January 2021

# Exploration Of Deep Learning Models For Maximum Throughput To Hybrid Beamforming And Compressive Channel Estimation

Youness Arjoune

# EXPLORATION OF DEEP LEARNING MODELS FOR MAXIMUM THROUGHPUT TO HYBRID BEAMFORMING AND COMPRESSIVE CHANNEL ESTIMATION

Youness Arjoune

Dissertation submitted to the Faculty of the

University of North Dakota

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Electrical Engineering

December 9, 2021

Grand Forks, North Dakota

Name: Youness Arjoune

Degree: Doctor of Philosophy

This document, submitted in partial fulfillment of the requirements for the degree from the University of North Dakota, has been read by the Faculty Advisory Committee under whom the work has been done and is hereby approved.

DocuSigned by:
Saleh Faruque
6D0F996935C5443...
Saleh Faruque

DocuSigned by:
Prakash Ranganathan
2E9D28B343204C6...
Prakash Ranganathan

DocuSigned by:
Ryan Adams
B19AB8472D964A9...
Ryan Adams

DocuSigned by:
Dr. Kouhyar Tavakolian
A24D898B29D94B8...
Kouhyar Tavakolian

DocuSigned by:
William Semke
228665DA08054F8...
William Semke

This document is being submitted by the appointed advisory committee as having met all the requirements of the School of Graduate Studies at the University of North Dakota and is hereby approved.

DocuSigned by:
Chris Nelson
2E6AF088C733403...
Chris Nelson
Dean of the School of Graduate Studies

11/23/2021
Date

# Permission

Title: Exploration of Deep Learning Models for Maximum Throughput to Hybrid Beamforming and Compressive Channel Estimation

Department: Electrical Engineering
Degree: Doctor of Philosophy

In presenting this dissertation in partial fulfillment of the requirements for a graduate degree from the University of North Dakota, I agree that the library of this University shall make it freely available for inspection. I further agree that permission for extensive copying for scholarly purposes may be granted by the professor who supervised my dissertation work or, in his absence, by the chairperson of the department or the dean of the Graduate School. It is understood that any copying or publication or other use of this thesis or part thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of North Dakota in any scholarly use which may be made of any material in my dissertation.

Youness Arjoune 12/09/2021

Youness Arjoune

# ABSTRACT

The overall objective of this dissertation is to improve the capacity of 5G and Beyond systems by improving spectral efficiency, efficiently managing interference, and addressing the radio spectrum's poor management while enhancing cost, energy, and computation efficiencies. The first contribution of this dissertation consists of designing effective hybrid beamforming solutions using the theory of deep reinforcement learning. The results demonstrate that deep reinforcement learning achieves near-optimal spectral efficiency and provides autonomous decision-making that can learn from interacting with the wireless environment. This method enhances hybrid beamforming computational efficiency, hardware efficiency, and energy efficiency. These methods are also desirable in scenarios where channel conditions change too fast, and we may not have existing channel datasets or the corresponding optimal beamforming solutions required for supervised learning. The second contribution of this dissertation is channel estimation in a hybrid architecture. Our numerical results demonstrated that compressive sensing could be leveraged to estimate the channel in hybrid architecture from a few training samples. The sensing matrix can be optimized, and the high dimensional channel can also be estimated using very few training samples. Our results revealed that orthogonal matching pursuit enables compressive channel estimation in low signal-to-noise ratio settings. We explored deep learning and deep reinforcement learning methods for channel estimation to design an end-to-end channel estimation where the model can map the training pilots directly to the channel estimate. The third contribution of this dissertation discusses the efficient design of jamming detection and mitigation algorithms. We developed jamming detection methods based on boosting decision trees. Our results indicated that boosting techniques achieve adequate detection performance and can

be trained in a short time. The last contribution of this dissertation consists of developing narrow band and wideband spectrum sensing techniques based on machine learning and compressive sensing, respectively. We demonstrated that Bayesian compressive sensing with a Toeplitz measurement matrix could sense the activity of the primary users with high probabilities of detection while reducing the number of acquired samples, resulting in a reduction of the sensing time and complexity of the algorithm.

(Keywords)

Machine Learning, Deep learning, Deep Reinforcement Learning, Compressive Sensing, Massive MIMO, mmWave, 5G, Jamming, Hybrid Beamforming, Spectrum Sharing.

# Dedication

*To my parents.*

# Acknowledgments

My deepest thanks go to my advisor, Professor Saleh Faruque, for his unwavering support. This dissertation would not have been possible without his dedication and contributions. The support and encouragement he provided helped me complete my Ph.D. I admire his dedication to science, his curiosity, and his unrelenting pursuit of the highest standards in wireless communication research. I want to express my deepest gratitude to the second readers of this dissertation, Dr. Prakash, Dr. Ryan Adams, Dr. Kouhyar Tavakolian, and Dr. William Semke, for accepting the invitation to serve on my committee. Dr. Prakash always provided me with his strong support. I admire his commitment to high expectations, guidance, and support during my Ph.D. program and involving me in different research projects. Dr. Ryan Adams made numerous suggestions that shaped the outline of my dissertation. I would like to thank him for his assistance. I am equally indebted to Dr. Kouhyar Tavakolian for teaching me machine learning and introducing me to this exciting research area. I thank him for his tireless effort in providing me with his strong support and guidance.

I am indebted to the Fulbright program and the Moroccan-American Commission for Educational & Cultural Exchange (MACECE) to make the dream of studying in the US come true. I am grateful to Dr. Naima Kaabouch and Dr. Hassan El Ghazi for recommending me for this program as well as assisting me during the application process and throughout the program. I wish to thank the School of Graduate Studies at the University of North Dakota for granting me the Summer Dissertation Completion Fellowship Award. I also acknowledge the support of the School of Electrical Engineering and Computer Science for supporting me during my Ph.D. program. Some parts of this dissertation were supported by NSF grant award number #1443861. I want to thank Dr. Naima Kaabouch (Principal

Investigator) for hiring me as a graduate research assistant to work on this project. I also wish to thank the School of Electrical Engineering and Computer Science (SEECS) for supporting me throughout my Ph.D. program. I thank the UND writing center and its members for their careful proofreading and for providing many valuable comments and suggestions for improving the quality of writing. I thank them for spending hours reading and providing feedback on specific sections of this dissertation. I want to thank also Dr. Anna Crowell, technical Writer/Editor CEM, for providing many valuable comments on chapter 3.

I want to thank my colleagues and my co-authors Dr. Ahmed Tamtaoui, Dr. Hassan El Ghazi, Zakaria El Mrabet, Dr. Mohsen Riahi Manesh, Dr. Fatima Salahedine and Dr. Lahoucine Balihi. I also wish to thank Dr. Ahmet. M. Elbir for his feedback for making this manuscript, and Dr. Foad Sohrabi for making it possible to reproduce some of his figures on this dissertation. I want to thank also Dr. Ahmed Alkhateeb for sharing some parts of his code on channel estimation.

Last but not least, I would like to thank my family and my friends for their love and support. Particularly, my family members, my brothers, and my sister, who were my first and main supporters, and my friends, who helped me overcome the many challenges encountered.

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

3GPP Third-Generation Partnership Project

4G Fourth generation

ADC Analog-to-digital converter

AI Artificial Intelligence

AltMin Altering minimization

AoA Angle-of-Arrival

AoD Angle-of-Departure

BCH Broadcast channel

BER Bit Error Rate

BF Beamforming

CDL Clustered delay line

CNN Convolutional neural network

CR Cognitive Radio

CS Compressed Sensing also known as Compressive Sampling (CS)

dB Decibel

DDPG Deep Deterministic Policy Gradient

DL Downlink

DL-SCH Downlink shared channel

DQN Deep Q-Network

DQN Double Deep Q-Network

$F_{BB}$ Digital baseband precoder

$F_{RF}$ Analog precoder

FDD Frequency-Division Duplexing

KKT Secondary synchronization signals

Massive MIMO massive multiple-input multiple-output

MIMO Multiple-input multiple-output

MMSE Minimum Mean Square Error

mmWave Milimeter-wave

MO Manifold Optimization

MO-AltMin Manifold optimization altering minimization

MSE Mean Square Error

MU-MIMO Multiple-User MIMO

OFDM Orthogonal Frequency-Division Multiplexing

OMP Orthogonal Matching Pursuit

OMP Orthogonal Matching pursuit

PBCH Physical broadcast channel

PDCCH Physical downlink control channel

PDSCH Physical downlink shared channel

PE-AltMin altering minimization

PSS Primary synchronization signals

PU Primary user

QAM Quadrature Amplitude Modulation

RF Radio Frequency

RF Random forest

RFC Radio Frequency Chain

RL Reinforcement Learning

SAC Soft actor critic

SDMA Space-Division Multiple Access

SNIR Signal-to-noise Interference ratio

SNR Signal-to-noise Ratio

SS Spectrum sensing

SSDF Spectrum sensing data falsification

SSS Secondary synchronization signals

SU Secondary user

SU-MIMO Single-User MIMO

SVD Singular Value Decomposition

SVD singular-vector decomposition

SVM Support vector machine

TD-3 Twin-Delayed Deep Deterministic Policy Gradient

TDD Time-Division Duplexing

TDL Tapped delay line

UL Uplink

ULA Uniform Linear Array

UPA Uniform Planar Array

$W_{BB}$ Digital baseband precoder

$W_{RF}$ Analog combiner

# Chapter 1

# Introduction

Wireless communication has seen unprecedented growth over the past years. Wireless resources demand increases drastically as the number of wireless devices connected increases. Some studies expect about 29.3 billion connected devices by 2023 against 8.0 billion in 2016. In addition, 60% of the wireless traffic content is in $4K/8K$ video format. Furthermore, next-generation wireless communication is a platform for innovation that supports autonomous driving, virtual reality, and augmented reality, to name a few. Most of these technologies require not only a high data rate and low latency but also high reliability. In addition, the fifth generation and the evolution to 6G support many industrial applications such as automation and control combined with machine learning to improve the productivity of many businesses. This initiated the $1000\times$ capacity challenge. The capacity generally depends on three main factors: bandwidth, spectral efficiency, and the number of simultaneous connections. Network densification, millimeter-wave (mmWave), massive multiple-input multiple-output (MIMO), beamforming (BF), and cognitive radio (CR) are the big five technologies that are expected to define the vision of future generation wireless communications physical layer and reach the capacity challenge. Furthermore, future generation wireless communications should be built on several key pillars: uniform coverage, green communications, massive connection, high-level of reliability and security, and cognitive networks.

mmWave bands are attractive because they provide larger bandwidth compared with sub-6 GHz [1, 4, 5]. The communication over these bands, however, is challenging as the receiver

aperture is much smaller [2]. Also, the noise power level increases with the bandwidth, therefore lowering the signal-to-noise ratio. In addition, mmWave signals are susceptible to blockages. According to some studies, the human body can cause approximately 40 $dB$ loss that lasts for 0.5 seconds [3]. Compared with sub 6-GHz communications, there is an extra path loss of approximately 20 dB [9].

Massive MIMO technology mitigates the extra path loss in mmWave bands. The small wavelength in mmWave enables patching more antennas in a small area. The large-scale antenna provides higher antenna gain, therefore compensating for the mmWave substantial path loss. Massive MIMO enables beamforming as it entails highly directional and steerable beams. Conventional beamforming is often performed at the digital baseband, known as fully digital beamforming. It requires a dedicated radio frequency chain per each antenna element. Fully digital beamforming techniques are costly and power-hungry when used with massive MIMO [10]. One way of reducing this cost and energy consumption is reducing the number of radio frequency chains because analog beamforming uses only one RF chain, and a network of phase-shifters controls the beams. A phase shifter network enables antenna element phase control, making it possible to mitigate interference and increase the signal-to-noise ratio. Analog beamforming reduces the cost and hardware complexity. However, it does not support space-division multiple access, which is one of the main benefits of MIMO systems.

Hybrid beamforming mitigates the shortcomings of both analog and digital beamforming and provides a trade-off between performance, cost, and power consumption. Hybrid beamforming uses a small number of radio frequency chains, supporting multi-stream transmission and limiting energy consumption and hardware cost. Hybrid beamforming requires channel knowledge as a first step and solving the optimal hybrid precoding and combining matrices as a second step. Hybrid beamforming is a non-convex problem; the solution can be found

but proving its optimality is difficult because the series of phase shifters impose a unit modulus constraint. Enabling low-cost hybrid beamforming enhances interference management, given that interference is one of the primary sources of performance degradation in wireless networks. Pencil-like beams can also assist in building robust communications against attacks that use interference to create denial-of-service attacks.

Estimating the mmWave channel is also challenging because of the limited number of RFCs compared with the number of antennas in a hybrid architecture. The number of antennas before the analog combiner is large compared to the number of ADCs. Given that we can only access the output of 4 to 6 ADCs, recovering the high dimensional channel is challenging due to the few measurements. Traditional channel estimation methods used with the MIMO systems are therefore not applicable. This problem becomes even more challenging if we consider practical constraints such as the phase shifter finite quantized resolution. Fortunately, mmWave channels are low rank and can be assumed to be sparse in the angular and time domains– there are only a few dominant propagation paths that contribute to the overall channel– [4, 5]. This sparsity assumption aligns with the field measurements conducted by Nokia Bell lab at the band 28GHz. Therefore one can exploit the sparsity to formulate channel estimation in hybrid beamforming as a sparse recovery problem and leverage the theory of compressive sensing to design low-cost estimation methods.

Given the favorable propagation environment of sub-6GHz, these bands have been used for many types of communication, such as GPS, WiMAX, Wi-Fi, 3G, 4G, S- and C-band satellite communications, and currently, sub-6GHz are being investigated for 5G and beyond. The use of sub-6GHz presents many advantages, mainly when used with massive MIMO as linear beamforming techniques can be used, in lieu of complicated dirty paper coding used with conventional MIMO systems to cancel interference to achieve near-optimal performance. The use of sub-6GHz, which is already crowded, for 5G systems necessitates

spectrum sharing. The spectrum sharing techniques have many advantages, such as spectrum wastage minimization and efficient use of the available frequency spectrum. Cognitive radio is one kind of spectrum sharing that can help in creating cognitive technology and improve spectrum access. However, the spectrum sensing function of cognitive radio is not reliable, and developing robust sensing mechanisms is still an open issue.

## 1.1 Research Questions and objectives

The design of hybrid beamforming systems in mmWave massive MIMO can be challenging because of the limited number of RF chains that makes hybrid beamforming non-convex and turns channel estimation into a difficult task. This document seeks to answer the following research questions: 1) Can hybrid beamforming architecture achieve the performance of fully digital beamforming for the case when the number of RFCs is less than twice the number of data streams and considering practical constraints such as power and quantization of the phase shifters? If yes, under which conditions, 2) How to deal with non-convex constraint imposed by the unit modulus imposed by the analog precoder when the number of RF chains is equal to the number of users while alleviating the burden of a huge computation? 3) How to deal with mmWave channel estimation difficulties in hybrid systems and recover the mmWave channel from a few measurements (small number of antenna elements)? 4) How should the measurement be performed? 5) How many measurements are enough to reconstruct the mmWave channel? 6) How to deal with a low signal-to-noise ratio during the beam training phase? 7) How to efficiently use the radio spectrum? 8) How to deal with low detection performance sensing techniques to enable efficient spectrum sharing?

Therefore, in this research project, we aim to enable future cognitive generation wireless communications using the theory of machine learning and compressive sensing techniques

to maximize the spectral efficiency while reducing the hardware cost, energy consumption, computational complexity, efficiently managing the radio spectrum, and enhancing the reliability.

## 1.2 Motivation and Contributions

We group the state-of-the-art hybrid beamforming techniques into two main categories: traditional techniques based category including "OMP" [11] and its variants "SOMP" as well as "KKT", "MO" [12], and MMSE [13] algorithms and supervised learning based category [14, 15, 16, 17, 18]. These techniques, which we discuss in more detail in the next chapter, solve hybrid beamforming under unrealistic assumptions such as mmWave channel perfect knowledge or the availability of the huge datasets labeled. Furthermore, only a few attempts have been made to explore the theory of deep reinforcement learning, which has great success in dealing with non-convex large-scale problems and presents many advantages over supervised learning approaches. In addition, most of the existing channel estimation in hybrid beamforming rely on the sparsity of the channel in the angular domain and developed techniques to estimate channel from a few measurements. The choice of the recovery and measurement technique as well as the number of measurements play a significant role in the performance of these techniques but have not been investigated before. Therefore, we need more research in this area to further reduce the hardware and power cost, as well as the computational complexity [19]. Furthermore, the state-of-the-art-spectrum sensing techniques are not reliable. For instance, energy detection is simple but very sensitive to noise; autocorrelation-based sensing techniques are complex; and matched filter-based methods require prior knowledge about the primary user signal. We provide more detail about the limitations of these techniques in the following chapters. Therefore, it is necessary to develop

a reliable spectrum sensing function to enable spectrum sharing and efficiently use the radio spectrum. Last but not least, interference detection and management is an open problem, as wireless communications are vulnerable to jamming attacks, which use interference to create denial-of-service.

This dissertation provides four significant contributions. The first contribution consists of deep reinforcement learning methods-based hybrid beamforming. Specifically, we proposed two methods to determine hybrid precoding and combining matrices in hybrid beamforming using advanced techniques in continuous action space based on the soft actor-critic method and twin-delayed deep deterministic policy gradient (TD-3). Also, we developed a method for discrete action space based on double deep Q-learning for overcoming the shortcomings of Q-learning-based hybrid beamforming methods, such as the substantial overestimation and the convergence to poor policies. The second contribution of this dissertation addresses compressive sensing-based channel estimation and beam training in hybrid beamforming massive MIMO. Exploiting the sparsity of the mmWave channel, we proposed a compressive sensing formulation for channel estimation and beam training in hybrid beamforming, which can be solved using convex and relaxation techniques, greedy techniques, and Bayesian compressive sensing techniques. Then, we proposed a performance comparison between two techniques from each category, and we showed that greedy and Bayesian methods provide a balance between performance and speed, while convex and relaxation techniques have the lowest recovery error but at the cost of high complexity. In the second part of this contribution, we proposed a performance comparison between different measurement matrices that belong to different categories, and we showed that random measurement matrices achieve adequate performance but at high computation cost while deterministic sensing matrix can achieve comparable performance to random measurement matrices while optimizing the sensing power. These results are used to design beam training and channel estimation

methods for hybrid beamforming techniques. In addition, we proposed deep learning and deep reinforcement learning techniques to estimate frequency selective channels from a few training samples. These methods are beneficial for the case where there is no prior knowledge on the sparsity of the mmWave channel. The next contribution focuses on interference detection and mitigation methods for robust communications. First, we investigated the vulnerabilities of 5G systems to jamming attacks. Second, we developed machine learning-based jamming detection techniques. We built a sizeable comprehensive dataset, and we used it to train boosting techniques, ensemble methods, and grid search. We showed that offline decision trees could detect jamming with 90% while online decision trees can achieve 83% accuracy. To improve the resilience of wireless communication systems to interference, we proposed a modified code-division multiple access technique; we increased the capacity and resilience to interference. This technique can correct 50% more errors compared with traditional CDMA techniques. The last contribution of dissertation deals with machine learning and compressive sensing-based spectrum sensing techniques to improve the reliability of the sensing function of a cognitive radio network. Machine learning and Bayesian compressive sensing methods are investigated for narrow-band spectrum sensing and wideband spectrum sensing, respectively. We showed that sensing could be sped up with 50% compared with traditional Nyquist-based wideband sensing techniques and achieved 90% plus detection accuracy. In addition, we proposed a spectrum sensing data falsification detection method based on a deep neural network to enhance the robustness of cooperative spectrum sensing.

## 1.3 Outline

The rest of the text is organized as follows: Chapter 2 presents background on mmWave communication, beamforming, and literature review as well as the motivation of this work.

This chapter establishes the context and the necessary concept required to introduce the issues of mmWave Massive MIMO hybrid beamforming. Chapter 3 deals with the hybrid beamforming system model and the formulation of the problem as well as its solution using advanced deep reinforcement learning techniques. Chapter 4 deals with compressed sensing techniques for mmWave and presents the formulation of the channel estimation in hybrid beamforming using the sparsity of the mmWave bands. Performance comparison between several sparse recovery techniques is investigated in this chapter as well. Chapter 5 explores deep learning for frequency selective channel estimation. Chapter 6 proposes a real-time jamming detection method based on the theory of machine learning based on decision trees. It presents a method for improving system capacity and robustness to interference. Chapter 7 deals with spectrum sharing for sub-6GHz enabled future wireless communications. It describes sensing techniques using machine learning and sub-Nyquist. Chapter 8 draws conclusions and points to future research directions.

## 1.4 Publications

The research on this project resulted in the publications of several papers:

**Peer Reviewed Journal Papers:**

1. *Arjoune, Youness* and Faruque, Saleh. Experience-Driven Efficient Learning-Based Intelligent Hybrid Beamforming for Massive MIMO mmWave Communication Systems. *Journal of Physical Communications*, Elsevier, In Press, November 2021.

2. *Arjoune, Youness* and Faruque, Saleh. Real-Time DOA Estimation for Adaptive Massive MIMO Beamforming Based on Light GBM and Compressed Learning *submitted to International Journal of Communication Systems*, 2021.

