows the performance of MTL with CNN under SNR = 4 to 24 dB. The result shows that the proposed model performs well in classifying the WLAN signal, although the received power is small, and



(a) Five-fold cross-validation training result

(b). Performance accuracy: 99% for packet format detecion and 84.7% for modulation recognition

Figure 4.15: (a). Five-fold cross-training accuracy; (b). Confusion matrix for packet format detection and modulation recognition.

satisfies the CCA threshold for detecting presence signals.

The simulation of the proposed model concerning time offset changes showed that the MTL-CNN model is robust in the face of time offset variations. We generated WLAN signal data fields at SNR = 24 dB and subsequently shifted the signal to assess the model's performance. Fig. 4.18 illustrates the proposed model's performance concerning variable time offsets. The results indicate that time offsets have a minimal impact on the accuracy of the MTL-CNN. Specifically, the MTL-CNN packet format model maintains a stable accuracy of approximately 99% across various timing offsets, even up to 90%. Conversely, the accuracy of the CNN modulation assessment experiences a slight reduction when tested with a 40% time offset. Regarding overall accuracy, MTL-CNN models demonstrate commendable performance in the presence of time offset variations.

4.3.3 MTL-CNN Testing

1. Software-based Testing

To validate our models, we tested them using the 802.11ac packet, MCS = 6 of the WLAN signal generated by the Matlab waveform generator. Pulse signals with different indices