3. *Arjoune, Youness* and Kaabouch, Naima. A comprehensive survey on spectrum sensing in cognitive radio networks: Recent advances, new challenges, and future research directions. *Journal of Sensors*, Multidisciplinary Digital Publishing Institute ,19:126, 2019. doi: https://doi.org/10.3390/s19010126

4. Arjoune, Youness and Kaabouch, Naima and El Ghazi, Hassan and Tamtaoui, Ahmed. A performance comparison of measurement matrices in compressive sensing. In: *International Journal of Communication Systems*, 31, 10, e3576, 2018. doi: https://doi.org/10.1002/dac.3576

5. Arjoune, Youness, Kaabouch, Naima. Wideband Spectrum Sensing: A Bayesian Compressive Sensing Approach. In: *Journal of Sensors*, 18, no. 6: 1839,2018. doi: https://doi.org/10.3390/s18061839

6. Arjoune, Youness, El Mrabet, Zakaria, and Kaabouch, Naima. Multi-Attributes, Utility-Based, Channel Quality Ranking Mechanism for Cognitive Radio Networks. In: *Appl. Sci.*, 2018, 8, 628. doi: https://doi.org/10.3390/app8040628

7. *Arjoune, Youness*, Kaabouch, Naima, El Ghazi, Hassan, and Tamtaoui, Ahmed. Compressive sensing: Performance comparison of sparse recovery algorithms. In: *2017 IEEE 7th annual computing and communication workshop and conference (CCWC)*, Monterey, Las Vegas, USA, 2017. doi: 10.1109/CCWC.2017.7868430

**Peer Reviewed Proceedings Papers:**

1. Arjoune, Y.; Faruque, S. Real-time Machine Learning Based on Hoeffding Decision Trees for Jamming Detection in 5G New Radio. IEEE Big Data Conference 2020, December 2020. doi: 10.1109/BigData50022.2020.9377912

2. Y. Arjoune, Z. E. Mrabet, H. E. Ghazi and A. Tamtaoui, "Spectrum sensing: Enhanced energy detection technique based on noise measurement," 2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, 2018, pp. 828-834, doi: 10.1109/CCWC.2018.8301619

3. Y. Arjoune, F. Salahdine, M. S. Islam, E. Ghribi and N. Kaabouch. A Novel Jamming Attacks Detection Approach Based on Machine Learning for Wireless Communication. In: *2020 International Conference on Information Networking (ICOIN)*, Barcelona, Spain, 2020, pp. 459-464, doi: 10.1109/ICOIN48656.2020.9016462

4. Y. Arjoune and S. Faruque. Smart Jamming Attacks in 5G New Radio: A Review. 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2020, pp. 1010-1015, doi:10.1109/CCWC47524.2020.9031175.

5. Y. Arjoune and S. Faruque. Artificial Intelligence for 5G Wireless Systems: Opportunities, Challenges, and Future Research Direction. 10th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2020, pp. 1023-1028, doi: 10.1109/CCWC47524.2020.9031117.

6. Y. Arjoune and N. Kaabouch. On Spectrum Sensing, a Machine Learning Method for Cognitive Radio Systems. IEEE International Conference on Electro Information Technology (EIT), Brookings, SD, USA, 2019, pp. 333-338, doi: 10.1109/EIT.2019.8834099.

7. Arjoune Youness, Peri Sai, Sugunaraj Niroop, Sadhukhan Debanjan, Nord Michael, Krishnamoorthy Gautham, Flynn David, Ranganathan Prakash. Thermal Imagery Based Instance Segmentation for Energy Audit Applications in Buildings. IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 5974-5976, doi: 10.1109/BigData47090.2019.9006077.

8. Y. Arjoune, W. C. Hu and N. Kaabouch. Chebyshev Vandermonde-like Measurement Matrix Based Compressive Spectrum Sensing. IEEE International Conference on Electro Information Technology (EIT), Brookings, SD, USA, 2019, pp. 028-033, doi: 10.1109/EIT.2019.8834261.

9. Y. Arjoune, and Saleh Faruque. Double Deep Q-learning and SAC-Based Hybrid Beamforming for 5G and Beyond millimeter-wave Systems. IEEE International Conference on Electro Information Technology (EIT), Michigan, MI, USA, 2021, pp. 028-033, doi: 10.1109/EIT.2019.8834261.

# Chapter 2

# Background and Literature Review

Next-generation wireless communication should be able to handle a much larger and more diverse set of wireless devices. As depicted in Fig. 2.1, 5.3 billion Internet users are expected by 2023. Also, as reported in Fig. 2.2, machine to machine communication has the most considerable part with 50% followed by smartphones with 23%. In addition, Fig. 2.3 shows that most of the technologies enabled by next-generation wireless communication require High Definition (HD) and Ultra High Definition (UHD), requiring high data rates. This explosive demand creates the 1000× capacity challenge for the fifth-generation wireless communication (5G) and beyond systems [9].

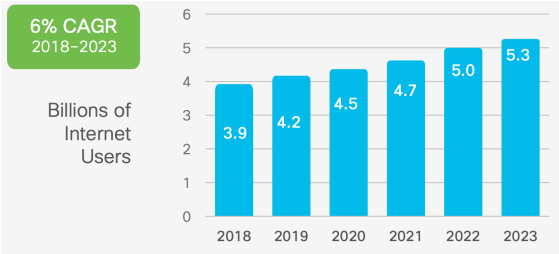


Figure 2.1: Source: Cisco Annual Internet Report, 2018-2023

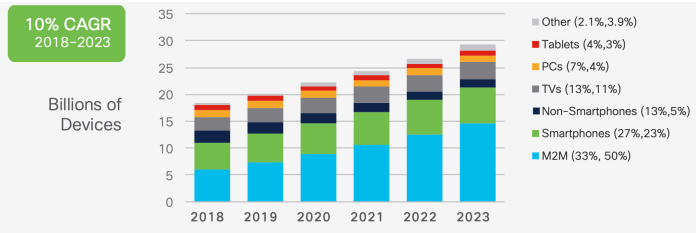


Figure 2.2: Source: Cisco Annual Internet Report, 2018-2023

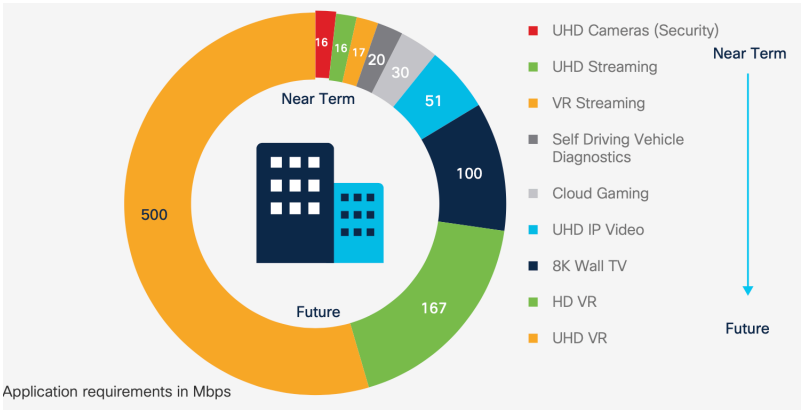


Figure 2.3: Source: Cisco Annual Internet Report, 2018-2023

Wireless capacity depends on several factors such as bandwidth, spectral efficiency, and the number of simultaneous transmissions. To achieve this goal, the maximization of all these factors is essential. Three technologies are combined to achieve the desired capacity: 1) network densification, 2) millimeter-wave communications, 3) and Massive MIMO beamforming [20, 21]. Network densification is a practical approach to increasing wireless communication's capacity by reducing the network cell size. This approach has been previously used with 3G and 4G. However, it presents several challenges, such as the unaffordable cost of installation and maintenance. As shown in Fig. 2.4, cell densification does not improve the network performance as most of the regions are blue and it does not achieve uniform coverage (Fig. 2.5). mmWave bands provide higher bandwidth, thus enabling next-generation wireless communication to increase the capacity. However, they have several propagation issues, including strong path loss, atmospheric loss, and rain absorption, therefore suitability only for the short-range transmission. Massive MIMO provides spatial multiplexing, enabling space-division multiple-access where parallel transmissions may increase network capacity. Massive MIMO provides an opportunity to enable beamforming, which enhances interference management, focusing the signal toward the desired users and nulling the undesired signals. Therefore, beamforming is an effective way of enhancing the cybersecurity

of wireless communication systems against denial-of-service attacks such as jamming, which exploits interference in a harmful way. Flashlight-like mmWave beams limit interference, therefore increasing wireless communication robustness to jamming attacks.

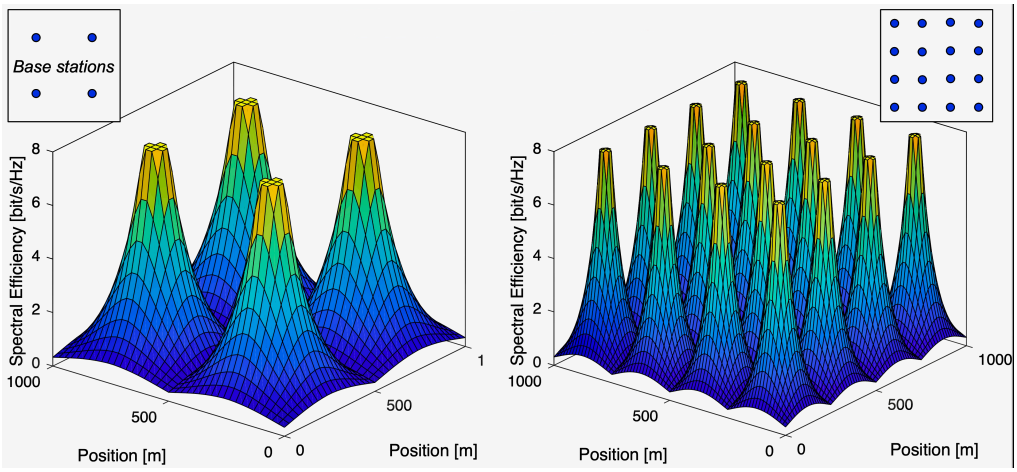


Figure 2.4: Cell Densification [1]

[1]Massive MIMO for 5G below 6 GHz Achieving Spectral Efficiency, Link Reliability, and Low-Power Operation

[2]Massive MIMO for 5G below 6 GHz Achieving Spectral Efficiency, Link Reliability, and Low-Power Operation

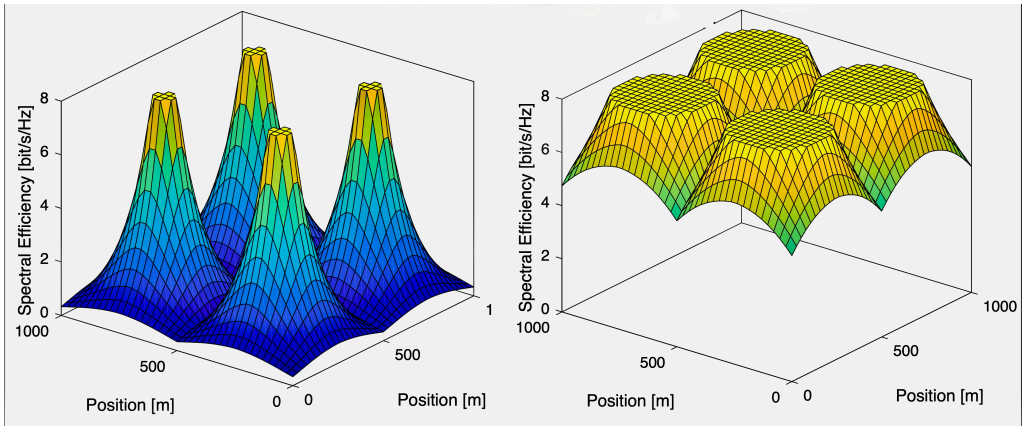


Figure 2.5: Desired Performance [2]

## 2.1 MMWAVE

Large radio spectrum chunks are highly vacant in the millimeter-wave bands, which prompted their exploration for cellular communication. As a result, as shown in Fig. 2.6, bands around 28$GHz$ and 38 $GHz$ have been allocated to cellular mobile communications while 60 $GHz$ has been allocated for Wi-Fi communications [20, 21].

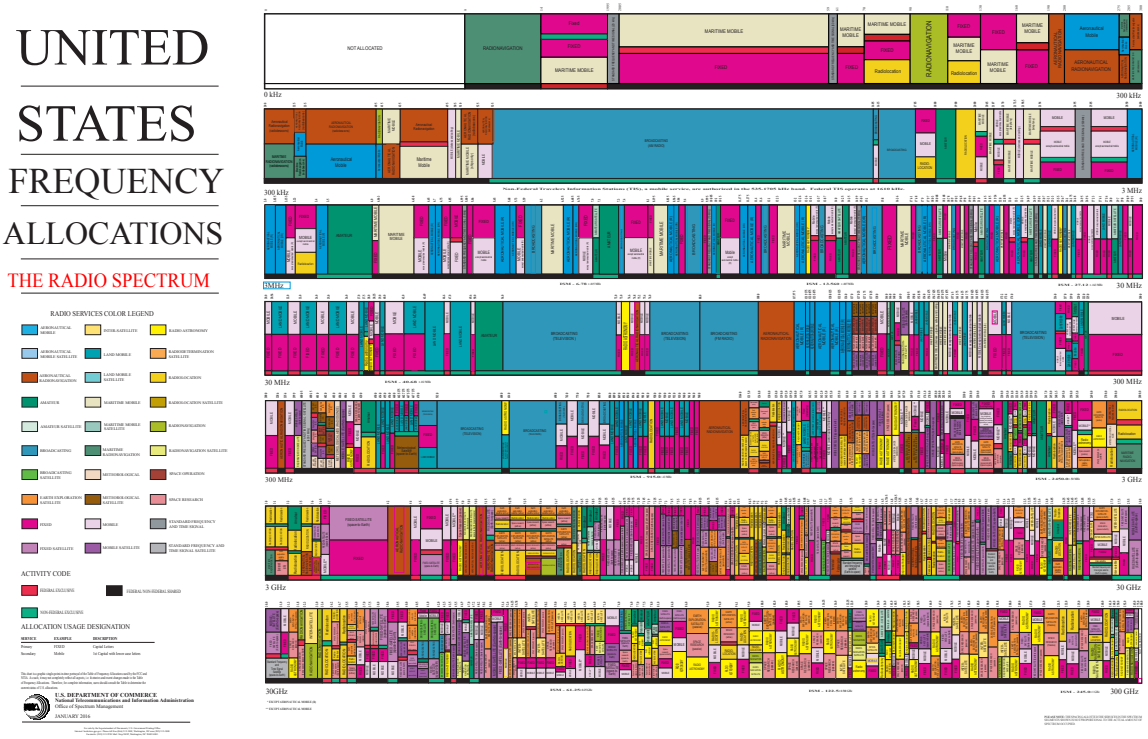


Figure 2.6: U.S. Frequency Allocation Chart as of January 2016

As depicted in Fig. 2.7, mmWave bands have higher bandwidth compared to sub- 6GHz bands. Increasing the frequency means shortening the wavelength, therefore giving rise to a smaller receive aperture, which results in smaller received power (Eq. 2.1). This small aperture explains the extra path loss in the mmWave bands. An order of magnitude frequency increase adds an extra 20 $dB$ path loss regardless of the transmitter-receiver distance [9].

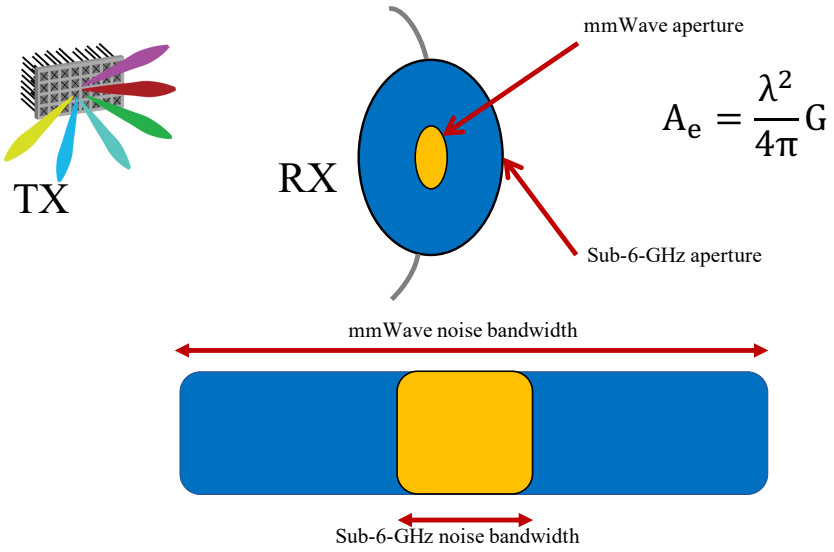


Figure 2.7: mmWave Aperture and Noise Bandwidth [3]

$$P_r = \frac{P_t}{4\pi d^2} \frac{\lambda^2}{4\pi} \tag{2.1}$$

In addition, bandwidth increase means higher noise level and lower Signal-to-Noise Ratio (SNR), which is a direct result of (2.2).

$$N_0 = kT_eB \tag{2.2}$$

As depicted in Fig. 2.8, the mmWave bands at 28 GHz exhibit a link loss of 0.06 $dB/km$ and at 60 GHz this loss becomes 20 dB/km. Fig. 2.8 also shows several attenuation peaks arising from the molecular resonances, which depend on oxygen density and water vapor. The regions between these peaks present longer propagation ranges. Fig. 2.9 shows under heavy rain (25mm/h), the attenuation at 200$m$ at 28 $GHz$ is about 1.4$dB$ while at 150mm/h, it becomes more than 20 $dB$ attenuation [22].

[3] A similar picture appears on "Millimeter Wave Link Configuration with Hybrid MIMO Architectures" by Javier Rodriguez-Fernandez, 2020

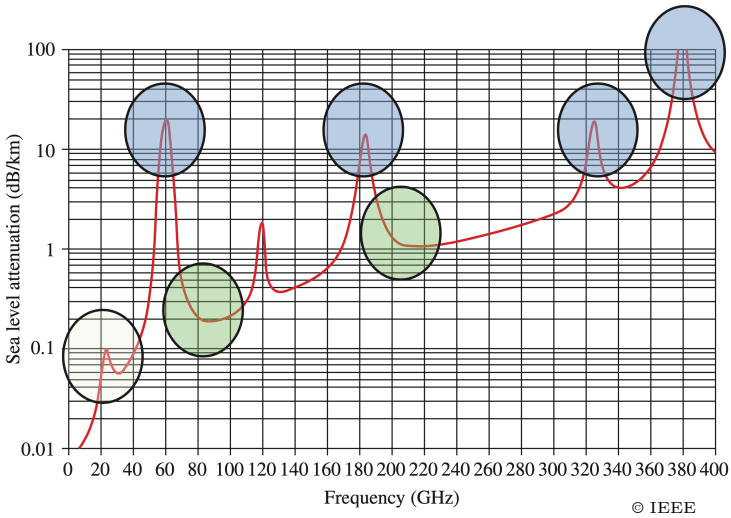


Figure 2.8: Free space attenuation [1]

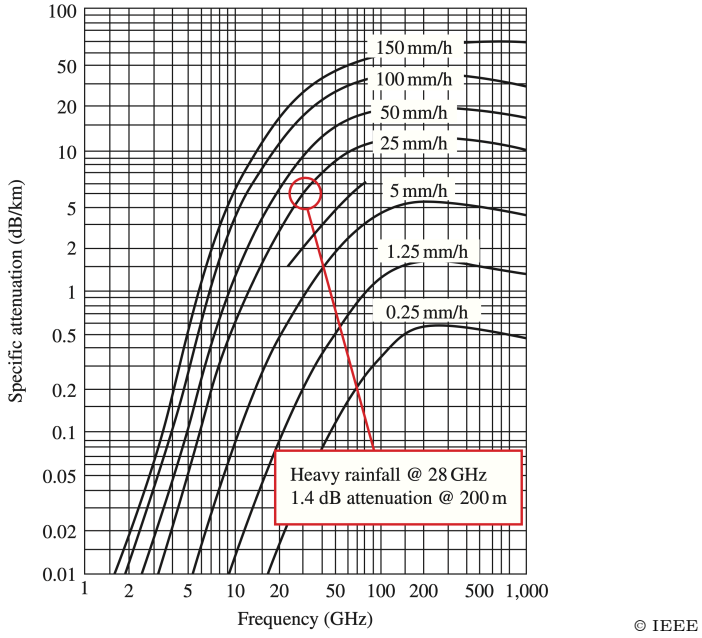


Figure 2.9: Attenuation due to rain versus the frequency [2]

Furthermore, mmWave bands are subject to higher absorptions, reflections, and scattering, making the mmWave signals highly sensitive to blockages. Fig. 2.10 shows about 40 dB of human body blockage that last about 0.5 seconds.

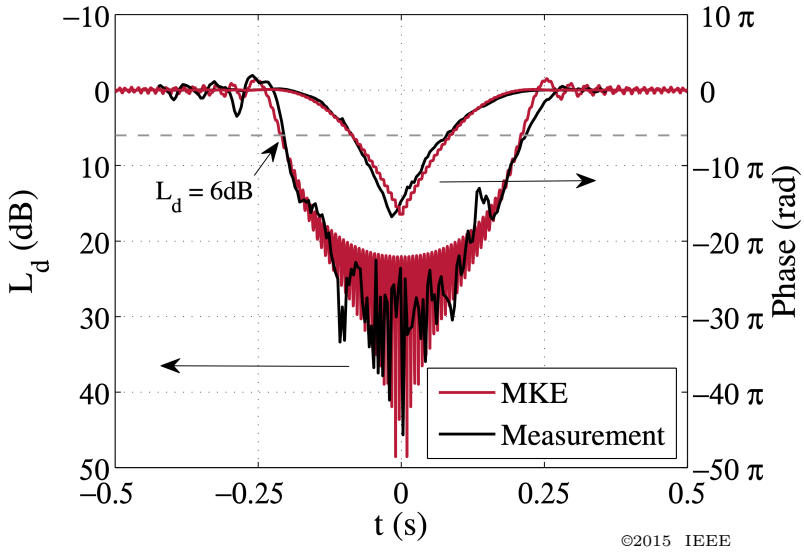


Figure 2.10: Human body blockage example [3]

## 2.2 Massive MIMO

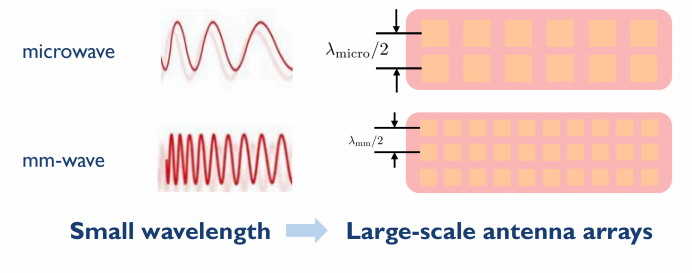


Figure 2.11: Large-scale antenna arrays [4]

Fortunately, the mmWave's small wavelength enables large-scale antenna patches on a small array, enabling a higher antenna gain that compensates for the extra path loss. Massive MIMO benefits include spatial multiplexing, which enables space-division multiple-access, therefore increasing the system capacity. Massive MIMO entails highly directional and steerable flashlight-like beams, subsequently enabling beamforming. Equation 2.3 shows that the received power can be written as the product of the transmit power density and receive

[4]https://research.ece.ncsu.edu/ai5gchallenge/

aperture and antenna gains, which means that increasing the antenna gains increases the received power.

$$P_r = \frac{P_t}{4\pi d^2}\frac{\lambda^2}{4\pi}G_t G_r \tag{2.3}$$

where $G_t$ and $G_r$ denote the antenna gains.

Massive MIMO performance depends heavily on the availability of complete channel state information (CSI) at the base station to find the analog and digital beamformers that maximize the spectral and energy efficiencies [10]. The MIMO systems were successfully deployed in previous wireless generation LTE/4G where the most common architecture is $2 \times 2$ MIMO to $8 \times 8$ MIMO (3GPP-R10). Massive MIMO uses at least 32 antennas and scaling up the number of antennas at the base station to hundreds bring new challenges.

## 2.3 Beamforming architecture

As shown in Fig. 2.12, in fully digital beamforming, each antenna element has a dedicated RFC. Therefore, this architecture achieves the highest performance and supports multiple data streams ($N_s \geq 1$), enabling space-division multiple access. As the number of antennas scales up, the number of RFCs also scales up. Given the high cost and excessive energy consumption of high-frequency hardware components such as ADCs, it is impractical to implement massive MIMO fully digital beamforming systems. It has been estimated that a mmWave massive MIMO system with an array of $N = 64$ antennas [23] requires 16 $W$ considering about 250 $mW$ per RFC [24].

Some of the signal processing functions are analogously performed, hence the analog beamforming architecture, as shown in Fig. 2.13, to reduce the cost and power consumption.

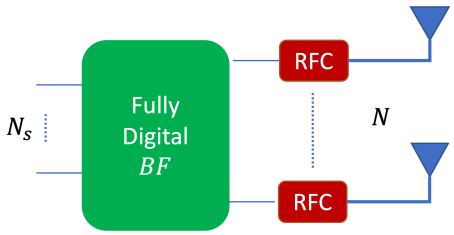


Figure 2.12: Fully digital beamforming

This architecture uses only one RFC, and phase shifters realize the analog precoder. This architecture limits the power consumption and reduces the cost as only one RFC is used, but it has some drawbacks such as the inability to support multiple streams, subsequently not supporting spatial multiplexing, and space-division multiple access, which is one of the main benefits of massive MIMO.

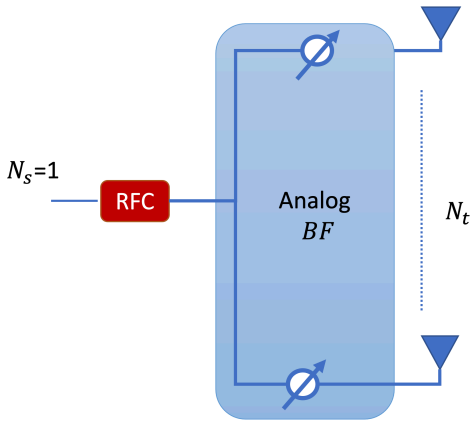


Figure 2.13: Analog beamforming

Hybrid beamforming architecture, shown in Fig. 2.14, mitigates the analog and digital beamforming architecture shortcomings. Known also as hybrid precoding/ combining, this architecture divides the beamformer into a digital beamformer and an analog beamformer. In lieu of one RFC per antenna element, RFCs are reduced to as few as four to eight. Hybrid beamforming enables spatial multiplexing, reduces hardware cost, and decreases

energy consumption. This architecture, therefore, combines analog and digital beamforming benefits [10].

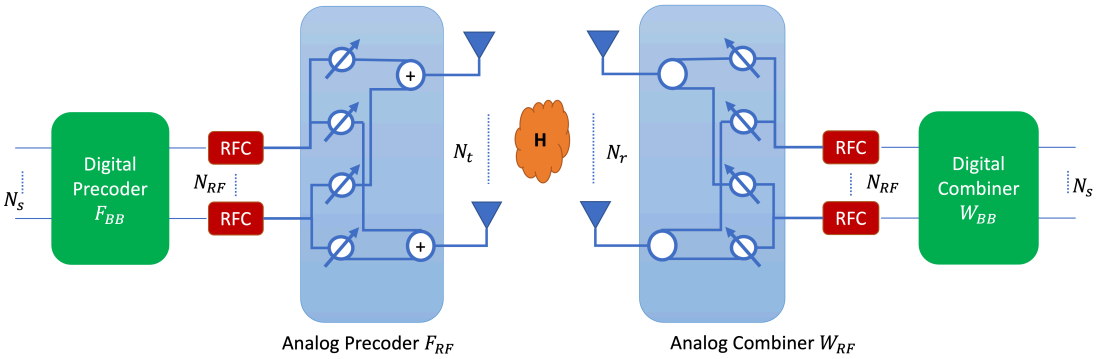


Figure 2.14: Hybrid beamforming

In hybrid beamforming, a low-dimensional digital baseband precoder is mapped to high-dimensional antennas through a series of phase shifters. This mapping can be either fully connected or partially connected. A fully-connected mapping strategy connects each RFC to every antenna element, while partially connected mapping connects each RFC to a group of antenna elements [25]. These mapping strategies answer the questions of which RFC should be connected to which antenna element and how the mapping should be implemented in terms of hardware; there are mainly two hardware implementations: single-phase shifter and double-phase shifter. The first implementation makes the problem of finding the precoding/-combining matrices non-convex and challenging to solve, while the second implementation relaxes the convexity and boosts the spectral efficiency but at the expense of high hardware cost.

In general, the design of hybrid beamforming aims to improve spectral efficiency, boost computational efficiency, and increase hardware efficiency. Several research studies that investigated this design are presented in detail in the next section.

Table 2.1: Summary of beamforming architecture

| **Architecture** | **Concept** | **Advantage** | **Disadvantages** |
|---|---|---|---|
| Fully digital Beamforming | • One RFC per antenna element<br>• Beamforming is in digital baseband | • Optimal performance<br>• less challenging design<br>• Low computation cost | • High power consumption<br>• High hardware cost |
| Analog beamforming | • One RFC<br>• Beamforming in analog | • Low hardware cost<br>• Low power consumption | • Low performance<br>• Does not support SDMA |
| Hybrid beamforming | • Divide beamforming into analog and digital beamformers<br>• Few RFCs | • Near-optimal performance<br>• Low hardware cost<br>• High energy efficiency | • Design is beamformers is non-convex<br>• High hardware cost |

## 2.4 Literature Review

### 2.4.1 Hybrid Beamforming

Preliminary work on hybrid beamforming states that two radio frequency chains are enough to achieve the performance of the fully digital beamforming [26] when the number of data streams is one, this result has been generalized to the case where the number of streams is higher than one. For example, the authors of [27, 28] showed that using a number of RFCs twice the number of users can achieve the performance of the fully digital beamforming. However, this solution is yet expensive in terms of hardware cost. Therefore, researchers are still working on achieving a performance close to the fully digital beamform-

ing with a number of RF chains less than twice the number of data streams. Consequently, several approaches have been investigated in the literature. As shown in Fig. 2.15, the state-of-the-art hybrid beamforming techniques can be categorized into two main categories: traditional optimization-based category and machine learning-based category. The first category includes "OMP" [11] and its variants "SOMP" as well as "KKT", "MO" [12], altMin, and MMSE [13] algorithms. The second category includes three sub-categories: machine learning-based [14] [15], deep learning-based [16] [17] [18], and reinforcement learning-based. Machine learning-based techniques investigated linear regression, multi-neuron perceptron, and support vector machines. Deep learning-based techniques investigated deep neural network [16], convolutional neural network [29], and quantized convolutional neural network [18]. Deep reinforcement learning-based techniques investigated tabular Q-learning [30], deep Q-learning [31], and DDPG [31]. These state-of-the-art techniques are summarized in 2.2, which shows the reference, the technique used, and the evaluation metric as well as the hybrid beamforming parameters used.

Traditional hybrid beamforming techniques solve the hybrid beamforming model using traditional optimization techniques. For instance, the authors of [11] proposed orthogonal matching pursuit (OMP) for sparse precoding in hybrid beamforming. The optimization of the precoders and the combiners is formulated as a sparse recovery problem and solved with OMP [38, 39] by exploiting the sparsity of the scattering in mmWave channels. The authors of [12] proposed a method based on manifold optimization and altering minimization (Altmin). The authors of [23] proposed a hybrid beamforming architecture inspired by the concept of successive interference cancellation (SIC) considering a partially-connected structure. Traditional optimization techniques achieve near-fully digital beamforming performance but present some drawbacks. For instance, OMP requires a perfect knowledge of the angle-of-departure [40] and show prominent performance loss for a small number of

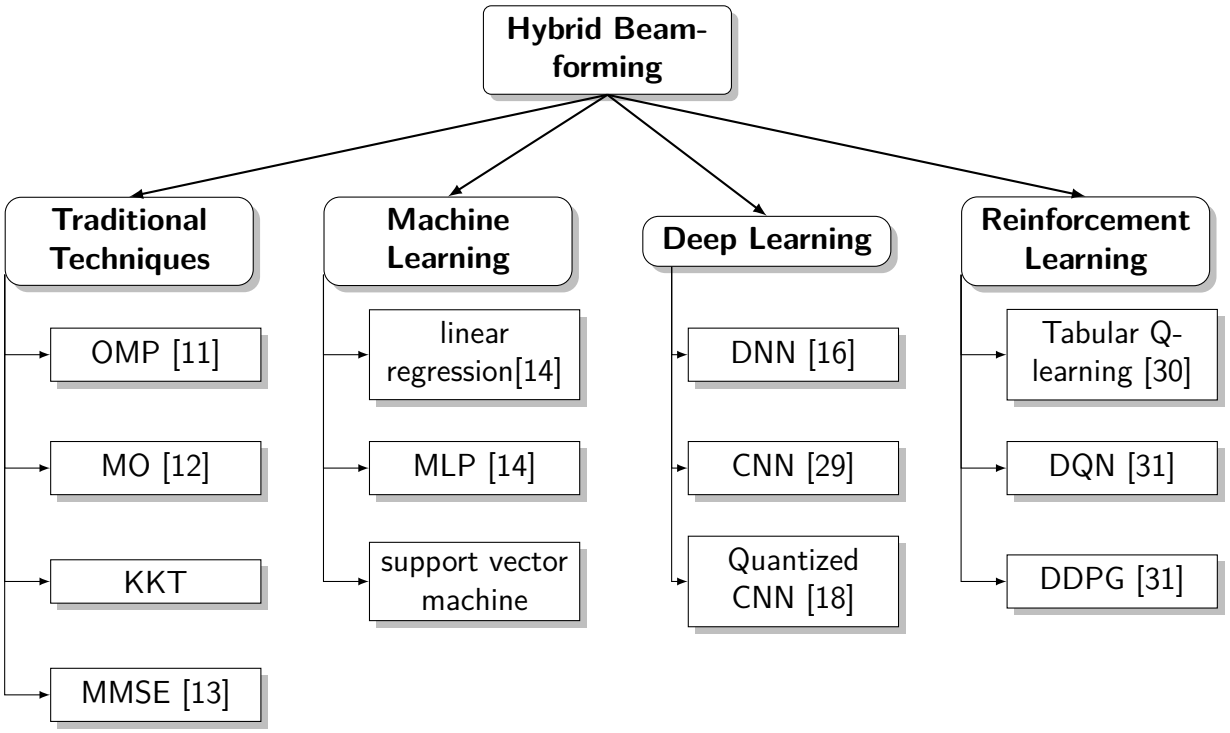


Figure 2.15: Taxonomy of hybrid beamforming techniques

RFCs, while the KKT-based and MO-based algorithm are extremely time-consuming. Altmin has low computational efficiency. Partially connected structures show low performance compared with fully-connected structures (OMP, Altmin) and need higher RFCs to achieve fully-digital beamforming performance.

Machine learning has been proposed to overcome the issues of traditional optimization techniques. For instance, the authors of [14] formulated hybrid beamforming as a regression problem, and they investigated several machine learning techniques, such as linear regression, RBFN, and exact RBFN networks, general regression neural network (GRNN), RBFN with Bayesian regularization, and multi-layer perceptron (MLP). The authors considered three configurations: $16 \times 4$, $64 \times 16$, and $144 \times 36$ to validate proposed models using spectral efficiency, and they used optimal beamforming as a baseline for this comparison.

Deep learning has been proposed, as it can outperform machine learning models. For in-

Table 2.2: Hybrid beamforming in 5G mmWave communications

| Ref | TA | ML | DL | RL | DQN | SE | BER | HB parameters |
|---|---|---|---|---|---|---|---|---|
| [11] | ✓ | | | | | ✓ | | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 64, 256$, $N_{tx}^{rf} = 4, 6$, $N_{rx} = 16, 64$, $N_s = 1, 2$ |
| [14] | | ✓ | | | | ✓ | | $N_{tx} = 16, 64, 144$, $N_{tx}^{rf} = 4, 6$, $N_{rx} = 4, 16, 36, 64$ |
| [16] | | | ✓ | | | ✓ | | - |
| [17] | | | ✓ | | | ✓ | | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 64$, $N_{tx}^{rf} = 2$, $N_{rx} = 16$, $N_s = 2$ |
| [29] | | | ✓ | | | ✓ | | $N_{tx} = 64$, $N_{tx}^{rf} = 4$, $N_{rx} = 64$, $N_s = 4$ ; $N_{tx} = 256$, $N_{tx}^{rf} = 16$, $N_{rx} = 256$, $N_s = 4$ |
| [31] | | | | | ✓ | ✓ | ✓ | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 128$, $N_{tx}^{rf} = 6$, $N_{rx} = 32$, $N_s = 6$ |
| [32] | | | | ✓ | | ✓ | | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 9$, $N_{tx}^{rf} = 2$, $N_{rx} = 4$, $N_s = 1, 2$ |
| [33] | | ✓ | | | | | ✓ | $N_{cl} = 5$, $N_{ray} = 5$, $N_{tx} = 64$, $N_{tx}^{rf} = 4$, $N_{rx} = 16$, $N_s = 4$ |
| [34] | | | ✓ | | | ✓ | | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 128$, $N_{tx}^{rf} = 4$, $N_{rx} = 16$, $N_s =$ |
| [35] | | | ✓ | | | ✓ | | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 9$, $N_{tx}^{rf} = 2$, $N_{rx} = 4$, $N_s = 1, 2$ |
| [36] | | | ✓ | | | ✓ | | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 9$, $N_{tx}^{rf} = 2$, $N_{rx} = 4$, $N_s = 1, 2$ |
| [37] | | | ✓ | | | ✓ | | $N_{cl} = 8$, $N_{ray} = 10$, $N_{tx} = 9$, $N_{tx}^{rf} = 2$, $N_{rx} = 4$, $N_s = 1, 2$ |
| [18] | | | ✓ | | | ✓ | | $N_{cl} = 4$, $N_{ray} = 5$, $N_{tx} = 25$, $N_{tx}^{rf} = 4$, $N_{rx} = 25$, $N_s = 1, 2$ |

stance, the authors in [16] proposed a deep learning framework for hybrid precoding in mmWave. This framework consists of three hidden layers with (128, 400, 256) units, respectively. The model shows better performance in terms of spectral efficiency and has lower complexity compared with SVD based technique. Another deep learning approach is proposed in [17], which can outperform OMP by $2dB$ in terms of bit error rate. Another deep neural network design is proposed in [41]. The model shows good performance, but it was

validated only with small-size arrays. A joint antenna selection and hybrid beamforming based on convolutional neural networks are proposed in [29]. A quantized convolutional neural network (QCNN) with three convolution layers and two fully connected layers of 512 neurons each is proposed in [18], which can achieve comparable performance to SOMP. Deep learning-based hybrid beamforming can achieve good performance. Training deep learning models, however, requires computational resources and time.

Only a few reinforcement learning approaches have been proposed for hybrid beamforming. For instance, the authors of [30] proposed tabular Q-learning-based hybrid beamforming. The authors considered the phase $\theta_{i,j}$ as the actions, which takes only two possible values $\pm\pi/2$. Each time the model selects a phase, it ends up in a new state, and this state is only the precoding matrix $F_{rf}$. The model is validated with a simulation that considers a small size array. The achievable data rate, which brute-force search provides, can be reached using the proposed approach. The authors of [31] proposed a deep deterministic policy gradient (DDPG), which consists of an actor-network $A$ and a critic network $C$, to deal with continuous actions space. In this formulation, the precoder/combiner matrices of the previous learning iteration are taken as the state, while those of the current learning iteration are taken as the action, while the reward is computed as the achievable data rate. This approach performs well in terms of spectral efficiency, bit error rate, and time consumption; in fact, it can achieve relative performance to some of the state-of-the-art algorithms such as “OMP”, “KKT”, and “MO”. DDPG, however, is often reported to suffer from instability and converge to poor solutions if not diverging [42].

### 2.4.2 Channel Estimation

Hybrid beamforming requires channel estimation, which is challenging to estimate due to the limited number of RFCs used in hybrid beamforming architecture. Channel estimation in a hybrid architecture is challenging for two reasons: As shown in Fig. 2.16, the high-dimensional channel **H** is compressed and seen at the output of the RFCs, which are few compared with the number of the receiving antennas. Therefore, the channel should be reconstructed from a few measurements. In addition, the signal-to-noise ratio at the early stage of channel acquisition is low, making the channel estimation more challenging.