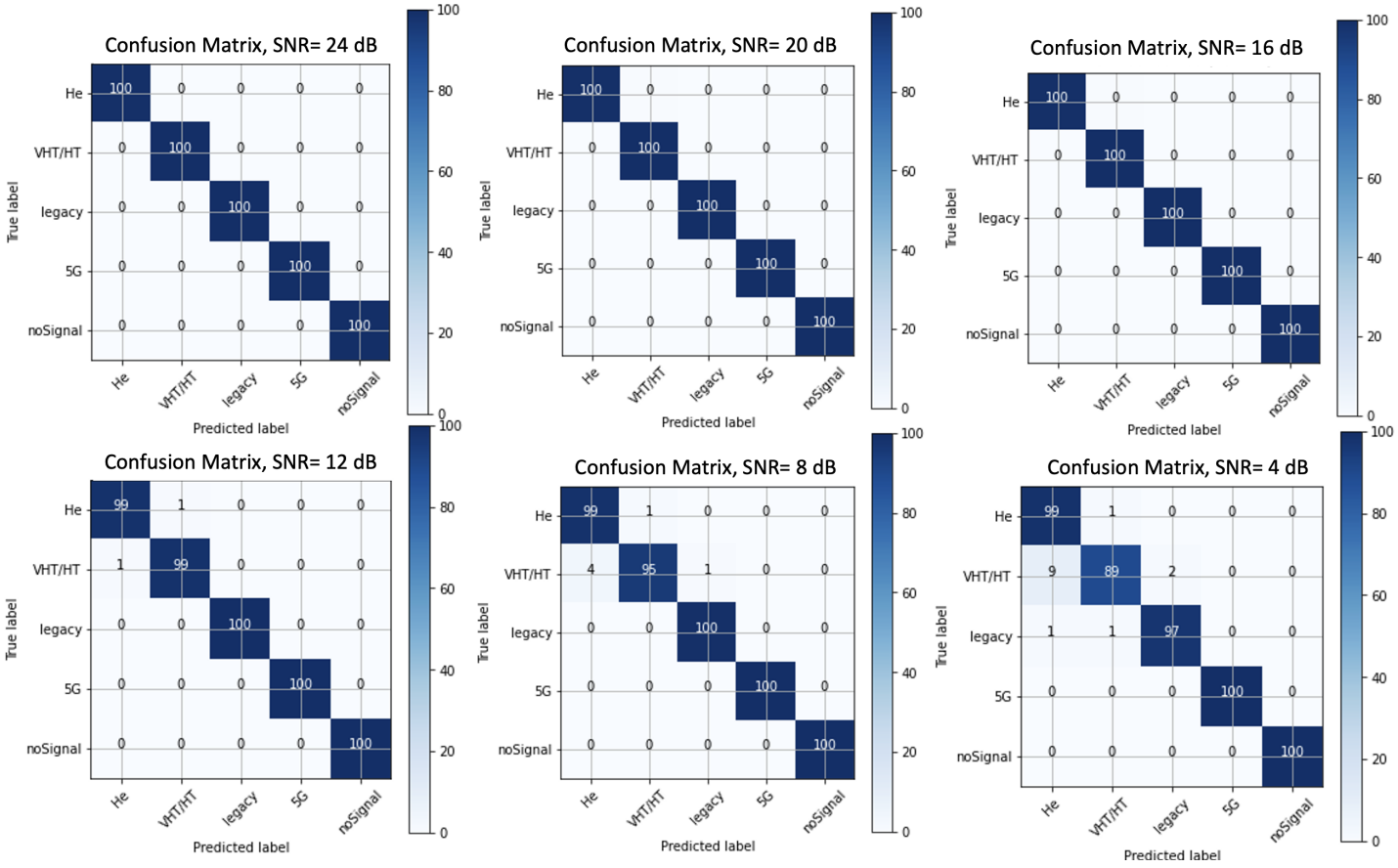


Figure 4.16: Confusion matrices of MTL-CNN for packet format detection across a range of SNRs from 24 to 4 dB.

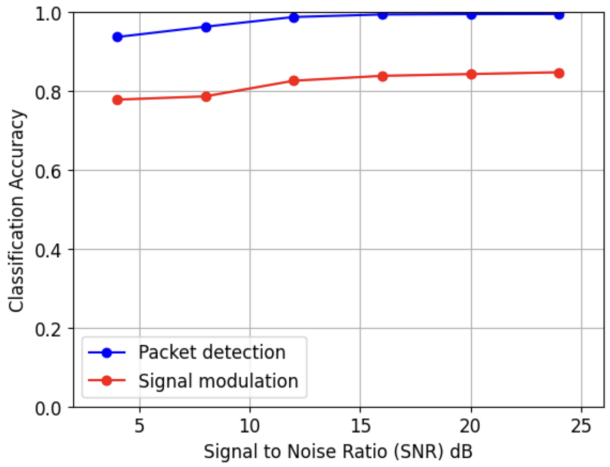


Figure 4.17: Classification performance for different SNR conditions.

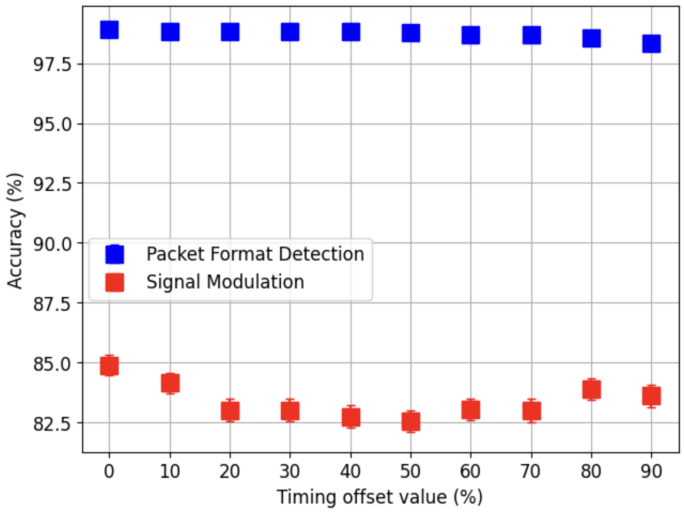


Figure 4.18: The MTL-CNN accuracy at SNR = 24 dB for different timing offset

depending on the classification class were used to represent the classification results. The experimental results show that the proposed model can accurately determine the packet format and modulation type of the WLAN packets. This result is shown in Fig. 4.19. The peaks in the display indicate the smallest misclassification. This visualization shows that MTL classification was done correctly using MTL-CNN.

2. Hardware Experimental Evaluation

After training and testing the MTL-CNN model in the simulation, we converted the trained model into its MATLAB equivalent. MATLAB also plays a crucial role in managing the FPGAs (Xilinx ZCU102) and SDRs (ADRV9009), facilitating actual transmission and reception. The experimental configuration is illustrated in Fig. 4.20. In this setup, one set of FPGA and SDR functions as the transmitter, sending signals to a receiver controlled by another set of FPGA and SDR. The transmitted signal alternates between 5G and VHT. With the aid of the trained CNN model, the receiver is expected to successfully identify the class of received signals, even in the absence of preamble information.

In the first scenario, the captured Wi-Fi signal (802.11ac or VHT) is altered and lacks preamble information. In the second scenario, the received 5G signals contain No-Signal. The proposed model demonstrates an ability to detect signals from various classes. While

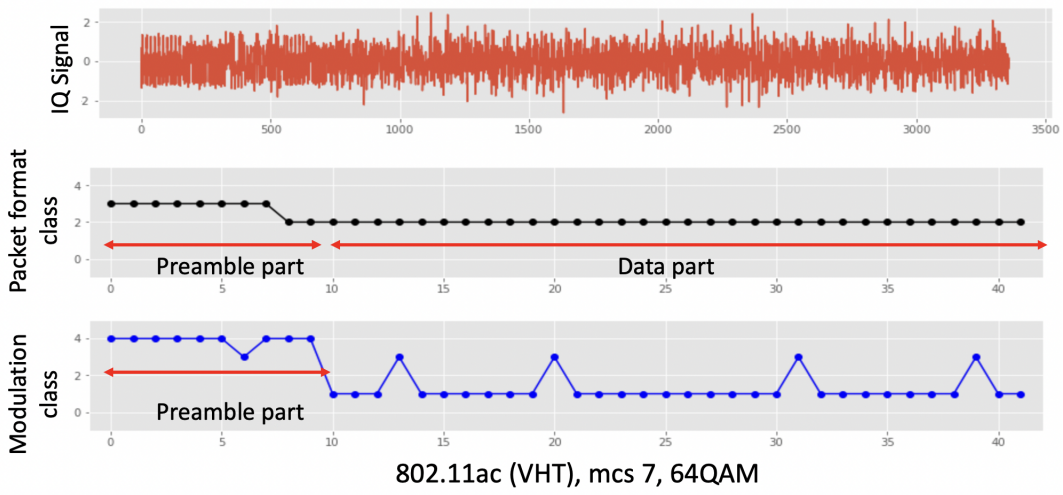


Figure 4.19: Testing for WLAN packet

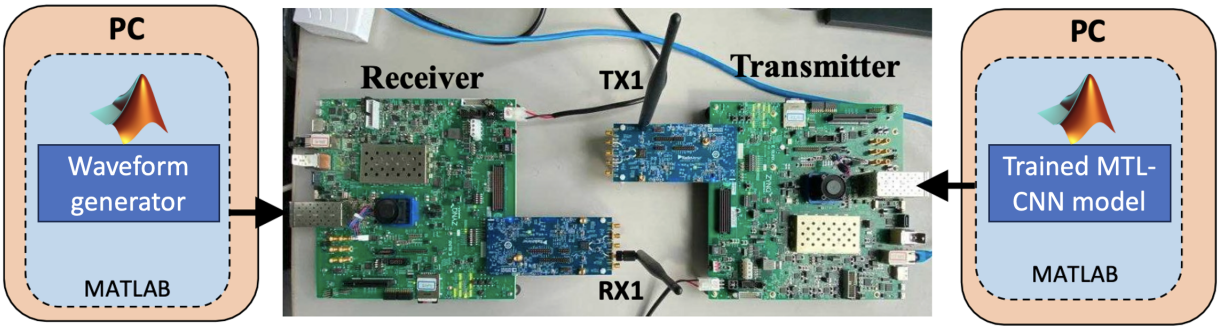


Figure 4.20: HW experiment setup.