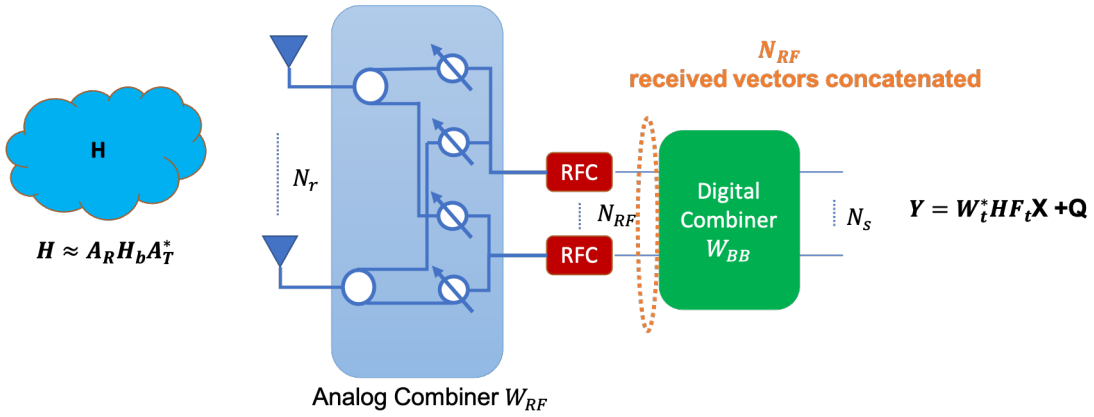


Figure 2.16: Channel estimation in hybrid architecture

Fortunately, as shown in Fig. 2.17 mmWave channels are sparse in the angular domain; therefore, angle-of-arrival, angle-of-departure, and the gain of the dominant path model the wireless channel. Thus, the compressive sensing framework (Fig. 2.18) can be used to reduce the complexity by exploiting the mmWave channel sparsity. The mmWave channel estimation can be formulated as a sparse recovery problem and solved using sparse recovery algorithms such as convex and relaxation techniques, greedy techniques, or Bayesian compressive sensing methods. For instance, the authors of [43] proposed a greedy approach to solve the sparse recovery algorithm. However, greedy techniques often lead to good performance, but it does not converge to the global optimal value and usually find a local minimum. The use of convex

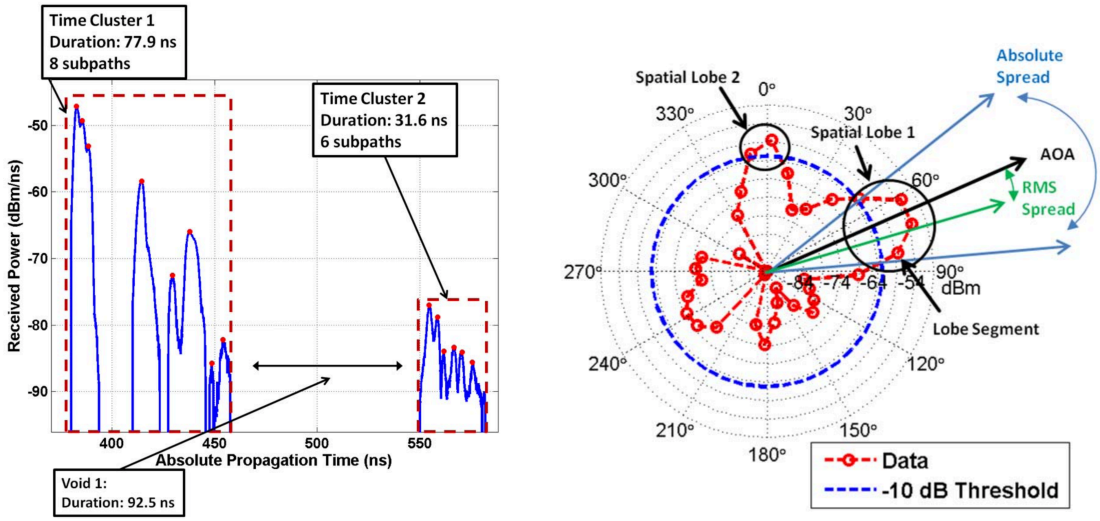


Figure 2.17: Clustered channel [4, 5] © 2014 IEEE

and relaxation techniques is costly in terms of computational complexity. Continuous basis pursuit is proposed

The authors of [44] proposed a channel estimation based on adaptive compressed sensing. The authors have formulated the channel estimation and hybrid precoding as a sparse recovery problem where the sensing matrix is formed by the matrix at the baseband. Deep learning compressed sensing (DLCS) channel estimation scheme is proposed in [45]. Deep learning requires the training dataset labeled and ready before the training can start. The authors of [46] proposed a channel estimation in hybrid beamforming using adaptive overcomplete dictionary techniques. They used the normalized mean square error to evaluate their techniques. The reported results showed that an adaptive overcomplete dictionary has almost similar performance as traditional techniques and 2D-MUSIC. However, the authors considered high signal-to-noise ratio values, which is not the case in hybrid beamforming at the training stage. The authors of [47] proposed an adaptive compressive sensing-based channel estimation method. The authors proposed continuous basis pursuit (CBP). The authors showed that this algorithm outperforms grid-based introduced by the authors of [48]

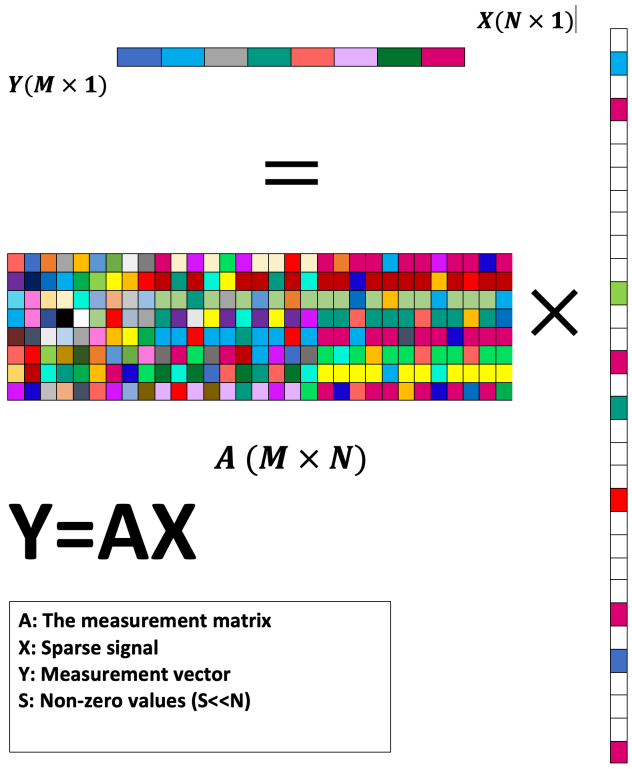


Figure 2.18: Compressive Sensing block diagram

which is expected as the discretization of the angles and using grid-based introduce quantization error. The authors [49] proposed orthogonal matching pursuit. The authors used a grid-based approach, but this time they used non-uniform angle quantization. This approach reduces the redundant dictionary coherence. This paper suggests that simple columns of the RF analog precoder matrix permutation improves channel estimation performance. The authors of [50] proposed time-domain compressed sensing mmWave channel estimation in a hybrid architecture. These techniques considered one RFC and two RFC. The results showed that the proposed technique requires at least 80 training frames to achieve near-optimal performance. The authors of [51] proposed an algorithm for beam tracking and channel estimation. This algorithm builds upon the greedy method OMP. The authors of [52] proposed a multi-resolution compressed sensing-based beam and channel acquisitions

in mmWave systems. While compressive sensing-based channel estimation is well studied, there are still some challenges and open research questions. How the measurement should be acquired and which algorithm has optimal performance depending on the sparsity level, the number of measurements, and the complexity time.

The authors of [53] estimated the OFDM-mmWave channel estimation using the theory of compressive sensing and deep learning based on CNNs. The basic idea of this methodology is that the estimation of the support provides information on the AoAs/AoDs of the strongest paths, while estimation of their amplitude provides information on the corresponding gain. The CNN takes the correlation of the received signal vectors and the measurement matrix to estimate the channel amplitudes. They tested the proposed method using the Raymobtime dataset.

The authors of [54] proposed to optimize the measurement vector using the surrounding environment. The authors indicated that leveraging out-of-band details on the propagation environment can help in reducing the number of training pilots. The authors considered a OFDM-mmWave channel and neural network for hybrid beam prediction and convolution layers to optimize channel compression. They tested their method using DeepMIMO dataset. The authors of [55] proposed deep learning for FDD massive MIMO channel estimation. The authors did not consider hybrid architecture. The authors of [56] proposed a reinforcement learning model for channel estimation in MIMO-OFDM system where the successive denoising is formulated as Markov decision process and solved using Q-learning. The authors of [57] proposed to optimize the conventional linear minimum mean squared error channel estimator for MIMO systems using reinforcement learning. This method optimizes the selection of the detected symbols obtained from the data as additional pilots to reduce the channel estimation error. Both of the previous papers did not consider hybrid beamforming architecture. The authors of [58] addressed the challenge of channel estima-

tion and hybrid combining in OFDM-terahertz massive MIMO system. They proposed a low-complexity beam squint mitigation scheme. Considering the sparsity of the terahertz channel, they proposed an algorithm based on OMP to estimate the channel from a few training pilots. The authors of [59] proposed Two-dimensional ESPRIT-based channel estimation in OFDM-mmWAve by estimating the AoAs/AoDs combined with minimum searching and least-square. The authors of [60] used the beamspace sparsity and an enhanced OMP to estimate the OFDM-mmWave channel. They estimated the path gain and doppler using a beamforming polling strategy to reduce the training overhead. The authors of [61] estimated the hybrid selective mmWave channel while addressing the issue of hardware impairments of the antenna array. The authors modeled these impairments in the sparsifying dictionaries. The authors solved this issue using OMP/SW-OMP.

### 2.4.3 Jamming Detection and Mitigation

The massive MIMO system may face the challenge of the near-far problem where the user retrieves the signal (IS) from the base station under a strong intended jamming signal. Therefore, detection and mitigation of jamming are essential to improve these systems' reliability and cybersecurity. Over the past decade, several detection and mitigation techniques have been proposed. Detection methods can be classified into: traditional approaches [62], [63], machine learning models [64, 65, 66, 67, 68], deep learning models [69, 70], and reinforcement learning methods [71].

Traditional jamming detection methods use statistical, threshold, or fuzzy logic approaches to detect any sudden change in the link performance. For instance, the authors of [62] proposed a statistical approach, exponentially weighted moving average (EMWA), for jamming detection in wireless sensor networks. EMWA monitors the packet inter-arrival time (IAT)

Table 2.3: State-of-art jamming detection methods

| **Approach** | **Models** | **Advantages** | **Disadvantages** |
|---|---|---|---|
| Traditional methods | Uses fuzzy logic, statistical methods, | Simple | Ineffective with jammers that constantly change their strategy |
| ML-based | SVM, Neural network, logistic regression, decision trees | Achieve high detection accuracy | Requires labeled datasets |
| DL-based | Convolutional Neural Networks | Good performance | Require huge datasets and computation resources |
| RL-based | Q-learning | Learn from experience | overestimation bias |

over time to detect the presence of the jammer. The use of IAT as the only metric reduces the complexity of the detection method but can be easily defeated. The authors of [63] defined several rules to detect jamming

Machine learning-based detection methods formulate jamming detection as a classification problem, and several classifiers can be used. For instance, the authors of [72] proposed machine learning-based jamming detection for IEEE802.11 where several machine learning techniques such as random forest, decision tree, adaptive boosting, support vector machine, and expectation-maximization were used to detect reactive and constant jammers.

Machine learning techniques–supervised, unsupervised, and reinforcement learning– have been proposed to detect jamming. For instance, supervised techniques have been proposed to detect jamming attacks using neural networks, support vector machines, and logistic regression [66, 67, 68, 73, 74, 75]. Unsupervised techniques have been proposed in [76].

Mitigation of jamming is well studied in the literature. Methods such as direct sequence spread spectrum [77], frequency-hop spread spectrum, channel surfing [78], game theory [79], deep learning, deep reinforcement learning [80], polarization techniques, beamforming techniques [81] have been investigated.

The spread spectrum provides protection against interference by making the information-bearing signal occupy large bandwidth than that necessary for its transmission. In this way, the signal may go undetected through the channel and make it harder for the jammer to know which frequency to target. As a result, the jammer is forced to spread its power over a wideband, consuming more power and less effective. For example, the authors of [82] proposed transversal filters to reject narrow-band interference in spread-spectrum systems. The authors of [83] proposed transform-domain adaptive filtering to suppress narrow-band jammers in a spread-spectrum receiver. The authors of [77] proposed a DSSS with variable PN code length for mitigating jamming. The authors of [84] studied the effect of jamming on DSSS, and they showed that DSSS is robust against jamming. The authors of [85] proposed the use of direct noise to generate DSSS instead of using pseudo-noise. The authors of [86] showed that the DSSS bit error rate could not be improved by blindly increasing the length of the PN code. DSSS techniques provide strong protection against jamming attacks, but they suffer from some practical limitations such as the inability of the physical devices to generate a large PN sequence [87].

Frequency-hop spread spectrum (FHSS) randomly hops the carrier from one frequency to another. This method requires pre-planning to determine the sequence of the frequency to be used [88]. The frequency-hopping spread spectrum is considered to be cost-effective and low-complex compared to the DSSS. Also, the performance of The FHSS depends on the hop rate [89]. For instance, FHSS with higher hop rates makes the synchronization between the transceivers more challenging and calls for more complicated synchronization techniques. On the other hand, FHSS with a slow rate can give follow jammer, which is a sophisticated jammer, the advantage to intercept, with an acceptable probability, the instantaneous frequency of FHSS [88]. Given that, the follow jammer can generate an appropriate jamming strategy. Some recent techniques combine DSSS and FHSS, called hybrid spread spectrum

(HSS) [90]. For instance, the authors of [91] combined FHSS using 16-bit PN code, which was derived from the same key used for FHSS channels generations. Some other varieties of FHSS, such as chaotic FHSS, have been proposed in [92]. The authors of [93] provided a performance analysis of FHSS and CFHSS under different jamming types. They showed that CFHSS has a slightly better performance compared with FHSS.

When two radio devices establish the communication, they operate on a specific channel. When a malicious node jams this channel, it seems natural to migrate to another channel. This is known as channel surfing is another common jamming mitigation technique [94]. Channel surfing is a link-layer technology that can be applied to wireless devices which do not possess any frequency hopping feature as FHSS is a physical layer technology [95]. For example, the authors of [78] proposed a channel hopping scheme to mitigate jamming attacks in wireless LANs. A drawback of channel surfing is that the next channel is determined after a negotiation between the transceivers. Those prior negotiations can be intercepted by the jammer [78].

The authors of [79] proposed a game theoretical model to mitigate jamming attacks. The authors of [96] proposed a timing channel to mitigate jamming attacks. The main polarization-based methods for countering active jamming are polarization encoding [97, 98], hyper- complex and spatio-temporal polarization processing [99], and polarization filtering [100, 101, 102].

Beamforming is a technique that counteracts the impact of jamming attacks. The authors of [103] proposed jamming mitigation using polarized beamforming for jamming that is close in space to the receiver. The authors of [81] proposed a jamming signal suppression using fully digital beamforming. Recently, the authors of [104] proposed a jamming mitigation technique based on intelligent reflecting surface (IRS) where they considered an aerial (IRS). The successive convex approximation has been used while the reflection beamformer is solved

using manifold optimization to determine the optimal location of IRS.

The authors of [105] proposed a deep learning framework for launching and mitigating jamming attacks where adversarial machine learning was introduced to launch jamming attacks. To attacker uses machine learning to learn the next channel. One way to mislead the attacker is to take wrong actions.

### 2.4.4 Spectrum Sensing

Cognitive radio [106] is a form of spectrum sharing [107, 108, 109]. It has been proposed to enable massive internet-of-things [110, 111] and overcome the scarcity of the radio spectrum. The sensing function of cognitive radio plays a key role in enabling cognitive radio [112]. Over the past decade, spectrum sensing has been addressed by several researchers focusing on both narrow-band and wide-band spectrum sensing techniques. As shown in Fig.2.19, spectrum sensing techniques can be classified into narrowband spectrum sensing and wideband spectrum sensing. Narrowband sensing techniques include energy detection [113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125], autocorrelation or cyclostationary detection [126, 127], and matched filter [128, 129, 130, 131] while wideband sensing techniques include nyquist and sub-nyquist based. Wideband nyquist based include wavelet techniques [132, 133, 134, 135, 136], joint multi-band detection [137], and filter bank [138, 139, 140]. Wideband sub-nyquist based techniques include adative wideband compressive sensing [141], data assisted wideband compressive sensing [142], geolocation compressive wideband sensing [143, 144].

Machine learning has been proposed to overcome the issues of model-based sensing algorithms. Examples of these techniques include machine learning for cooperative sensing [145, 146, 147]

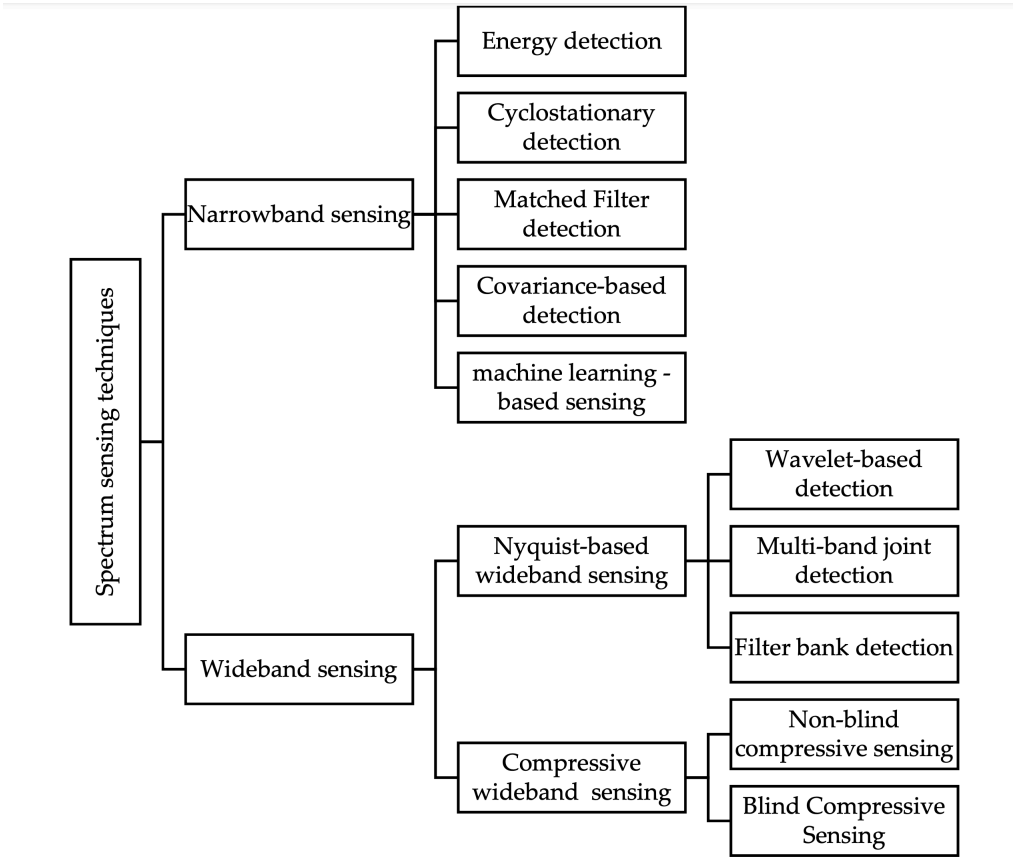


Figure 2.19: Spectrum sensing taxonomy

In addition, the sensing function is vulnerable to cyber-attacks such as spectrum sensing data falsification (SSDF) [148, 149]. This type of attack tends to disrupt the sensing function of cognitive radios to create intentional interference or mislead secondary users not to access an available channel [148, 150, 151].

Several detection methods have been proposed [152, 153, 154, 155]. For instance, the authors of [156] proposed an adaptive learning mechanism. The authors of [157] proposed abnormality detection algorithm.The authors of [158] proposed unsupervised learning.The authors of [159] proposed a Bayesian learning method. The authors of [160] proposed SVDD algorithm. The authors of [161] proposed a self-organizing map-based scheme against probabilistic SSDF attack.

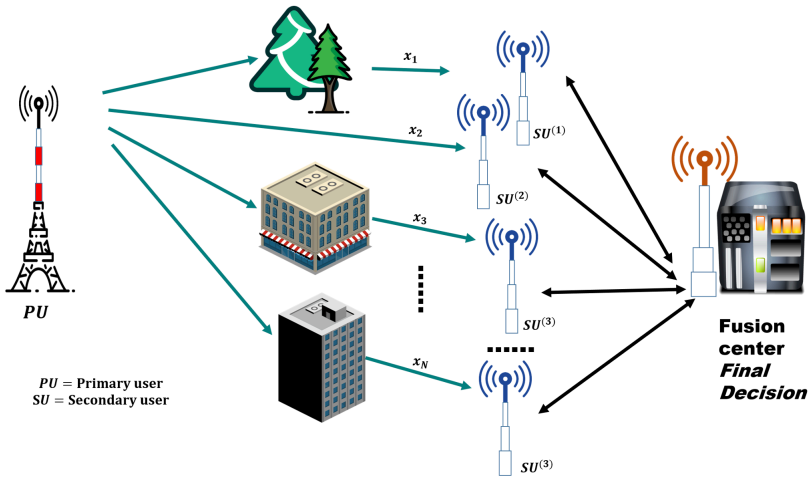


Figure 2.20: Cooperative spectrum sensing

## 2.5 Summary

Next-generation wireless communications will exploit mmWave communications and massive MIMO to provide high antenna gain that can be used to compensate for the extra path loss in mmWave bands. Digital beamforming and analog beamforming presents many shortcomings. Subsequently, future generations of communications systems will adopt hybrid architecture to reduce hardware costs and power consumption. Beamforming and channel estimation are more challenging in hybrid architecture compared with digital systems, as there are only a few RF chains and the non-convex constraint imposed by the series of phase shifters. Hybrid beamforming and channel estimation remain an open research problem. The mmWave channel's sparsity makes it possible to leverage the framework of compressive sensing to develop low-complexity algorithms. In addition, the framework of deep reinforcement learning is suitable to deal with non-convex problems such as the one of hybrid beamforming and is not yet fully explored.

The speed and low latency are not the only requirements of the future wireless communication

systems but also the reliability, especially that wireless networks have an open transmission medium which makes them prone to interference. The development of techniques for better interference management is highly desirable.

# Chapter 3

# Deep Reinforcement Learning Based Hybrid Beamforming

In this chapter, we describe the system model of massive MIMO hybrid beamforming in mmWave. We formulate the problem of finding the precoder/the combiner using the Markov decision process framework and describe its solution using the theory of deep reinforcement learning. In this respect, we have considered hybrid beamforming with a fully connected mapping strategy and single-phase shifter hardware implementation. First, we describe the hybrid beamforming model; then, we briefly review the theory of reinforcement learning. Next, we describe the hybrid beamforming transformation using reinforcement learning, and we provide its solution using advanced deep reinforcement learning using soft-actor critic and TD-3 methods, and we show that implementation details and examples of the obtained results.

## 3.1 Hybrid Beamforming System Model

Consider a single user massive MIMO (SU-MIMO) hybrid beamforming system with $N_t$ and $N_r$ transmitting and receiving antennas, respectively. Let $N_t^{RF}$ and $N_r^{RF}$ denote the numbers of the RF chains at the transmitter and the receiver, respectively. Also, $N_s$ denotes the number of data streams.