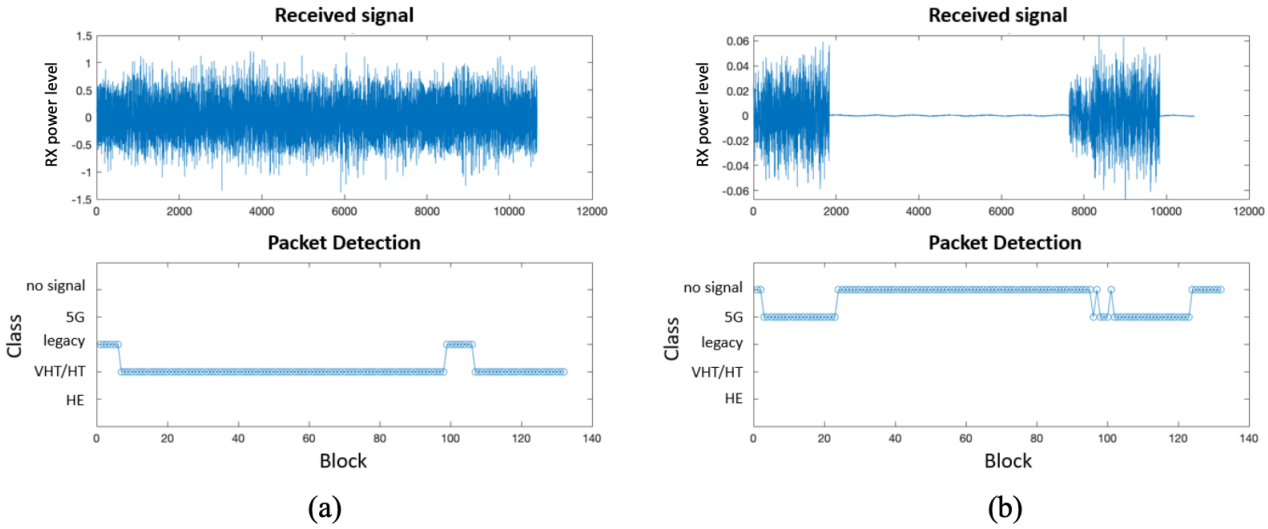


Figure 4.21: HW experiment results for (a) VHT packet, and (b) 5G packet.

there are instances where the MTL-CNN model struggles to identify the exact class of the signal (e.g., detecting a VHT packet as a legacy packet), it still accurately classifies the majority of the signal. Fig. 4.21 shows the experimental results.

4.4 Summary

This chapter introduces an MTL-CNN model designed for the dual tasks of packet detection and modulation recognition within WLAN packets. Our approach allows for the accurate identification of five distinct packet formats and four modulated signals. The proposed MTL-CNN model acts as a valuable complement, enhancing the sensitivity of existing CCA performance. Upon receiving the data field of a transmitting packet, MTL-CNN simultaneously recognizes packet formats (Legacy, HT/VHT, HE, and 5G NR) and modulation types (BPSK, QPSK, 16QAM, and 64QAM). The MTL-CNN model achieves an impressive 99% accuracy for packet detection and an 84.7% accuracy for signal modulation classification. The MTL-CNN packet format model maintains a consistent accuracy of approximately 99% even under various timing offsets, including up to 90%. Overall,

MTL-CNN models demonstrate commendable performance in handling time offset variations. Additionally, we conducted evaluations of the MTL-CNN model using signals from FPGA and SDR, specifically 802.11ac and 5G NR. The results confirm the MTL-CNN's capability to successfully distinguish between the actual signals of 802.11ac and 5G NR."

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This thesis proposes USHWAP, a Unified Software/Hardware Wireless-AI Platform designed to meet the increasing demand for an integrated environment that combines multi-SW simulation with real-time HW testing. Additionally, it introduces MTL-CNN for packet format detection and modulation recognition. The primary goal of this thesis is to present a cohesive SW/HW co-design platform specifically for the development of wireless AI applications. USHWAP empowers researchers to create and assess the effectiveness of their intelligent algorithms within wireless systems. This versatile platform accommodates multi-SW simulation preferences, such as MATLAB or Python, or a combination of both. It facilitates subsequent HW testing employing FPGA and SDR using a unified connectivity. USHWAP represents a significant contribution to the field, offering a unified solution for researchers to transition seamlessly from SW simulation to HW experimentation. USHWAP encompasses several key contributions:

1. Multi-application development: Enable SW simulation and HW implementation.
2. Multi-SW simulation: Supports simulation in both MATLAB and Python.
3. Generic and Flexible with Unified SW/HW Connectivity:
    - No need for special MATLAB toolboxes.

- Scalable HW for device computing and multiple devices.

4. High throughput and low latency: Supports rapid execution of multiple test scenarios.

The second topic discuss the development of MTL-CNN for packet format detection and modulation recognition of WLAN signals. The MTL-CNN model is implemented on USHWAP to showcase the effectiveness of the proposed platform for the developing and testing phase. This method complements the existing CCA method, which prevents packet collisions and improves the coexistence of 5G NR and WLAN. The MTL-CNN model enables simultaneous analysis of packets, efficiently identifying various packet formats (Legacy, HT/VHT, HE, and 5G) and modulation types (BPSK, QPSK, 16QAM, and 64QAM) without relying on the preamble part. Demonstrating high accuracy, the MTL-CNN achieves a remarkable 99% accuracy for packet detection and an 84.7% accuracy for modulation classification. This comprehensive and accurate approach showcases the effectiveness of the MTL-CNN model in handling the intricacies of wireless communication. The advantages of MTL-CNN including:

1. Benefits of MTL-CNN Over Single-Task Learning:
    - Ability to perform multiple tasks concurrently on a single network model
    - Efficiency improvement with lightweight neural network parameters
2. Benefits of MTL-CNN for Packet Detection:
    - Improved CCA Sensitivity
    - Complementary to Existing CCA Method

5.2 Future Works

While the proposed unified SW/HW wireless-AI platform shows promise, there is room for improvement in user experience, usability, and accessibility of the verification framework. To address this, an integration of a Graphical User Interface (GUI) is suggested in the future. This front-end interface will offer users an intuitive means to interact with the

co-working platform, reducing the potential for human errors often associated with script-based simulations. Introducing this feature aims to cater to a broader user base, including those who may need a deeper understanding of scripting languages. This GUI is anticipated to significantly elevate the overall user experience when carrying out verification tasks within the framework.

In the case of MTL-CNN, further research will be conducted to enhance its performance under non-Gaussian noise conditions. In the future, we plan to develop a new algorithm that integrates the MTL-CNN model with the existing CCA method to enhance the CCA's sensitivity in detecting the presence of WLAN packets and avoiding collisions for next-generation WLAN.

Acknowledgment

I am greatly indebted and would like to acknowledge the following individuals:

- Prof. Hiroshi Ochi for the guidance, supervision, and support during the doctoral course in Kyutech.
- Prof. Masayuki Kurosaki and Dr. Yuhei Nagao for providing assistance and giving insightful comments in finishing this work.
- Prof. Mario Koppenk, Prof. Tsuyoshi Okita, and Dr. Wahyul Amien Syafei, who act as members of the graduation committee, spend much time reading the manuscript and giving constructive comments.
- Fellow students, researchers, and staff: Harry, Taqi, Han, Ishiki, Tsuji, Nakashima, Murayama, Kobayashi-san, Hamada-san, Ogata-san, Urabe-san and many others with whom I was involved in many discussions and providing assistance in many ways during my stay in Japan.
- Fellow Indonesian students in Kyutech who provided me with a good balance with out-of-work life.
- To whom undoubtedly I am indebted most; my wife (Shelvi Ekariani), my son (Fathih and Athar), and my family for the never-ending support and the extra bit of motivation to finish the doctoral course.

Bibliography

[1] Manjrul, Z. Ahsan, and Siddiqu, "Machine-learning-based disease diagnosis: A comprehensive review," *Healthcare*, vol. 10, no. 3, 2022.

[2] L. Lanante, S. Roy, S. E. Carpenter, and S. Deronne, "Improved abstraction for clear channel assessment in ns-3 802.11 wlan model," in *Proceedings of the 2019 Workshop on ns-3*, Florence, Italy, June 2019, pp. 49–56.

[3] S. Szott, K. Kosek-Szott, P. Gawłowicz, J. T. Gómez, B. Bellalta, A. Zubow, and F. Dressler, "Wi-fi meets ml: A survey on improving ieee 802.11 performance with machine learning," *IEEE Communications Surveys and Tutorials*, vol. 24, no. 3, pp. 1843–1893, 2022.

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

[5] E. Khorov, A. Kiryanov, A. Lyakhov, and G. Bianchi, "A tutorial on ieee 802.11ax high efficiency wlans," *IEEE Communications Surveys and Tutorials*, vol. 21, no. 1, pp. 197–216, 2019.

[6] B. Bellalta, "Ieee 802.11ax: High-efficiency wlans," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 38–46, 2016.

[7] M. S. Afaqui, E. Garcia-Villegas, and E. Lopez-Aguilera, "Ieee 802.11ax: Challenges and requirements for future high efficiency wifi," *IEEE Wireless Communications*, vol. 24, no. 3, pp. 130–137, 2017.

[8] C. Deng, X. Fang, X. Han, X. Wang, L. Yan, R. He, Y. Long, and Y. Guo, "Ieee 802.11be wi-fi 7: New challenges and opportunities," *IEEE Communications Surveys and Tutorials*, vol. 22, no. 4, pp. 2136–2166, 2020.