In this architecture, the digital precoder structure applies the baseband precoder $F_{BB} \in \mathbb{C}^{N_t^{RF} \times N_s}$ to the transmit signal $s \in \mathbb{C}^{N_s}$. Then, it is passed through the RF precoders $F_{RF} \in \mathbb{C}^{N_t \times N_t^{RF}}$ to transmit antennas. Thus, the transmit signal can be written as [15],

$$\mathbf{s}_t = \mathbf{F}_{RF}\mathbf{F}_{BB}\mathbf{s} \tag{3.1}$$

where $\mathbf{s}$ is the symbol vector.

The analog precoder is given as,

$$\mathbf{F}_{RF} = \begin{pmatrix} e^{j\theta_{1,1}} & e^{j\theta_{1,2}} & \dots & e^{j\theta_{1,N_t^{RF}}} \\ e^{j\theta_{2,1}} & e^{j\theta_{2,2}} & \dots & e^{j\theta_{2,N_t^{RF}}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{j\theta_{N_t,1}} & e^{j\theta_{N_t,2}} & \dots & e^{j\theta_{N_t,N_t^{RF}}} \end{pmatrix} \tag{3.2}$$

where $\theta_{i,j} \in \Theta, i \in \{1, \cdots, N_t\}, j \in \{1, \cdots, N_t^{RF}\}$ and $\Theta = [0, 2\pi]$ for continuous analog phase shifter and $\Theta = \{0, \frac{2\pi}{2^{qbit}}, \cdots, \frac{(2^{qbit}-1)2\pi}{2^{qbit}}\}$ and $qbit$ denotes the number of quantization bits of the phase shifters.

The received signal at the receive antenna given in [29, 43] by,

$$\mathbf{y} = \sqrt{\rho}\mathbf{H}\mathbf{F}_{RF}\mathbf{F}_{BB}\mathbf{s} + \mathbf{n} \tag{3.3}$$

where $n$ denotes an additive white Gaussian noise and $\mathbf{H}$ is the mm-Wave channel representation, which is based on the Saleh-Valenzuela (SV) model, i.e., the channel matrix $\mathbf{H}$ consists of the contributions of a number of $N_{cl}$ scattering clusters each of which includes $N_{ray}$ propagation paths. This channel is expressed as,

$$\mathbf{H} = \gamma \sum_{i,j}^{N_{cl},N_{ray}} \alpha_{i,j} g_r(\Theta_r^{(i,j)}) g_t(\Theta_r^{(i,j)}) \mathbf{a}_r(\Theta_r^{(i,j)}) \mathbf{a}_t^H(\Theta_t^{(i,j)}) \tag{3.4}$$

where $\Theta_r^{(ij)} = (\phi_r^{(ij)}, \theta_r^{(ij)})$ and $\Theta_t^{(ij)} = (\phi_t^{(ij)}, \theta_t^{(ij)})$ denotes the angle of arrivals and angle of departures; $\gamma = \sqrt{N_t N_{rs}/N_c N_{ray}}$ denotes the normalization factor; $\alpha_{ij}$ denotes the complex channel gain associated with the $i^{th}$ scattering cluster and $j^{th}$ path; and $g_t(\Theta_r^{(ij)})$ and $g_r(\Theta_r^{(ij)})$ are the antenna element gains for the transmit and receive antennas.

The $\mathbf{a}_r$ and $\mathbf{a}_t$ are the array response. The array response of a uniform planar array is given below as,

$$\mathbf{a}_{UPA}(\Theta) = \frac{1}{\sqrt{N}}[1, ..., e^{jkd(t_x \times sin(\phi) sin(\theta) + r_x \times cos(\theta))}, ..., e^{jkd(T_x \times sin(\phi) sin(\theta) + R_x \times cos(\theta))}] \tag{3.5}$$

The receiver applies the analog and digital combiner on the signal given in 3.3. This can be expressed mathematically as,

$$\hat{\mathbf{y}} = \sqrt{\rho} \mathbf{W}_{BB}^H \mathbf{W}_{RF}^H \mathbf{H} \mathbf{F}_{RF} \mathbf{F}_{BB} \mathbf{s} + \mathbf{W}_{BB}^H \mathbf{W}_{RF}^H \mathbf{n}, \tag{3.6}$$

where $\mathbf{W}_{RF} \in \mathbb{C}^{N_r \times N_r^{RF}}$ denotes the analog combiner, and $\mathbf{W}_{BB} \in \mathbb{C}^{N_s \times N_r^{RF}}$ denotes the baseband combiner matrix.

In the hybrid beamformer, the analog and digital beamformers are optimized to maximize the achievable rate. Assuming the mm-Wave channel, this rate is given by

$$SE = log_2(|\boldsymbol{I}_{N_s} + \frac{\rho}{N_s} \boldsymbol{Z}_n^{-1} \mathbf{W}_{BB}^H \mathbf{W}_{RF}^H \mathbf{H} \times \mathbf{F}_{RF} \mathbf{F}_{BB} \mathbf{F}_{BB}^H \mathbf{F}_{RF}^H \mathbf{H}^H \mathbf{W}_{RF} \mathbf{W}_{BB}|) \tag{3.7}$$

where $\boldsymbol{Z}_n = \sigma_n^2 \mathbf{W}_{BB}^H \mathbf{W}_{RF}^H \mathbf{W}_{RF} \mathbf{W}_{BB} \in \mathbb{C}^{N_s \times N_s}$ is the covariance matrix of the noise.

Designing hybrid beamforming requires finding the optimal combining and precoding matrices to maximize the achievable data rate. The generic hybrid precoder design problem [12] can be formulated as :

$$\begin{aligned} &\min_{\mathbf{F}_{RF},\mathbf{F}_{BB}} \quad \|\mathbf{F}_{opt} - \mathbf{F}_{RF}\mathbf{F}_{BB}\|_F^2 \\ &s.b.t. \quad \|\mathbf{F}_{RF}\mathbf{F}_{BB}\|_F < P_{max}, \\ &\quad and \qquad |(\mathbf{F}_{RF})_{i,j}| = 1 \end{aligned} \tag{3.8}$$

Channel matrix singular-value decomposition (SVD) gives the globally optimal precoding matrix $\boldsymbol{F}_{opt}$ as follows [162],

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^* \tag{3.9}$$

where $\boldsymbol{U}$ is the $N_r \times N_r$ order matrix, $\boldsymbol{S}$ is the $N_r \times N_t$ order singular value matrix, and $\boldsymbol{V}$ is the $N_t \times N_t$ order matrix.

The globally optimal beamforming solution $\boldsymbol{F}_{opt}$ is built using the front $N_s$ columns of right matrix $\boldsymbol{V}$ [162],

$$\boldsymbol{F}_{opt} = \boldsymbol{V}(:, [1 : N_s]) \tag{3.10}$$

where $N_s$ denotes the number of data streams.

Similarly, the combiner

$$\begin{aligned} &\min_{\mathbf{W}_{RF},\mathbf{W}_{BB}} \quad \|\mathbf{W}_{opt} - \mathbf{W}_{RF}\mathbf{W}_{BB}\|_F^2 \\ &s.b.t. \quad \|\mathbf{W}_{RF}\mathbf{W}_{BB}\|_F < P_{max}, \\ &\quad and \qquad |(\mathbf{W}_{RF})_{i,j}| = 1 \end{aligned} \tag{3.11}$$

where $W_{opt}$ is built using the front $N_s$ columns of the left matrix of the channel SVD.

$$\boldsymbol{W}_{opt} = \boldsymbol{U}(:, [1 : N_s]) \tag{3.12}$$

## 3.2 Brief Overview on Reinforcement Learning

Reinforcement learning is defined as reinforcing the desired behavior in a given agent, which acts in an environment. Reinforcement learning is built upon the theory of Markov processes. There are several Markov models, such as the Markov model, Markov decision process, hidden Markov model, and partially observable Markov decision process. Markov models are defined for autonomous systems in which the states are fully observed. Hidden Markov models are defined for autonomous systems in which the states are partially observed. Markov decision process and partially-observable MDPs are defined for a system that is controlled in which the states are fully observed and partially observed, respectively.

### 3.2.1 Markov Decision Processes

Markov Decision Processes (MDPs) are the framework upon which the theory of Q-learning and Q deep learning is derived. MDPs can be defined as a discrete-time stochastic control process. MDPs are built based upon an agent and an environment. The environment denotes the framework upon which the agent acts. The environment is fixed, and it delivers the next states and its associated reward. The agent has to maximize the sum of the future rewards, known as the return, or more specifically, the expected sum of the future rewards, known as the value function, as the agent does not know what the future rewards are. The agent has to iterate for a maximum iteration to gain experience to reinforce the desired behavior.

Each iteration is called, episode, and each episode ends when the agent lands in the terminal state, a state at which the episode ends. In order to map the states to the actions, we define a policy. Thus, the policy can be defined as a mapping function between states and actions. The policy can be deterministic as it can be probabilistic. It can be a function, an equation, or a neural network. Given a policy, the agent has to maximize the value function. One way that enables the calculation of the value function is the Bellman equations. MDPs work under the assumption that the transition probability from one state to another depends only on the previous state; this is known as the Markov Property or the first-order Markov assumption.

Deep reinforcement learning is an attractive active research area with many applications in robotic control and games. Recently, deep reinforcement learning has found its way to wireless communication and has been applied successfully in many aspects of wireless communications, including jamming detection and mitigation. Deep reinforcement learning is based on the theory of the Markov decision process and deep learning. At first, we present the theory underlying reinforcement learning, then present the theory of deep learning. Reinforcement learning can be defined as "the science of learning decision-making by interactions". Reinforcement learning is based on the Markov decision process (MDP). In reinforcement learning, the MDP is defined as an agent that performs certain actions to move the environment to a new state and receive a reward. The decisions are Markov as the current state includes the history of previous states. The mapping between the states and actions is called policy. The agent's goal in reinforcement learning is to maximize the expected sum of future rewards, known as the value function or the action-value function, while actively generating experiences to enhance online learning. An MDP is "solved" when the optimal policy is determined. For every MDP, an optimal policy exists, which is better than or equal to other policies. Reinforcement learning can be classified into two main cate-

gories: model-based and model-free, also known as sample-based RL, itself can be classified into value-based, policy-based, and actor-critic-based.

### 3.2.2 Model-based RL

In model-based reinforcement learning, the agent learns the model that governs the environment. Dynamic programming can be used to compute the optimal policies given a perfect model of the environment. Dynamic programming, DP for short, consists of two steps: policy evaluation and policy improvement. Policy evaluation turns the Bellman equation to an update to find the values. The policy evaluation convergence is guaranteed as long as the discount factor is less than 1. Policy improvement consists of selecting a new policy (greedy policy), and it follows the policy improvement theorem. Thus, solving the MDP consists of an iterative process, which is known as policy iteration. Evaluating, improving, evaluating, improving, etc., until converging to the optimal policy. DP is used when there is perfect knowledge about the model. However, model-based presents some limitations, such as the capture of irrelevant details as the agent tends to learn everything about the environment, which may not be necessary to find the optimal solution.

### 3.2.3 Model-free RL

Model-free RL can learn without knowledge of the MDPs and use experience samples to learn. Model-free can be classified into two main categories. Value-based, policy-based, and actor-critic-based. In value-based, the agent aims at finding the optimal value function and from which the optimal policy is derived. Values-based methods include MC, TD, Q-learning, and SARSA. For instance, Monte Carlo uses episodes samples to learn. In other words, in MC, the model averages the reward samples for each action. In temporal difference (TD) learning,

the model samples and bootstraps for each action. TD has many advantages over MC as this latter learns only after each episode ends, while TD can learn even before knowing the final outcome as TD learns online at every step and from incomplete sequences. However, TD target is a biased estimate of the values as it sample and bootstrap while MC target is the return which is an unbiased estimate of the values. Temporal difference learning has much lower variance as TD target depends on random action, transition, reward. At the same time, the MC target is the return, which depends on many random actions, transitions, rewards. Some other forms of TD learning are SARSA and Q-learning, which have the advantage of selecting actions without a model of the MDP. An important distinction here is that there are two types of RL: On and off-policy. In the on-policy setting (e.g., the traditional TD and SARSA algorithms), RL learns the value of the policy that is following. In the off-policy setting (e.g., Q-learning algorithm), RL learns the value of a policy different from the one it follows.

### 3.2.4 Policy-based RL

In policy-based, the optimal policy is directly determined without going through the optimal value function. Policy gradient methods are policy-based methods that aim at using gradient ascent to improve the policy. In actor-critic-based RL, both the values and policy are learned. This combines the advantages of value-based and policy-based. For instance, value-based presents the advantage of training with replay experience, and policy-based presents one advantage: the ability to capture continuous actions as the policy is continuous. It is worth mentioning that the policy can be either deterministic as it can be stochastic. Examples of actor-critic methods are advantage actor-critic (A2C) and Soft actor-critic (SAC).

### 3.2.5 Value-based RL

Value-based DRL uses methods such as Q-learning and SARSA to approximate the value function. Estimating the values is known as the prediction. One property of Q-learning is the overestimation of the value function—bad states have high value—. Value-based DRL uses the temporal difference to estimate the Q-value, which uses the Bellman equation to update the estimate of the value function. In the Bellman equation, the estimate of the current state-action pair is based on the next pair. So as the model iterate, the estimation error accumulates and results in overestimation of the Q-value. Value-based methods use double Q-learning, which uses two estimates of the value function to solve the overestimation of the Q-value in DQN. This solution seems to work in discrete space, but it falls short in continuous action space. In continuous action space, exploration in large space becomes computationally expensive and not practical. One way to perform exploration in reinforcement learning with continuous action space is the use of Gaussian distribution.

## 3.3 HBF Transformation Based on RL

In the following, we provide a hybrid beamforming optimization problem formulation using deep reinforcement learning framework under complete channel state information (CSI).

### 3.3.1 Formulation 1

In this formulation, the environment is the mmWave communication system and the agent is the central controller of the base station. At each discrete time step $t$, given a state $s \in S$, the agent selects an action $a \in A$ with respect to a policy $\pi : S \longrightarrow A$. For each action $a$, the agent ends up in a different state $s'$ and collects a reward $r$. The agent's goal is to

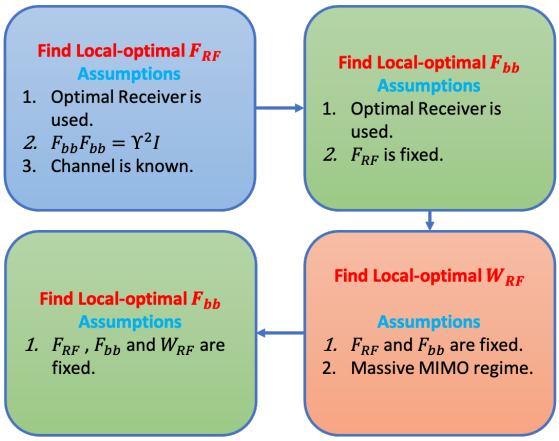


Figure 3.1: Block diagram of HBF based on Deep Learning

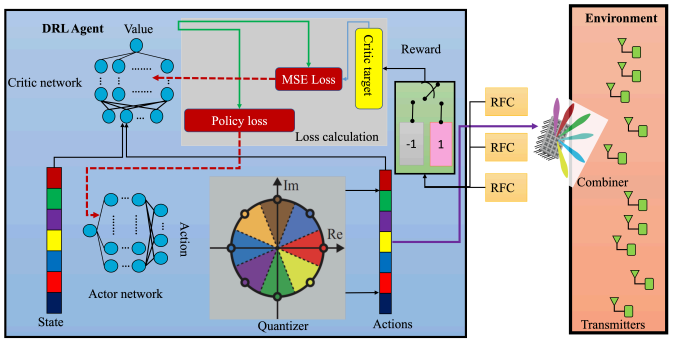


Figure 3.2: DRL-based HBF

maximise the expected sum of the future rewards, known as the return. This objective can be reached by finding the optimal policy $\pi$

For this formulation, to find the analog precoder, we have fixed the baseband precoder. The authors of [163] have shown that regardless of the precoding matrix $\mathbf{F}_{RF}$, the digital precoder typically satisfies $\mathbf{F}_{BB}\mathbf{F}_{BB}^{H} \propto \mathbf{F}$.

**State space:** Let S denotes the hybrid beamforming state space. The current state space consists of the vectorized analog precoder phase matrix.

$$s = vec(\mathbf{F}_{RF}^{(t-1)}) \tag{3.13}$$

where $\mathbf{F}_{RF}^{(t-1)}$ is the analog precoder matrix and $-$ denotes the phases of the matrix' elements.

**Action space:** Let A denotes the hybrid precoding action space. The agent receives the previous state and selects a new set of phases. The action at the current learning iteration can be expressed as,

$$action^{(t)} = vec(\mathbf{F}_{RF}^{(t)}) \tag{3.14}$$

**Transition probability:** Let $P(s' \mid s, action)$ represents the transition probability of transitioning to a new state $s' \in S$, given the action performed in the state $s$

**Reward signal:** In this paper, the objective of DRL based hybrid beamforming design is to maximize the spectral efficiency, therefore we define the reward in terms of the spectral efficiency. If the formed precoder achieves a minimum spectral efficiency, the agent receives a positive reward, otherwise it receives a negative reward.

$$reward = \begin{cases} +1 & R < R^{min} \\ 1 - abs(SE_{opt} - SE)^{0.4} & \text{otherwise.} \end{cases} \tag{3.15}$$

### 3.3.2 Hybrid Beamforming Formulation 2

Given the assumption that the channel is sparse, the authors of [43] have shown that the analog precoder can be restricted on the set of columns of the SVD of the channel matrix.

OMP algorithm shown in Algorithm 3 takes as input the optimal beamformer. The analog precoder is built iteratively wherein each iteration OMP selects the column index with higher correlation with $A^t$. Then, after removing the contribution of the selected column, the algorithm selects the following columns until selecting the final column. This process is done sequentially. The digital precoder is then computed using the pseudo-inverse and normalized.

In our second formulation using reinforcement learning, we aim to select the columns from the candidate matrix $A_t$, which maximizes spectral efficiency. The selection of these columns is made one by one, and each time the agent selects a new column, it ends up in a new state. Therefore,

**Actions:** Are the set of columns from the candidate matrix $A_t$

**States:** The states are numerical state that corresponds to the number of number of selected columns.

**Algorithm 1:** OMP based Hybrid Precoding [164, 165]

**Initialize:** $F_{RF}$ to an empty matrix
**Initialize:** $F_{res} = F_{opt}$
**Data:** The optimal precoder $F_{opt}$
**for** *index is in range (1,* $N_t^{RF}$*)* **do**
    $\Psi = A_t^* F_{res}$
    $k = {}_{l=1,\ldots,N_{cl}N_{ray}} (\Psi\Psi_{l,l}^*)$
                                    `// the column index with high correlation`
    $F_{RF} = [F_{RF}|A_t^{(k)}]$
    `// append the column to the matrix`
    $F_{BB} = (F_{RF}^* F_{RF})^{-1} F_{RF}^* F_{OPT}$
    $F_{res} = \dfrac{F_{opt} - F_{RF}F_{BB}}{||F_{opt} - F_{RF}F_{BB}||_F}$
$F_{BB} = \sqrt{N_s}\dfrac{F_{BB}}{||F_{RF}F_{BB}||_F}$                    `// Normalize the digital precoder`
$F_{BB}, F_{RF}$

In this methodology, the mmWave channel sparsity is considered, and the encoder/combiner can be formulated as a sparse recovery algorithm, which can be solved iteratively. Hence, the action is the index whose column corresponds to the highest correlation of the matrix. Each time a column is extracted, the effect of this column has to be removed, and the agent ends in a new state. Thus, the state corresponds to the state at which the agent is after removing the contribution of the selected columns. In addition, the initial state corresponds to the agent at the first iteration, where no column has been extracted yet, and the terminal

state is the state at which all the matrix columns are extracted.

The choice of the reward function is essential to improve the efficiency of the agent. An appropriate reward function can save a big deal of time when it comes to hyperparameters tuning. For instance, the binary reward choice, which is $+1$ for successful action and $-1$ for failed action, often leads the model to be stuck in the exploration phase and not get to the exploitation phase. The model starts with a total reward of zero, and if it takes failed action, which is very likely to happen at the start, push the agent to explore new action and accumulate negative reward and keep exploring. Some alternative ways for reward functions are the choice of $+1$ reward for successful actions and minor decay negative function for failed actions to encourage the model to exploit and accumulate positive results before the exploration phase.

### 3.3.3 Hybrid Beamforming MDP Solution

The goal of the learning agent is to search for an optimal policy that maximizes the long-term expected discounted reward.