[9] E. Khorov, I. Levitsky, and I. F. Akyildiz, "Current status and directions of ieee 802.11be, the future wi-fi 7," *IEEE Access*, vol. 8, pp. 88 664–88 688, 2020.

[10] A. Garcia-Rodriguez, D. López-Pérez, L. Galati-Giordano, and G. Geraci, "Ieee 802.11be: Wi-fi 7 strikes back," *IEEE Communications Magazine*, vol. 59, no. 4, pp. 102–108, 2021.

[11] T. Wang, S. Wang, and Z.-H. Zhou, "Machine learning for 5g and beyond: From model-based to data-driven mobile wireless networks," *China Communications*, vol. 16, no. 1, pp. 165–175, 2019.

[12] S. Han, I. Chih-Lin, G. Li, S. Wang, and Q. Sun, "Big data enabled mobile network design for 5g and beyond," *IEEE Communications Magazine*, vol. 55, no. 9, pp. 150–157, 2017.

[13] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in ofdm systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2018.

[14] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2554–2564, 2019.

[15] I. Ahmad, S. Shahabuddin, H. Malik, E. Harjula, T. Leppänen, L. Lovén, A. Anttonen, A. H. Sodhro, M. Mahtab Alam, M. Juntti, A. Ylä-Jääski, T. Sauter, A. Gurtov, M. Ylianttila, and J. Riekki, "Machine learning meets communication networks: Current trends and future challenges," *IEEE Access*, vol. 8, pp. 223 418–223 460, 2020.

[16] M. R. Mahmood, M. A. Matin, P. Sarigiannidis, and S. K. Goudos, "A comprehensive review on artificial intelligence/machine learning algorithms for empowering the future iot toward 6g era," *IEEE Access*, vol. 10, pp. 87 535–87 562, 2022.

[17] D. Ichwana Putra, M. H. B. Pratama, Y. Nagao, M. Kurosaki, and H. Ochi, "Multi-task learning with convolutional neural network for packet format detection and modulation classification of wireless LAN," in *International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju Island, Republic of Korea, October 2022, pp. 642–644.

[18] R. Saravanan and P. Sujatha, "A state of art techniques on machine learning algorithms: A perspective of supervised learning approaches in data classification," in *2018 Second International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2018, pp. 945–949.

[19] W. Yu, F. Sohrabi, and T. Jiang, "Role of deep learning in wireless communications," *IEEE BITS the Information Theory Magazine*, vol. 2, no. 2, pp. 56–72, 2022.

[20] S. Liu, T. Wang, and S. Wang, "Toward intelligent wireless communications: Deep learning - based physical layer technologies," *Digital Communications and Networks*, vol. 7, no. 4, pp. 589–597, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352864821000742

[21] Z. Qin, H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep learning in physical layer communications," *IEEE Wireless Communications*, vol. 26, no. 2, pp. 93–99, 2019.

[22] T. Wang and S. Wang, "Online convex optimization for efficient and robust inter-slice radio resource management," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 6050–6062, 2021.

[23] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[24] N. Sutisna, R. Hongyo, L. Lanante, Y. Nagao, M. Kurosaki, and H. Ochi, "Fast design exploration with unified hw/sw co-verification framework for high throughput wireless communication system," in *2016 International Conference on IC Design and Technology (ICICDT)*, 2016, pp. 1–4.

[25] N. Sutisna, L. Lanante, Y. Nagao, M. Kurosaki, and H. Ochi, "A unified hw/sw system-level simulation framework for next generation wireless system," in *2017 30th IEEE International System-on-Chip Conference (SOCC)*, 2017, pp. 322–327.

[26] N. Surantha, N. Sutisna, Y. Nagao, and H. Ochi, "Soc design with hw/sw co-design methodology for wireless communication system," in *2017 17th International Symposium on Communications and Information Technologies (ISCIT)*, 2017, pp. 1–6.

[27] Y. Huang, S. Liu, C. Zhang, X. You, and H. Wu, "True-data testbed for 5G/B5G intelligent network," *Intelligent and Converged Networks*, vol. 2, no. 2, pp. 133–149, 2021.

[28] N. Sutisna, L. Lanante, Y. Nagao, M. Kurosaki, and H. Ochi, "Unified hw/sw framework for efficient system level simulation," in *2016 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, 2016, pp. 518–521.

[29] Z. Jiang, S. Fu, S. Zhou, Z. Niu, S. Zhang, and S. Xu, "AI-assisted low information latency wireless networking," *IEEE Wireless Communications*, vol. 27, no. 1, pp. 108–115, 2020.

[30] A. Zappone, M. D Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Transactions on Communications*, vol. 67, no. 10, pp. 7331–7376, 2019.

[31] S. Ponnaluru and S. Penke, "A software-defined radio testbed for deep learning-based automatic modulation classification," *International Journal of Communication Systems*, vol. 33, no. 15, p. e4556, 2020.

[32] A. M. Ashleibta, A. Zahid, S. A. Shah, Q. H. Abbasi, and M. A. Imran, "Flexible and scalable software defined radio based testbed for large scale body movement," *Electronics*, vol. 9, no. 9, p. 1354, 2020.

[33] C. V. Nahum, L. De Nóvoa Martins Pinto, V. B. Tavares, P. Batista, S. Lins, N. Linder, and A. Klautau, "Testbed for 5G connected artificial intelligence on virtualized networks," *IEEE Access*, vol. 8, pp. 223 202–223 213, 2020.

[34] J. Wu, P. Du, Z. Zhang, and Q. Wang, "A software defined radio testbed for over-the-air cognitive cycle demonstration," in *IEEE 11th Sensor Array and Multichannel Signal Processing Workshop (SAM)*, Hangzhou, China, June 2020, pp. 1–4.

[35] Z. Lu, Z. Hu, Z. Han, L. Wang, R. Knopp, and Y. Zhang, "An artificial intelligence enabled F-RAN testbed," *IEEE Wireless Communications*, vol. 27, no. 2, pp. 65–71, 2020.

[36] M. S. Gawade and R. D. Joshi, "Gnu radio and usrp b210 based software defined radio for ofdm data transmission," in *IEEE International Conference on Mobile Networks and Wireless Communications (ICMNWC)*, Tumkur, India, December 2021, pp. 1–5.

[37] F. A. Toasa, L. Tello-Oquendo, C. R. Peńafiel-Ojeda, and G. Cuzco, "Experimental demonstration for indoor localization based on aoa of bluetooth 5.1 using software defined radio," in *IEEE 18th Annual Consumer Communications and Networking Conference (CCNC)*, Las Vegas, USA, March 2021, pp. 1–4.

[38] T. Nakashima, L. Lanante, M. Pratama, M. Kurosaki, and H. Ochi, "ns3-ai: Rate control for wireless lan by deep q-network," in *International Workshop on Smart Info-Media System in Asia (SISA2022)*, Online, September 2022.

[39] Cisco, "Visual networking index: Forecast and trends, 2017–2022," San Jose, CA, USA, Rep. C11-741490-00, 2018.

[40] S. Shaik and S. Kirthiga, "Automatic modulation classification using densenet," in *2021 5th International Conference on Computer, Communication and Signal Processing (ICCCSP)*, 2021, pp. 301–305.

[41] Yin, W. Zhendong, C. Zhilu, Z. Yunfei, and Nan, "A robust modulation classification method using convolutional neural networks," *EURASIP Journal on Advances in Signal Processing*, vol. 2019, no. 21, pp. 1687–6180, 2019.

[42] Kim, Z. Minjae, D. Zhongfeng, S. Kim, and Choi, "Deep-learning-based frame format detection for ieee 802.11 wireless local area networks," *Electronics*, vol. 9, no. 7, 2020.

[43] S. Hong, Y. Wang, Y. Pan, H. Gu, M. Liu, J. Yang, and G. Gui, "Convolutional neural network aided signal modulation recognition in ofdm systems," in *2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring)*, Antwerp, Belgium, 2020, pp. 1–5.

[44] X.-R. Jiang, H. Chen, Y.-D. Zhao, and W.-Q. Wang, "Automatic modulation recognition based on mixed-type features," *International Journal of Electronics*, vol. 108, no. 1, pp. 105–114, Jan 2021.

[45] K. Liu, W. Gao, and Q. Huang, "Automatic Modulation Recognition Based on a DCN-BiLSTM Network," *Sensors*, vol. 21, no. 5, p. 1577, Feb 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/5/1577

[46] M. Kim, Z. Zhang, D. Kim, and S. Choi, "Deep-Learning-Based Frame Format Detection for IEEE 802.11 Wireless Local Area Networks," *Electronics*, vol. 9, no. 7, p. 1170, July 2020. [Online]. Available: https://www.mdpi.com/2079-9292/9/7/1170

[47] Y. Zhang and Q. Yang, "A survey on multi-task learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 34, no. 12, pp. 5586–5609, 2022.