In order to reinforce the desired behavior of a system, the agent's goal is to maximize the expected sum of the future reward G(t). The sum of future reward is expressed mathematically as,

$$G(t) = R(t+1) + R(t+2) + ... = \sum_{\tau=0}^{\infty} R(T+\tau+1) \tag{3.16}$$

$G(t)$ can be defined also in a recursive manner as

$$G(t) = R(t+1) + G(t+1) \tag{3.17}$$

The sum of the future reward,$G(t)$, in some cases, is defined with a discounting factor, which

can be expressed mathematically as

$$G(t) = R(t+1) + \gamma G(t+1) \tag{3.18}$$

where $\gamma$ denotes the discounting factor whose value ranges between 0 and 1. Since, the value function is defined as the expected sum of the future rewards, then, the value function for a given policy at the state $s$ is defined as:

$$V_\pi(s) = \mathbb{E}_\pi[G(t)|S_t = s] \tag{3.19}$$

or

$$V_\pi(s) = \mathbb{E}_\pi[R(t+1) + \gamma G(t+1)|S_t = s] \tag{3.20}$$

The state-value function can be expressed using the bellman's equation as

$$V_\pi(s) = \mathbb{E}_\pi[R(t+1) + \gamma V_\pi(s')|S_t = s] \tag{3.21}$$

or

$$V_\pi(s) = \sum_{s'} \sum_{r} \sum_{a} \pi(a|s) p(s', r|s, a)[R(t+1) + \gamma V_\pi(s')] \tag{3.22}$$

$$V_\pi(s) = \sum_{a} \pi(a|s) \sum_{s'} \sum_{r} p(s', r|s, a)[R(t+1) + \gamma V_\pi(s')] \tag{3.23}$$

The action-value function $Q_\pi(s, a)$ can be defined as

$$Q_\pi(s, a) = \mathbb{E}_\pi[R(t+1) + \gamma G(t+1)|S_t = s, A_t = a] \tag{3.24}$$

$$Q_\pi(s,a) = \mathbb{E}_\pi[G(t)|S_t = s, A_t = a] \tag{3.25}$$

The bellman's equation for action-value function is given by

$$Q_\pi(s,a) = \mathbb{E}_\pi[G(t)|S_t = s, A_t = a] \tag{3.26}$$

$$= \mathbb{E}_\pi[R(t+1) + \gamma \sum_{a'} \pi(a'|s') Q_\pi(s',a')|S_t = s, A_t = a] \tag{3.27}$$

The state-value and action-value functions are related using the following equation

$$V_\pi(s) = \sum_a \pi(a|s) Q_\pi(s,a) \tag{3.28}$$

The state value function $V_\pi(s)$ is useful for evaluating a policy $\pi$ and the action-value function $Q_\pi(s,a)$ is useful for control. In other words, given a state $s$ and two actions $a_1$ and $a_2$, the agent can compare action-value function $Q_\pi(s,a_1)$ and action-value function $Q_\pi(s,a_2)$ to decide which action is better, which corresponds to the action with highest reward.

The optimal policy and optimal value function: The optimal value function is given by

$$V^*(s) = \max_\pi V_\pi(s), \quad \forall s \in S \tag{3.29}$$

The optimal policy is given by

$$\pi^*(s) = \arg\max_\pi V_\pi(s), \quad \forall s \in S \tag{3.30}$$

The optimal action-value function is given by

$$Q^*(s,a) = \max_{\pi} Q_\pi(s,a), \quad \forall s \in S \; \forall a \in A \tag{3.31}$$

**Standard Q-learning**

Standard Q-learning [166], after taking an action $a_t$ in state $s_t$ and observing the associated return $R_{t+1}$ and resulting a new state $s_{t+1}$, updates the parameters as

$$\theta_{t+1} = \theta_t + \alpha(T_t^{DQN} - Q(S_{t+1}, a; \theta_t))\Delta_{\theta_t} Q(S_{t+1}, a; \theta_t) \tag{3.32}$$

where the $\alpha$ is the step size and the target $Target_t^Q$ is given by,

$$T_t^{DQN} = R_{t+1} + \gamma max_a Q(S_{t+1}, a; \theta_t) \tag{3.33}$$

**DQN**

Deep Q-learning or double deep Q-learning uses deep learning to estimate the Q-values. It takes as input the state and it generates a vector of action values $Q_(state, ., \theta)$ where $\theta$ are the parameters of the neural network. In DQN, the system uses a target network with parameters $\theta_t^-$ which are the parameters of the online neural network [166]. The target used by DQN is then as

$$T_t^{DQN} = R_{t+1} + \gamma max_a Q(S_{t+1}, a; \theta_t^-) \tag{3.34}$$

the weights are updated as:

$$\theta_{t+1} = \theta_t + \alpha(T_t^{DQN} - Q(S_{t+1}, a; \theta_t))\Delta_{\theta_t} Q(S_{t+1}, a; \theta_t) \tag{3.35}$$

**Algorithm 2:** Deep Q-learning [164, 165]

**Initialize:** Target network Q
**Initialize:** Replay buffer R
**for** *each env step* **do**
    `// sampling phase`
    Observe a state $s_t$ and select an action $a_t$ using a policy $\epsilon$-greedy
    Execute $a_t$ and observe the next state $s_{t+1}$ and the reward $r_t$
    Store the tuple $\{s_t, a_t, r_t, s_{t+1}\}$ in replay buffer
    **for** *each update step* **do**
        `// Normalize the digital precoder`
        Sample a minibach of N tuples from R
        **if** *Done is True* **then**
            $\underline{y}_i = r_i$
        **else**
            $$y_i = r_i + \gamma \max_{a'} \hat{Q}(s_i', a')$$
            `/* Now this is a for loop */`
        Compute the loss $L = \frac{1}{N}\sum_i (Q(s_i, a_i) - y_i)^2$
        Update Q using Gradient Descent
        $$\theta' \leftarrow \theta$$

### Double Q-Learning

Double Q-learning uses two deep neural networks to estimate the optimal action value given a state $s$. The parameters are updated as:

$$\theta_{t+1} = \theta_t + \alpha(T_t^{DDQN} - Q(S_{t+1}, a; \theta_t))\Delta_{\theta_t} Q(S_{t+1}, a; \theta_t) \tag{3.36}$$

where the target is given by

$$T_t^{DDQN} = R_{t+1} + \gamma \times Q(S_{t+1}, Q(s_{t+1}, a; \theta_t); \theta_t') \tag{3.37}$$

**Algorithm 3:** Double Deep Q-learning [164, 165]

**Initialize:** a primary network $Q_\theta$ and a target network $Q_{\theta'}$, replay buffer R, $\tau << 1$
**for** *each environment step* **do**
    Observe the state $s_t$ and select $a_t \sim \pi(a_t, s_t)$
    execute $a_t$ and observe the next state $s_{t+1}$ and the reward $r_t$
    Store the tuple $\{s_t, a_t, r_t, s_{t+1}\}$ in replay buffer R
    **for** *each update step* **do**
        Sample a minibach $e_t = \{s_t, a_t, r_t, s_{t+1}\} \sim R$
        Compute target Q value:
        $Q() \sim r_t + \gamma Q_\theta(s_{t+1,a'} Q_{\theta'}(s_{t+1}, a'))$
        Perform gradient step on $(Q(s_t, a_t) - Q_\theta(s_t, a_t))^2$
        Update target network parameters:
        Update target network parameters:
        $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$;

### Soft-Actor Critic-Based Hybrid Beamforming [167]

Overestimation of the Q-value abides also in actor-critic methods as well. One may think that this problem can be solved using double deep Q-learning, but the authors of [168] showed that it does not. Therefore, SAC has been proposed to overcome the overestimation

bias in continuous action space as SAC augments DDPG to any stochastic policy [167]. SAC combines off-policy actor critic training with a stochastic actor; in fact it augments the reward with a general maximum entropy. This reward is given as,

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t) + \alpha \mathbf{H}(\pi(. \mid \mathbf{s_t}))] \tag{3.38}$$

where $\alpha$ is the temperature parameter and $\mathbf{H}$ denotes the entropy. In soft actor critic, the soft value function is trained to minimize the squared residual error and mathematically can be expressed as,

$$\begin{aligned} J_v(\psi) =& \mathbb{E}_{s_t\sim D}[1/2(V_\psi(s_t) - \mathbb{E}_{a_t\sim\pi_\phi} \\ & [Q_\theta(s_t,a_t) - log\pi_\phi(a_t \mid s_t)])^2] \end{aligned} \tag{3.39}$$

where $D$ denotes the distribution of the previous sampled states and actions.

The gradient of the $J_v(\psi)$ given in previous equation can be estimated using an unbiased estimator,

$$\begin{aligned} \widehat{\nabla}_\psi J_v(\psi) =& \nabla_\psi \nabla_\psi(s_t)(V_\psi(s_t) - Q_\theta(s_t,a_t) + \\ & log\pi_\phi(a_t \mid s_t)) \end{aligned} \tag{3.40}$$

The soft Q-function can be trained to minimize the bellman resisual given by,

$$J_Q(\theta) = \mathbb{E}_{s_t\sim D}[(1/2(Q_\theta(s_t,a_t) - \widehat{Q}_(s_t,a_t))^2] \tag{3.41}$$

with

$$\widehat{Q}_(s_t,a_t) = r(s_t,a_t) + \gamma \mathbb{E}_{s_{t+1}\sim\rho_\pi}[V_{\widehat{\psi}}(S_{t+1})] \tag{3.42}$$

which can be optimized using stochastic gradients as,

$$\nabla_\theta J_Q(\theta) = \nabla_\theta Q_\theta(s_t, a_t)(Q_\theta(s_t, a_t))r(s_t, a_t) + \gamma V_{\widehat{\psi}}(S_{t+1}) \tag{3.43}$$

The update makes use of a target value $V_{\widehat{\psi}}$

The policy parameters can be learned by minimzing the expected KL-divergence

$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[D_{KL}(\pi_\phi(. \mid s_t)\|\frac{exp(Q_\theta(s_t, .))}{Z_\theta(s_t)})] \tag{3.44}$$

The policy is parameterized using a neural network transformation

$$a_t = f_\phi(\epsilon_t; s_t) \tag{3.45}$$

where $\epsilon_t$ is a noise vector sampled from a fixed distribution. Given this parametrized policy, the objective can be rewritten as

$$J_{\pi(}\phi) = \mathbb{E}_{s_t\sim D,\epsilon\sim N}[((log\pi_\phi(f_\phi(\epsilon_t; s_t)) - \widehat{Q}_(s_t, a_t))^2] \tag{3.46}$$

where $\pi_(\phi)$ is defined in terms of $f_\phi$ Finally, the gradient can be approximated with

$$\begin{aligned}\widehat{\nabla}_\phi J_\pi(\phi) =& \nabla_\phi log\ \pi_\phi(a_t \mid s_t) + (\nabla_{a_t}\ log\pi_\phi(a_t \mid s_t) \\ &- \nabla_{a_t} Q(s_t, a_t))\nabla_\phi(f_\phi(\epsilon_t; s_t))\end{aligned} \tag{3.47}$$

SAC extends DDPG to any tractable stochastic policy.

**Algorithm 4:** DDPG

**Initialize:** the critic network $Q(s, a \mid \theta^Q)$ and the actor $\mu(s \mid \theta^\mu)$ with random weights $\theta^Q$ $\theta^\mu$

**Initialize:** target network $Q'$ and $\mu'$ with weights $\theta^{Q'} \leftarrow \theta^Q$ $\theta^{\mu'} \leftarrow \theta^\mu$

**initialize** replay buffer $R$

**for** *episode is in range (1, M)* **do**

  Initialize a random process N for action exploration

  Receive initial observation state $s_1$

  **for** *t in range(1,T)* **do**

    Select an action $a_t = \mu(s_t \mid \theta^\mu) + N_t$ according to the current policy and exploration noise

    Execute action $a_t$ and observe the reward $r_t$ and observe new state $s_{t+1}$

    Store tuple $\{s_t, a_t, r_t, s_{t+1}\}$ in replay buffer

    Sample a random minibach of N tuples $\{s_j, a_j, r_j, s_{j+1}\}$ from buffer

    Set $y_j$ to $r_j + \gamma Q'(s_{j+1}, \mu'(s_{j+1} \mid theta^{\mu'}) \mid \theta^{Q'})$

    Update critic by minimizing the loss: $L = \frac{1}{N}\sum_j (y_j - Q(s_j, a_j \mid \theta^Q))^2$

    Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \sim \frac{1}{N} \sum_j \nabla_a Q(s, a \mid \theta^Q) \mid_{s=s_j, a=\mu(s_j)} \nabla_{\theta^\mu} \mu(s \mid \theta^\mu) \mid_{s_j}$$

    Update the target networks:

    $\theta^{Q'} \leftarrow \tau\theta^Q + (1-\tau)\theta^{Q'}$

    $\theta^{\mu'} \leftarrow \tau\theta^\mu + (1-\tau)\theta^{\mu'}$

### 3.3.4 Proposed Twin Delayed Deep Policy Gradient Learning

The base station central controller agent generates experiences to find the optimal policy $\pi_\phi$, with a parameter $\phi$ which maximizes the expected return $J(\phi) = \mathbb{E}_{s_i \sim p_\pi, a_i \sim \pi}[R_0]$.

Actor-critic methods consist of two models, one called actor and the other one is called critic.

- **Actor:**updates the policy parameters in the direction suggested by the critic.
- **Critic** updates the value function, which can be either state-value or action-value

**Algorithm 5:** Soft actor critic [167]

**Initialization:** the parameter vector $\psi$ , $\hat{\psi}$, $\theta$, $\phi$
**while** *iteration < total number of iterations* **do**
  **while** *not terminal state* **do**
    $a_t \sim \pi(\phi)(a_t \mid s_t)$
    $s_{t+1} \sim p(\phi)(s_{t+1} \mid s_t, at)$
    $D \longleftarrow D \cup \{(s_t, a_t, r(s_t, a_t), s_{t+1})\}$
  **while** *each gradient step is not final step* **do**
    $\psi \longleftarrow \psi - \lambda_v \hat{\nabla}_{\psi} J_v(\psi)$
    $\theta_i \longleftarrow \theta_i - \lambda_Q \hat{\nabla}_{\theta_i} J_Q(\theta_i), \quad i \in \{1, 2\}$
    $\phi \longleftarrow \phi - \lambda_\pi \hat{\nabla}_{\phi} J_\pi(\phi)$
    $\hat{\psi} \longleftarrow \tau\psi + (1 - \tau)\hat{\psi}$;

Deep Deterministic Policy Gradients (DDPG) [169] is a model-free off-policy actor-critic method. DDPG combines DPG with DQN. DQN works in discrete space, and DDPG extends it to continuous space with the actor-critic framework while learning a deterministic policy.

Actor-critic methods updates the policy using a deterministic gradient as follows,

$$\nabla_\phi J(\phi) = \mathbb{E}_{s_i \sim p_\pi}[\nabla_a Q^\pi(\mathbf{s}, \mathbf{a}) \mid_{\mathbf{a}=\pi(\mathbf{s})} \nabla_\phi \pi_\phi \pi_\phi(\mathbf{s})] \tag{3.48}$$

where $Q^\pi(\mathbf{s}, \mathbf{a}) = \mathbb{E}_{s_i \sim p_\pi}[R_t \mid \mathbf{s}, \mathbf{a}]$, the expected return when performing action $\mathbf{a}$ in a state $\mathbf{s}$ following a policy $\pi$, is known as the critic or the value function.

In Q-learning, the critic can be learned using temporal difference learning that is to say,

$$Q^\pi(\mathbf{s}, \mathbf{a}) = r + \gamma \mathbb{E}_{\mathbf{s}', \mathbf{a}'}[Q^\pi(\mathbf{s}', \mathbf{a}')], \mathbf{a}' \sim \pi(\mathbf{s}') \tag{3.49}$$

in deep Q-learning, the critic $Q_\theta(\mathbf{s}, \mathbf{a})$ is parameterized and can be estimated using a deep

neural network. The target is defined as

$$y = r + \gamma Q_{\theta'}(\mathbf{s}', \mathbf{a}'), \mathbf{a}' \sim \pi_{\phi'}(\mathbf{s}') \tag{3.50}$$

where the actions are selected from a target actor network $\pi_{\phi'}$. The weights of the target networks are updated as

$$\theta' \rightarrow \tau\theta + (1 - \tau)\theta' \tag{3.51}$$

where $\tau$ is between 0 and 1.

This update is known as soft target network updates. If $\tau = 1$, then this is known as hard target update.

DDPG is based on Q-learning; therefore it is subject to policy breaking, causing the model to fail when finding the optimal policy. Twin-delayed DDPG [168] uses three different strategies to address this issue: using double Q-learning to solve the issue of overestimation bias, delaying the policy updates, and smoothing the target policy. In this respect, Twin delayed DDPG modifies the target as follows,

$$y = r + \gamma Q_{\theta'}(s', \pi_{\phi'}(s') + \epsilon), \epsilon \sim clip(N(0, \sigma), -c, c) \tag{3.52}$$

**Algorithm 6:** TD 3-based hybrid Beamforming Learning

Hybrid Beamforming Environment Simulator and the system parameters **Initialize:** actor-network and critic-network with random weights
**Initialize** target network with the weights of actor and critic networks
**Initialize** replay memory and mini-batch size MB
**Initialize** Random exploration process
**for** *index is in range (n − games)* **do**
    **Initialize:** phase vector with a random initial state
    **Initialize:** Random exploration process
    **while** *not done* **do**
        Receive an action from actor-critic network with exploration noise given a state
        Execute the action, observe the reward, the updated state, and done
            `// Agent is playing several episodes to generate a training set`
        store the tuple $(s_t, a_t, r_t, s_{t+1})$ in replay buffer
        `/* Agent learning */`
        sample a mini-batch of size MB from replay buffer D
        select action according to policy and add clipped noise
        Compute the target Q-value
        Obtain current Q estimates
        calculate target using 3.52
        Compute critic loss
        Optimize the critic
                `// update the actor critic using delayed policy updates`

        Compute actor loss
        Optimize the actor
        Update the frozen target models
Your output

## 3.4 Illustrative Results and Discussions

### 3.4.1 Simulation Setup, Implementation, and Parameters

We simulated a $64 \times 16$ massive MIMO hybrid beamforming system. In this system, the transmitter has a 64-element square array with 4 RFCs, and the receiver has a 16-element square array with 4 RFCs. In this simulation, we considered a fully connected structure where

each antenna is connected to all the RFCs. Therefore, each antenna element is connected to 4 phase shifters. To simulate the wireless channel, we have assumed a scattering environment. We randomly distributed in space six scattering clusters. Within each cluster, we placed eight closely located scatterers with a spread angle of 5 degrees, resulting in a total of 48 scatterers. The path gain for each scatterer follows a normal distribution. For each SNR value, we have considered Monte Carlo simulation with 50 trials for each value of SNR. These simulation parameters are summarized in Table 3.1

Table 3.1: Simulation parameters

| **Simulation parameter** | **Value** |
|---|---|
| No. of transmit antenna | 64, 256 |
| No. of receive antenna | 16, 64 |
| No. of RF chain | 4 |
| No. of scatterers | 48 |
| Frequency | 28 GHz |
| No. of episodes | 1000 |
| No. of hidden layers | 2 |
| No. of neurons in layers | 400 , 300 |
| Learning rate $\alpha$ | 0.001 |
| Learning rate $\beta$ | 0.001 |
| TAU | 0.005 |
| $N_{update}$ | 10 |

Considering this simulation setup, Fig. 3.3 shows that 3D pattern, which indicate that there are only a few dominant paths even in multipath environment

### Experience Replay

Experience Replay saves the agent's experiences in memory (state, reward, action, next state), and from which batches of experiences are randomly sampled to be used for training the neural network. Thus, one can states that deep reinforcement learning with Experience Replay consists of two phases: gaining experience and updating the model. The size of the

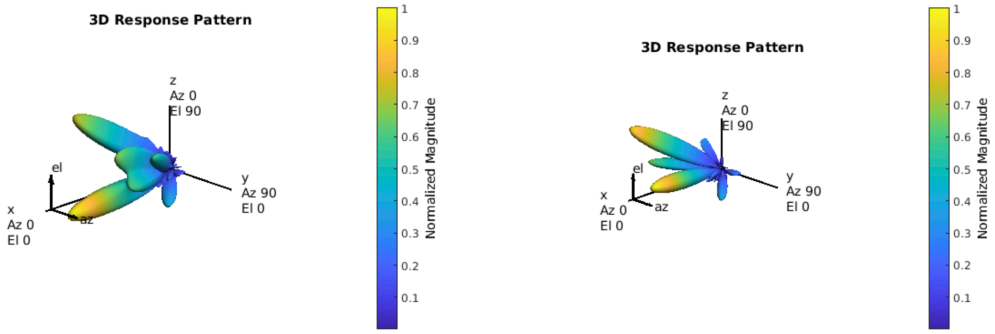


Figure 3.3: 3D Response Pattern

replay controls how many experiences can be used for updating the network. Experience Replay improves the performance of deep learning and helps it to converge earlier.

Another issue of deep Q-learning is the tendency to overestimate the reward, which often yields instability in training and poor policy. This inherently results from the systematic overestimation that introduces a maximization bias. In fact, Q-learning involves learning policy estimations from estimates, and as the model evolves, this overestimation can be a problem.

To solve this issue, double Q-learning has been proposed by Hasselt et al. in [166] which define a new target network, and the Q values are estimated using this new model, which is different from the original network as it is updated after a certain number of episodes. This can result in slowing the training, but it leads to robust performance over time.

### Soft Target Network Updates

Soft Target Network Updates is another method for updating the target network, which has been introduced by the authors of [169]. This method, instead of updating the weights after a certain number of steps, incrementally updates the target network after every run using the formula,

$$\theta_{t+1} = \theta_t \times (1 - \tau) + \theta_{t+1} \times \tau \tag{3.53}$$

where $0 < \tau < 1$

### 3.4.2 Illustrative Results and Discussions

Examples of the obtained results are shown in Figs.3.4 through **??**. Figs. 3.4 through 3.6 shows the reward as a function of number of episodes for simple DQN, double DQN, and double DQN with replay experience and soft target network updates, respectively. To obtain these results, we have set the number of iterations to 6000, the learning rate to 0.001, the replay size to 10 and number of updates to 10. It can be seen from Fig. 3.4 that DQN reward is an increasing function when the number of episodes is less than 1500, and for a number of episodes between 1500 and 4000, the reward decreases suddenly (agent is collecting negative rewards), and only after 4000 episodes, the reward of DQN is increasing again (collecting positive rewards). In general, it can be said that DQN training is unstable and not robust. Fig. 3.5 shows the reward of double DQN with hard target network updates increasing as the number of episodes increases but, around 2500, the model suffers from a sudden decrease (the agent is collecting only negative rewards). DDQN recovers (collect positive rewards) but only after several iterations. It can be said that double DQN is more stable compared with DQN. From Fig. 3.6, it can be observed that DDQN with replay experience and soft target network updates is more stable compared with DDQN with hard target update and simple DQN, as the model recovers faster whenever it obtains negative rewards.

Fig. 3.7 shows the spectral efficiency (bits/S/Hz) as a function of the signal-to-noise ratio (SNR) in $dB$ for optimal beamforming, OMP, and double DQN . In this Figure, it can be seen that double DQN has very comparable performance to the optimal beamforming and

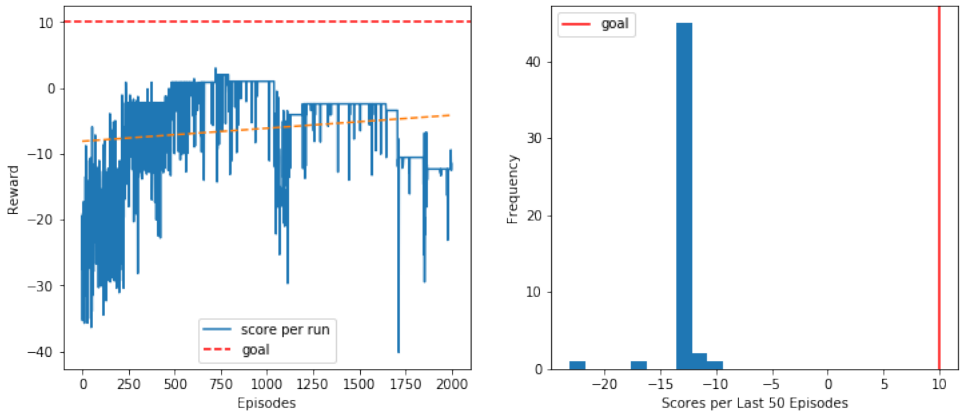


Figure 3.4: Learning curve of DQN

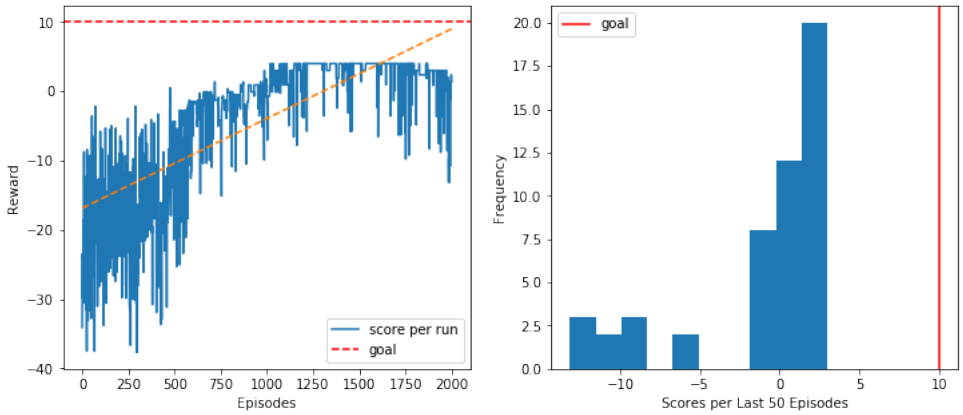


Figure 3.5: Learning curve of DDQN

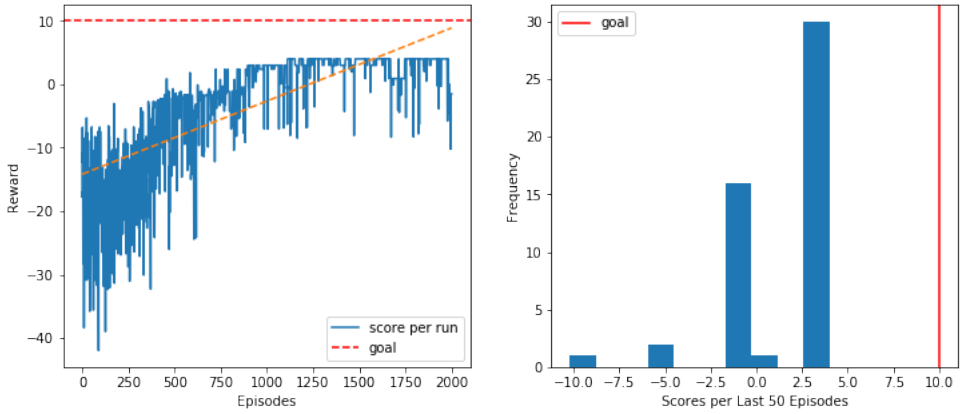


Figure 3.6: Learning curve of DDQN-RST

also to OMP . It can also be seen that the spectral efficiency increases as the signal-to-noise ratio increases for all three methods. One can see that the spectral efficiency increases as the number of streams increases. One can see also that the spectral efficiency achieved by double D-learning with replay experience and soft target update can achieve near-optimal spectral efficiency for all three data streams.

Fig. 3.8 indicates the beam pattern for optimal beamforming, OMP, and double DQN-RST. It can be seen that the main lobe overlap with the optimal one.

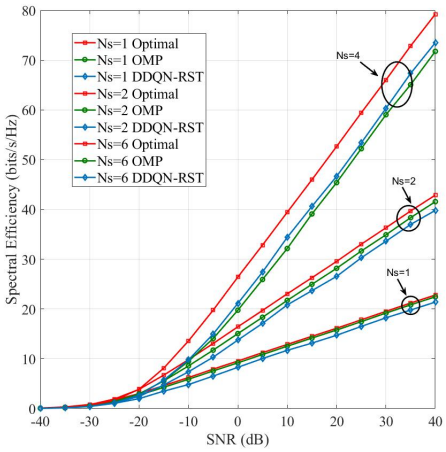


Figure 3.7: Spectral efficiency versus SNR

Fig. 3.11 depicts the spectral efficiency versus SNR of three algorithms SAC-HBF, optimal beamforming, and OMP-based HBF for three values of number of streams $N_s = 1$, $N_s = 2$ and $N_s = 4$. From this figure, it can be seen that SAC-HBF achieves a higher spectral efficiency compared with OMP-based hybrid beamforming and approaching unconstrained SVD-based HBF. One trained model is applied to generate the precoding matrices for all three values of the number of streams. This means that the model is able to generalize the optimal policy to find the analog precoding matrix. Also, it can be seen that SAC-HB performs well when the number of data streams is equal to the number of RF chains, which means that we can transmit a higher number of data streams using this technique.

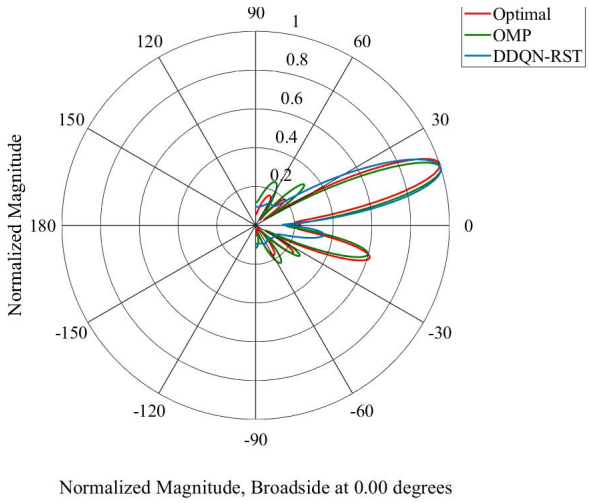


Figure 3.8: Beam Pattern of Three algorithms OMP, Optimal, and DDQN-RST

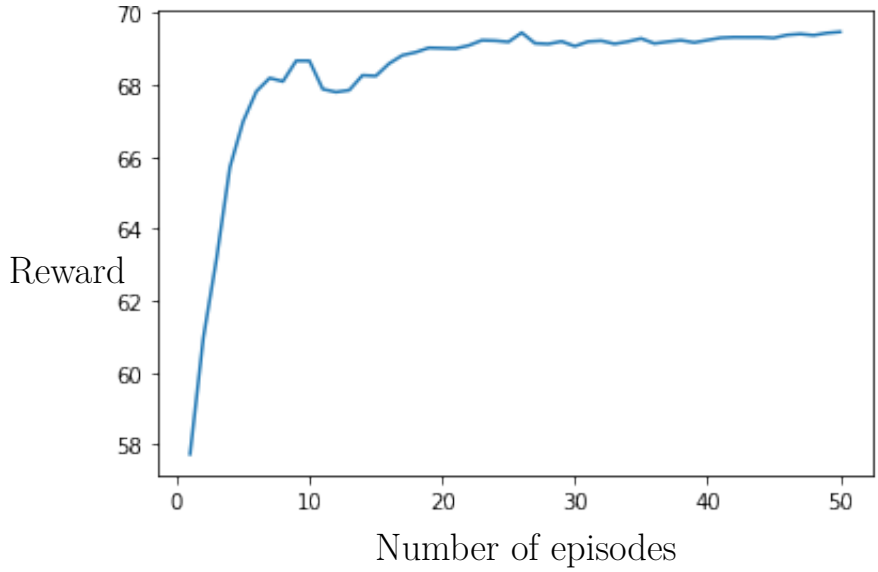


Figure 3.9: Learning Curve of Soft Actor Critic with two hidden layers of 300 and 200 nodes

Fig. 3.10 shows the beam pattern generated by SAC-based hybrid beamforming, optimal beamforming, and OMP. One can see that the beam pattern generated using SAC approaches that one of optimal beamforming and orthogonal matching pursuit. These results are in favor of the results shown in Fig. 3.11

Examples of the obtained results are presented in Figs.3.12 through 3.14. Fig. 3.12 illustrates

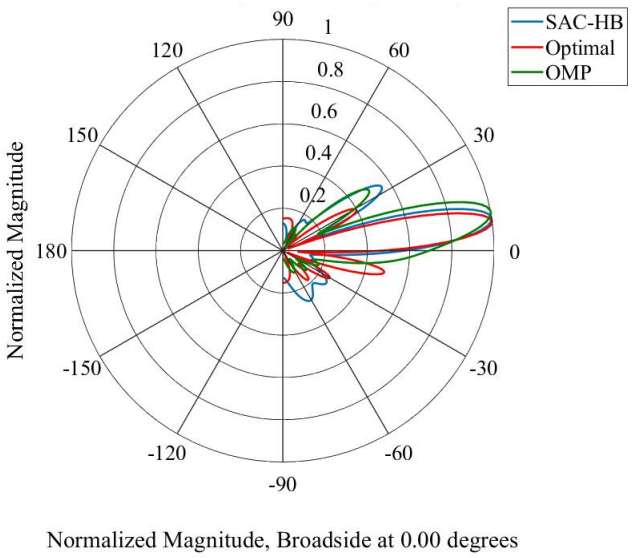


Figure 3.10: Beam patterns ($64 \times 64$ MIMO) Azimuth Cut (elevation angle= 0.0)

the learning curve of the twin delayed deep deterministic policy gradient method. The learning rate is set to 0.001 for $\alpha$, and 0.001 for $\beta$, 0.005 for $\tau$, and the number of hidden layers is set to two hidden layers of 400 and 300 units each. The number of episodes is 1000. This figure indicates that the learning curve is smooth and exponentially increasing for a number of episodes between 1 and 100 episodes. The agent begins by collecting negative rewards in the early stage of the training; however, as the number of episodes increases, the agent learns the optimal policy, resulting in the collection of positive rewards. The performance of the TD-3 agent is stable and scores about 70 when the iteration number is higher than 100. This graph indicates that the learning efficiency is high, as the agent did not need many episodes to learn the optimal policy, demonstrating the convergence speed of the proposed model.

Fig. 3.13 depicts the system spectral efficiency (bits/S/Hz) as a function of the signal-to-noise ratio (SNR) in $dB$ for optimal beamforming, OMP, and TD3-based hybrid beamforming in SU-MIMO system. This figure indicates that TD3 performs closely to optimal beam-

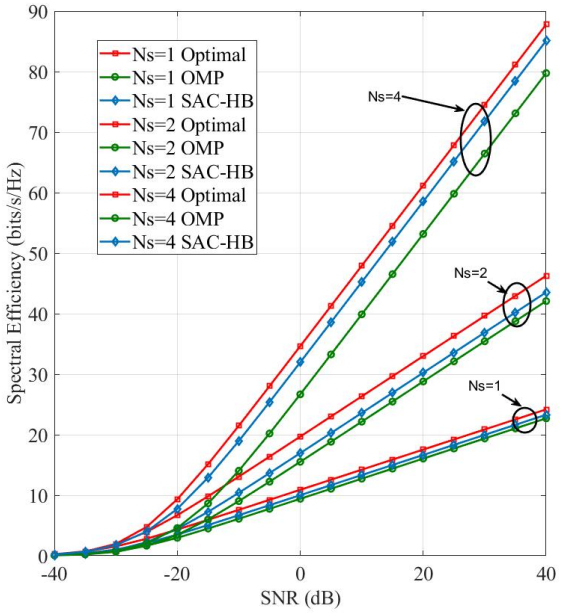


Figure 3.11: Learning Curve of Soft Actor Critic with a two hidden layers of 300 and 200 nodes

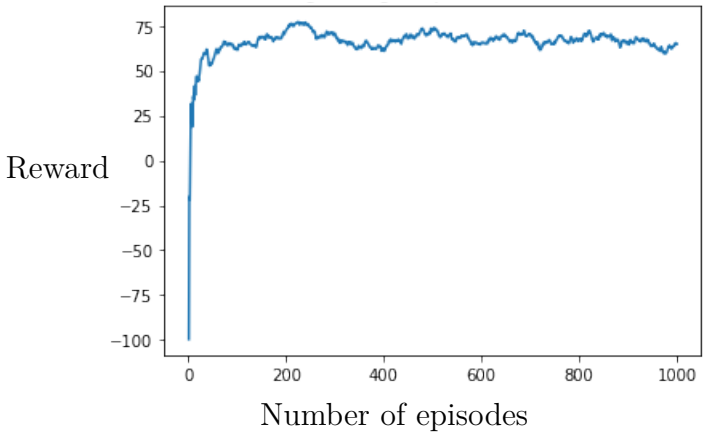


Figure 3.12: Learning curve TD3

forming and outperforms OMP in all three cases ($N_s = 1$, $N_s = 2$, and $N_s = 4$). The spectral efficiency increases as SNR increases for all three methods, and the spectral efficiency is an increasing function of the signal-to-noise ratio and the number of data streams. The spectral efficiency achieved by TD3 for the three values of the number of data streams approaches

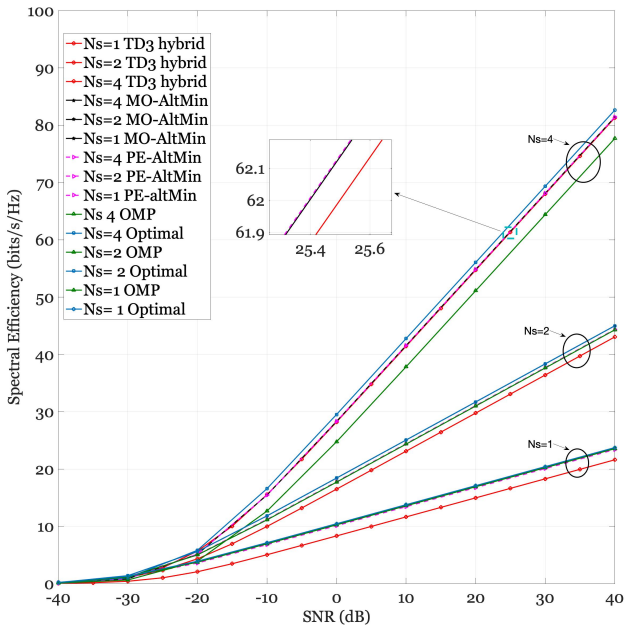


Figure 3.13: Spectral efficiency versus SNR ($64 \times 64$ MIMO) (Lr=0.001, two hidden layers (400, 300) and trained on 1000 episodes with $\tau = 0.005$) tested under incomplete channel

near-optimal spectral efficiency. The performance of OMP decreases as the number of data streams is equal to the number of RFCs.

Fig. 3.14 illustrates the spectral efficiency versus SNR for TD3-HB, optimal beamforming, and OMP based hybrid beamforming for three data stream values: $N_s = 1$, $N_s = 2$ and $N_s = 4$. TD-3 achieves a higher spectral efficiency compared to OMP-based hybrid beamforming. This spectral efficiency approaches that of optimal hybrid beamforming. One trained model is applied to generate the precoding matrices for all three values of the number of streams. This means that the model can generalize the optimal policy to find the analog precoding matrix. Also, it can be seen that TD3-HB performs well when the number of data streams is equal to the number of RF chains, which means that we can transmit a higher number of data streams using this technique.

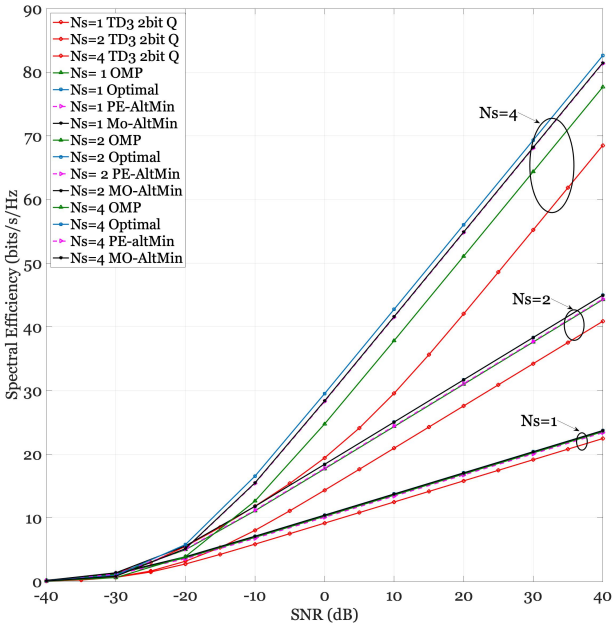


Figure 3.14: Spectral efficiency when the phase shifter is quantized with 2 bits ($64 \times 64$ MIMO)

## 3.5 Summary

In this chapter, we provided the system model of the SU-MIMO system, and we transformed it using reinforcement learning. Then, we proposed two algorithms to solve hybrid beamforming. The first algorithm uses one important result: there are only a few dominant directions, even in the case of multipath environments. Using a double deep Q-learning algorithm, we showed that reinforcement learning could select the atoms from a candidate matrix to form the analog precoder. The second algorithm uses the soft actor-critic method and forms the analog precoder by selecting the optimal phase for each matrix element. The results indicated that the soft actor-critic method could form the analog precoder that achieves a comparable performance to optimal beamforming. Both of the algorithms assumed that the channel was perfectly known.

# Chapter 4

# Compressed Sensing for mmWave Communications

The still-exorbitant energy consumption of the analog-to-digital converters (ADCs) at high frequencies makes the use of fully digital beamforming with large-scale- antenna arrays unfeasible. On the flip side, the mmWave channels exhibit high sparsity levels due to the limited number of dominant propagation paths; therefore, compressive sensing techniques can be leveraged to perform low-cost beam training and channel estimation. This chapter shows that the mmWave channel estimation can be formulated as a sparse channel recovery problem, where the channel measurement matrix can be constructed from the precoding and combining matrices. This sparse recovery problem can be solved using several techniques such as greedy methods, Bayesian methods, or convex and relaxation methods. We propose a performance comparison between these algorithms to find the optimal channel reconstruction technique. Also, we provide a performance comparison between different measurement matrices to select the most appropriate one for designing a low-cost beamformer. Most channel estimation methods adopt random channel measurement matrices, which means that the sensing power is distributed in all spatial directions. Compressive sensing techniques can efficiently sense and reconstruct the mmWave channel while requiring a few training pilots compared with exhaustive beam training and channel estimation techniques.

## 4.1 Channel Estimation

Channel estimation and beamforming algorithms in hybrid architecture are more complex compared to the ones in fully digital systems because of the small number of RFCs and the constraints imposed by the phase shifters [170]. The limited number of RFCs makes it challenging to estimate the channel because the receiver has only access to a few measurements, mainly that the ADCs have limited quantization bits. At the same time, the unit modulus constraint transforms the design of hybrid beamforming into a non-convex optimization problem.

Hybrid beamforming requires accurate channel state information, which can be obtained through the channel estimation process. However, it is challenging to obtain perfect knowledge of the mmWave channel given a hybrid architecture. It requires large bandwidth and significant training overhead; therefore, there is a need to reduce this overhead so this time can be used to speed up the hybrid beamforming construction and therefore minimize the communication delay. Developing low complexity channel estimation results in a significant decrease in the dimensions of the channel matrix, which reduces the complexity of the beamformers' design. In addition, rapid channel variations result in estimation delays which in turn can reduce the achievable data rate [170]. Hence, developing low complexity channel estimation methods is highly

To design low-complexity channel estimation in a hybrid architecture, the spatial sparsity of the mmWave channel is one of the main practical assumptions that are made [171, 172]. In this respect, we assume that the mmWave channel is sparse in the angular domain, and if not, there are several directions that can be taken to make this assumption hold—for instance, considering more considerable channel support or performing some sparse representation/projections.

This assumption is important in the sense that the compressive sensing framework deals only with sparse signals. This framework enables the recovery of high dimensional sparse signals from a few measurements [173]. The application of compressive sensing develops low complexity channel estimation and reduces the number of estimated channel parameters [170] and therefore reduces the number of training pilots.