[48] D. I. Putra, M. H. B. Pratama, L. Lanante, and H. O. Grad, "Packet format detection and modulation classification of wireless lan using distributed convolutional neural network," in *2021 International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS)*, Hualien City, Taiwan, November 2021, pp. 1–2.

[49] E. Perahia and R. Stacey, *Next Generation Wireless LANs: 802.11n and 802.11ac*, 2nd ed. Cambridge University Press, 2013.

[50] S. Hong, Y. Zhang, Y. Wang, H. Gu, G. Gui, and H. Sari, "Deep learning-based signal modulation identification in ofdm systems," *IEEE Access*, vol. 7, pp. 114 631–114 638, 2019.

[51] H. Ye, G. Y. Li, and B. Juang, "Power of deep learning for channel estimation and signal detection in ofdm systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2018.

[52] A. Jagannath and J. Jagannath, "Multi-task learning approach for modulation and wireless signal classification for 5g and beyond: Edge deployment via model compression," *Physical Communication*, vol. 54, p. 101793, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1874490722001100

[53] Y. Wang, G. Gui, T. Ohtsuki, and F. Adachi, "Multi-Task Learning for Generalized Automatic Modulation Classification Under Non-Gaussian Noise With Varying SNR Conditions," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3587–3596, Jun 2021.

Publication List

Journals

1. Dody Ichwana Putra, Muhammad Harry Bintang Pratama, Ryotaro Isshiki, Yuhei Nagao, Leonardo Lanante Jr, and Hiroshi Ochi, *"A Unified Software and Hardware Platform for Machine Learning Aided Wireless Systems"*, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Science, Vol. E106-A, No.10, Aug. 2023. DOI:10.1587/transfun.2023SDP0006

International Conferences

1. Dody Ichwana Putra, Harry Bintang Pratama, Tomoki Nakashima, Yuhei Nagao, Masayuki Kurosaki and Hiroshi Ochi, *"Multi-Task Learning with Convolutional Neural Network Approach for Packet Collision Avoidance in 802.11 WLAN"*, IEEE 9th NAFOSTED Conference on Information and Computer Science (NICS 2022),Ho Chi Minh City, Vietnam, pp. 305-310, Oct 2022.
   DOI:10.1109/NICS56915.2022.10013420

2. Dody Ichwana Putra, Harry Bintang Pratama, Yuhei Nagao, Masayuki Kurosaki and Hiroshi Ochi, *"Multi-Task Learning with Convolutional Neural Network for Packet Format Detection and Modulation Classification of WLAN "*, IEEE 13th International Conference on International and Communication Technology Convergence (ICTC 2022), Jeju Island, Republic of Korea, pp. 642-644, Nov 2022.
   DOI:10.1109/ICTC55196.2022.9952462

3. Dody Ichwana Putra, Harry Bintang Pratama, Leonardo Lanante, and Hiroshi Ochi, *"Packet Format Detection and Modulation Classification of Wireless LAN Using Distributed Convolutional Neural Network "*, IEEE International Symposium on Intelligent Signal Processing and Communication Systems (ISPACS, 2021), Hualien City, Taiwan, pp. 1-2, Dec 2021. DOI: 10.1109/ISPACS51563.2021.9651091

4. Harry Bintang Pratama, Dody Ichwana Putra, Leonardo Lanante, and Hiroshi Ochi, *"Convolutional Neural Network for Asynchronous Packet Parameter Identification in Dense Wi-Fi"*, IEEE International Conference on Communications Workshops (ICC Workshops, 2021), Montreal, QC, Canada, pp. 1-6, June 2021. DOI: 10.1109/ICCWorkshops50388.2021.9473621

Seminars

1. Harry Bintang Pratama, Dody Ichwana Putra, Muhammad Taqiyuddin, Masayuki Kurosaki and Hiroshi Ochi, *"Experimental Evaluation of CNN-based Packet Detection to Ensure 5G and Wi-Fi Network Coexistance"*, 9th International Forum on Advanced Technologies, 2023, Taipe, Taiwan.