## 4.2 Compressive Sensing

Compressed Sensing (CS) uses the sparsity of signal on a given basis and recovers the signal from a few measurements [173]. These measurements are often obtained using a measurement matrix.

Compressive sensing often involves three main processes: sparse projection, measurement or encoding, and sparse recovery or decoding. The sparse projection consists of projecting a given signal on a suitable basis so it can be sparse. A signal is sparse if and only if a few of its components are non-zeros. Discrete cosine transform (DCT), wavelet transform (WT), and fast Fourier transform (FFT) are the most common sparse projections.

The measurement process consists of acquiring only a few samples from the sparse signal. This acquisition can be completed by multiplying the sparse signal with a measurement matrix. This measurement matrix is required to satisfy some properties such as the restricted isometry property (RIP) or mutual incoherence to ensure the sparse signal recovery from the few samples with a high probability.

Numerous measurement matrices have been proposed, which can be classified into two main categories: random and deterministic. Random matrices are generated following a sub-Gaussian distribution. Examples of random measurement matrices include Gaussian,

Bernoulli, and uniform. On the other hand, deterministic matrices are constructed in a deterministic manner to satisfy the Restricted Isometry Property. Examples of deterministic matrices are Toeplitz, circulant, second-order Reed-Muller, and chirp-based matrix. The recovery process aims at recovering the original sparse signal from a few measurements given the measurement matrix. The sparse recovery problem, which is an undetermined system of linear equations, may be formulated as an optimization problem with constraint. Several sparse recovery techniques have been proposed, and we can distinguish between three types: convex and relaxation, Greedy, and Bayesian techniques. Examples of the first type are Basis Pursuit and Gradient Descent. Examples of the second type are Orthogonal Matching Pursuit and its derivatives. Bayesian via Laplace Prior and Bayesian via Relevance Vector Machine are examples of the last type.

In order to construct good channel estimators, selecting an appropriate measurement matrix and channel recovery technique is crucial; therefore, we have performed a performance comparison between greedy techniques, Bayesian techniques, and convex and relaxation techniques, as well as a performance comparison between several measurement techniques matrices. Next, we show important results that can be used for low-cost and energy-efficient beam training and channel estimation.

## 4.3 System Model

In this model, we consider a base station (BS) with $N_{BS}$ antennas communicating with a mobile station (MS) with $N_{MS}$ antennas. The number of RF chains are $N_{rf} << N_{BS}$. The base station transmits a signal after applying the baseband and analog precoding matrix on

the vector of transmitted symbols $s$.

$$\mathbf{x} = \mathbf{F_T s} + \mathbf{w} \tag{4.1}$$

where $\mathbf{F_T}$ denotes the beamformer, in the case of hybrid beamforming, $\mathbf{F_T}$ is equal to $\mathbf{F_{RF}F_{BB}}$. $s$ is a transmitted signal which consists of the concatenation of several data streams, and $w$ denotes an additive white Gaussian noise.

At the mobile station side, the received signal can be expressed as

$$\mathbf{r} = \mathbf{HF_T s} + \mathbf{w} \tag{4.2}$$

where $\mathbf{H}$ denotes the mmWave channel matrix, $\mathbf{w}$ denotes the Gaussian noise matrix.

On the MS side, after the digital and analog combiners are applied to the observed signal, the received signal can be expressed as

$$\mathbf{r} = \mathbf{W_T^H HF_T s} + \mathbf{W_T^H noise} \tag{4.3}$$

Given that the channel is of limited scattering, we consider a channel of L scatterers, each of which contributes a single propagation path between the transmitter and the receiver. Under this model, the channel can be expressed as

$$\mathbf{H} = \sqrt{\frac{N_t N_r}{\rho}} \sum_{l=1}^{L} \alpha_l \mathbf{a}_{MS}(\theta_l)\mathbf{a}_{BS}^H(\phi_l) \tag{4.4}$$

where $\mathbf{a}_{MS}(\theta)$ and $\mathbf{a}_{BS}(\phi_l)$are the antenna array response vectors at the BS and MS, respectively, $\alpha_l$ is the complex gain of the $l^{th}$ path, and $\rho$ denotes the average pathloss. b

$$\mathbf{a}_{BS}(\phi_l) = \frac{1}{\sqrt{N_{BS}}}[1, e^{j2\pi/\lambda dsin(\phi_l)}, ..., e^{j(N_{BS}-1)2\pi/\lambda dsin(\phi_l)}]^T \tag{4.5}$$

where $\lambda$ is the wavelength, and d is the distance between antenna elements. In the same fashion,

$$\mathbf{a}_{MS}(\theta) = \frac{1}{\sqrt{N_{MS}}}[1, e^{j2\pi/\lambda dsin(\theta_l)}, ..., e^{j(N_{MS}-1)2\pi/\lambda dsin(\theta_l)}]^T \tag{4.6}$$

In more compact format, the channel can be written as

$$\mathbf{H} = \mathbf{A}_{MS} diag(\alpha) \mathbf{A}_{BS}^H \tag{4.7}$$

where $\boldsymbol{\alpha} = \frac{\sqrt{N_{BS}N_{MS}}}{\rho}[\alpha_1, ...\alpha_L]^T$

$$\mathbf{A}_{MS} = [\mathbf{a}_{MS}(\theta_1), \mathbf{a}_{MS}(\theta_2), ..., \mathbf{a}_{MS}(\theta_L)] \tag{4.8}$$

$$\mathbf{A}_{BS} = [\mathbf{a}_{BS}(\phi_1), \mathbf{a}_{BS}(\phi_2), ..., \mathbf{a}_{BS}(\phi_L)] \tag{4.9}$$

In the first step, we assume that the mmWave channel has no prior knowledge of the channel. Thus the base station performs compressive sensing to recover the channel. In the second part, the estimated channel is used to construct the hybrid precoding and decoding matrices.

### 4.3.1 Sparse Formulation of mmWave Channel Estimation Problem

During the training phase,

$$\mathbf{y_{p,q}} = \mathbf{w_q^H H f_p s_p} + \mathbf{W_q^H n_{q,p}} \tag{4.10}$$

if each MS perform the measurements,

$$\mathbf{y_p} = \mathbf{W^H H f_p s_p} + diag(\mathbf{W^H}[\mathbf{n_{1,p}}, ..., \mathbf{n_{M_{MS},p}}]) \tag{4.11}$$

where $\mathbf{W} = [\mathbf{w_1}, \mathbf{w_2}, \mathbf{w_q}, ..., \mathbf{w_{M_{MS}}}]$ is the measurement matrix, which to be formed the MS has to perform successive measurements, $w_q$ is the combining vector at the $q^{th}$ instant.

$$\mathbf{Y} = \mathbf{W^H H F S} + \mathbf{Q} \tag{4.12}$$

$\mathbf{F} = [\mathbf{f_1}, \mathbf{f_2}, \mathbf{f_3}, ..., \mathbf{f_{M_{BS}}}]$ is the beamforming matrix used at the base station, and $\mathbf{Q}$ is the noise matrix.

As all the data stream are equal and with the same power.

$$\mathbf{Y} = \sqrt{P}\mathbf{W^H H F} + \mathbf{Q} \tag{4.13}$$

To exploit the sparsity nature of the mmWave, we vectorize

$$\mathbf{y_v} = vec(\mathbf{Y}) = \sqrt{P}vec(\mathbf{W^H H F}) + vec(\mathbf{Q}) \tag{4.14}$$

using the properties of the Kronecker's product, we can write

$$\mathbf{y_v} = vec(\mathbf{Y}) = \sqrt{P}vec(\mathbf{W^H H F}) + vec(\mathbf{Q}) \tag{4.15}$$

$$\mathbf{y_v} = vec(\mathbf{Y}) = \sqrt{P}(\mathbf{W^H} \otimes \mathbf{F})vec(\mathbf{H}) + vec(\mathbf{Q}) \tag{4.16}$$

$$\mathbf{y_v} = vec(\mathbf{Y}) = \sqrt{P}(\mathbf{F^T} \otimes \mathbf{W^H})(\mathbf{A}^*_{BS} \odot \mathbf{A}_{MS})\alpha + vec(\mathbf{Q}) \tag{4.17}$$

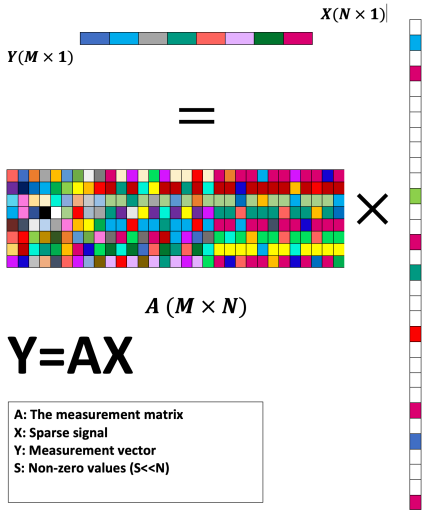


Figure 4.1: Compressive Sensing

where $\odot$ denotes the Khatri-Rao product, and the matrix $\mathbf{A}^*_{BS}()\mathbf{A}_{MS}()$ consists of the columns made of $\mathbf{a}^*_{BS}(\phi_l) \otimes \mathbf{a}_{MS}(\theta_l)$, $l = 1, 2, ..., L$ where each column $l$ represents the Kronecker product of the BS and MS array response vectors associated with the AoA/AoD of the $l^{th}$ path of the channel. Further, the AoAs and AoDs are assumed to be taken from a uniform grid of $N$ points with $N >> L$.

We may write, if the quantization error is negligible,

$$\mathbf{y_v} = \sqrt{P}(\mathbf{F^T} \otimes \mathbf{W^H})\mathbf{A_D}\mathbf{h} + \mathbf{n_Q} \tag{4.18}$$

where $\mathbf{A_D}$ is an $N_{BS}N_{MS} \times N^2$ dictionary matrix that consists of $N^2$ columns $\mathbf{a}^*_{BS}(\phi_u) \otimes \mathbf{a}_{MS}(\theta_v)$ and $\phi_u = 2\pi u/N$ and $\theta_v = 2\pi v/N$. $\mathbf{h}$ denotes an $N^2 \times 1$ path gain vector. The element of this vector corresponds to the quantized directions.

$\mathbf{y_v}$ shown in 4.18 formulates channel estimation as a sparse recovery problem where $\phi = (\mathbf{F^T} \otimes \mathbf{W^H})\mathbf{A_D}$ denotes the sensing matrix, which should guarantee the recovery of the non-zero elements of the vector $z$ with high probability, and with a small number of measure-

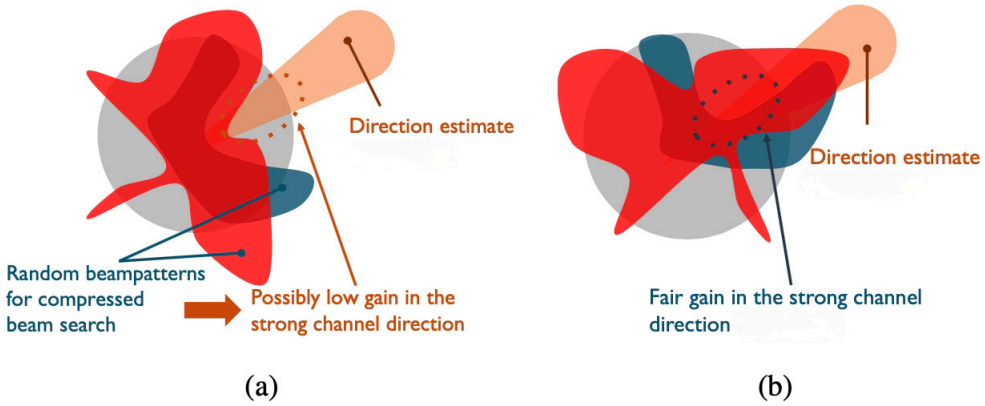


Figure 4.2: Choice of sensing matrix [6] © 2017 IEEE

ments. This sensing matrix can be constructed using random beamformers or structure-based beamformers, which enables the recovery of the sparse channel from a few measurements.

The sparse channel recovery can be formulated as

$$minimize\ ||h_b||_0 \quad subject\ \ to\ \ ||\mathbf{y_v} - \sqrt{P}(\mathbf{F^T} \otimes \mathbf{W^H})\mathbf{A_D}\mathbf{h_b}||_2 \leq \epsilon \tag{4.19}$$

This problem can be relaxed with $l_1$ norm

$$minimize\ ||h_b||_1 \quad subject\ \ to\ \ ||\mathbf{y_v} - \sqrt{P}(\mathbf{F^T} \otimes \mathbf{W^H})\mathbf{A_D}\mathbf{h_b}||_2 \leq \epsilon \tag{4.20}$$

This problem can be solved using greedy, Bayesian, or convex and relaxation techniques given a suitable measurement matrix. The channel estimation can be carried out using several measurement matrices and recovery algorithms discussed next.

## 4.4 Sensing Process

Solving the problem shown in (4.18) is guaranteed as long as the measurement matrix $\phi$ has a small mutual coherence or satisfies RIP of order k.

**Definition 1:** The coherence measures the maximum correlation between any two columns of the measurement matrix $\phi$. If $\phi$ is a $M \times N$ matrix whose normalized column vectors are $\phi_{\mathbf{1}}$,$\phi_{\mathbf{2}}$,$\phi_{\mathbf{3}}$,...,$\phi_{\mathbf{N}}$. Then, the mutual Coherence Constant (MCC) is defined as:

$$\mu(\phi) = \max_{i \neq j} \frac{| < \phi_{\mathbf{i}}, \phi_{\mathbf{j}} > |}{\|\phi_{\mathbf{i}}\|_2 \|\phi_{\mathbf{j}}\|_2} \tag{4.21}$$

Compressive sensing deals with low coherence, which means that a few samples are required to recover the sparse signal fully.

**Definition 2:** A measurement matrix satisfies the Restricted Isometry Property if there exists a constant $\delta_k$ such as:

$$(1 - \delta_k)\|x\|_2^2 \leq \|\phi x\|_2^2 \leq (1 + \delta_k)\|x\|_2^2 \tag{4.22}$$

Where $\|x\|_2$ denotes the $l2 - norm$ and $\delta_k \in [0, 1]$ is called the Restricted Isometry Constant (RIC) of $\phi$ which should be much smaller than 1.

### 4.4.1 Measurement matrices

Measurement matrices can be classified into two categories: Random measurement matrices and deterministic measurement matrices. The first category is generated using a sub-Gaussian distribution and satisfies the restricted isometry property with high probability. Examples of these matrices include Gaussian, Bernoulli, and Uniform. Deterministic

measurement matrices are defined using a deterministic function or structure to reduce the randomness. Examples of these measurement matrices includes Toepliz, Circulant [174], Binary BCH [175], second-order Reed-Solomon, Chirp sensing [176], and quasi-cyclic low-density parity-check code (QC-LDPC) [177] matrices are examples of full-deterministic type matrices.

### Random Measurement Matrices

The entries of random measurement matrices follow an underlying probability distribution. Gaussian, Bernoulli, and uniform matrices have normal, Bernoulli, and uniform distribution, respectively. Random Gaussian and Bernoulli matrices satisfy the RIP with probability at least $1-\epsilon$ given that the sparsity satisfies $K \leq C_1 MLog(N/K) + C_2 log(\epsilon^{-}1)$ where K denotes the channel sparsity, M is the number of measurements, and N is the length of the sparse channel [178].

Some other examples of random measurement matrices are random partial Fourier matrix and random Hadamard matrices. Random Partial Fourier matrix, which consists of choosing random M rows of the Discrete Fourier matrix, satisfies the RIP with a probability of at least $1-\epsilon$ if the number of measurement M satisfies $M \geq CKLog(N/\epsilon)$ [178].

The Hadamard measurement matrix is a matrix whose entries are 1 and -1. The columns of this matrix are orthogonal. The Random Partial Hadamard matrix consists of taking random rows from the Hadamard matrix. This measurement matrix satisfies the RIP with probability at least $(1-5/N-e^{-\epsilon})$ provided $M \geq C_0(1+\beta)Klog(N)$ with $\beta$ and $C_0$ are positive constants, K is the sparsity of the signal, N is its length and M is the number of measurements [179].

**Deterministic Measurement Matrices**

Deterministic measurement matrices are generated following a structure to reduce the randomness, memory storage, and processing time [180]. For instance, circulant and toeplitz matrices are examples of deterministic measurement matrices. Given vector $c = (c_1, c_2, ...c_n) \in R^N$, its associated circulant matrix whose (i,j)-th entry is given by:

$$C_{ij} = c_{j-i} \tag{4.23}$$

Where $i, j = 1, 2, ..., N$.

If a random subset of cardinality $M <= 1$, then the partial constructed circulant matrices satisfies the RIP with high probability given that $N \geq C_\delta K^2 log(N/K)$ where N is the length of the sparse channel and K is the channel sparsity [181].

The Toeplitz matrix is a matrix with a constant diagonal i.e. $T_{ij} = T_{(i+1j+1)}$ [182]. If we randomly select a subset S of columns from this matrix, the Restricted Isometry Constant of the Toeplitz matrix restricted to the rows indexed by the set S satisfies $\delta_K \leq \delta$ with a high probability provided $N \geq C_\delta K^2 log(N/K)$ where $K$ is the sparsity of the signal and $N$ is its length [37].

## 4.5 Reconstruction Methods

Various techniques can be used to recover the original signal from a few measurements. These techniques, as mentioned before, can be mainly grouped into three categories: convex and relaxation methods, greedy, and Bayesian category. In this work, we considered two representative techniques from each category. From the first category, we have considered

basic pursuit and gradient descent methods. From the second category, we have considered orthogonal matching pursuit and iterative hard thresholding methods. From the last category, we have considered Laplace prior and relevance vector machine prior methods.

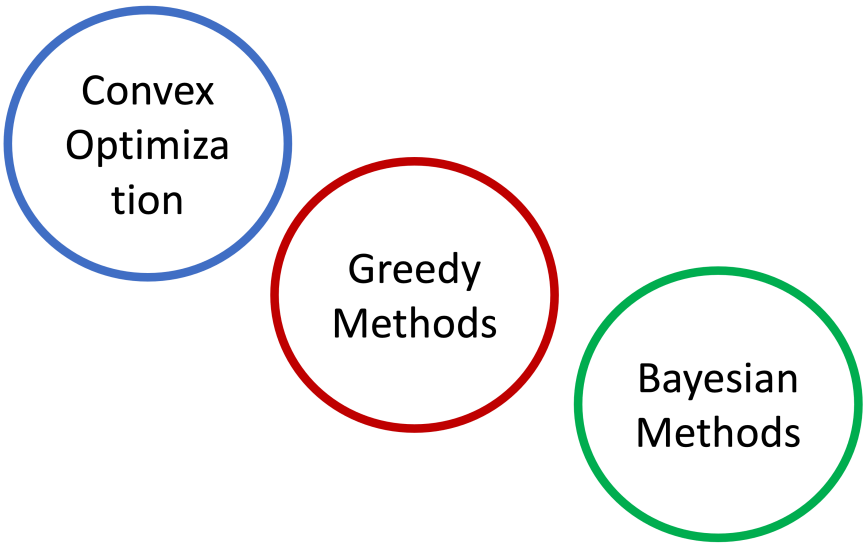


Figure 4.3: Three sparse Recovery Algorithm Categories

### 4.5.1 Convex and Relaxation category

#### Basis Pursuit

Basis pursuit method finds the sparse vector $x$ with the smallest $l_1 - norm$ instead of the $l_0 - norm$; therefore relaxing the non-convex optimization problem [183]. This formulation can be expressed as,

$$\min ||x||_1 \quad subject \quad to \quad \phi x = y \tag{4.24}$$

where $y \in R^M$ denotes the vector of measurements, $\phi_{M \times N}$ denotes the measurement matrix, $x \in R^N$ denotes the sparse signal with length $1 \times N$ where the number of measurement

$M << N$, $|\dot{|}|_1$ denotes the $l_1 - norm$

#### Gradient Descent

Gradient descent solves the sparse recovery problem using an iterative process [184]. This method builds iteratively the sparse signal $x \in R^N$ from a few measurements $y \in R^M$ using the following formula:

$$x = L_s(x + \frac{1}{\gamma}\phi^T r) \tag{4.25}$$

where $\delta_{2s} + 1/3$ is a constant, $\phi^T$ denotes the transpose of measurement matrix, $r$ denotes the residue, which is computed as $r = y - \phi x$, and $L_s$ denotes an operator that keeps only the largest magnitude coordinates and null all the rest.

Gradient descent method initializes the residue $r$ to $y$ , $x$ to zero, and the weight $\gamma$ to 4/3. Then, it updates $x$ using the formula 4.25. This process iterates until reaching a stopping criteria. More details about stopping criteria can be found in [184].

### 4.5.2 Greedy category

#### Orthogonal Matching Pursuit

Orthogonal matching pursuit is a greedy method that builds the sparse signal in several iterations to reduce the recovery complexity [185]. At each iteration, it locates the atom or the column index (from the measurement matrix $\phi$) whose values gives the largest correlation with the residue $r = y - \phi x$ by taking the higher absolute value of the inner product calculated between each column and the residue using the following formula:

$$k = \arg\max_j \phi^T r_j \tag{4.26}$$

Where the $\phi^T$ denotes the transpose of the measurement matrix $\phi$ and $k$ denotes the matrix column's index that has the largest correlation or the atom. Then, it appends the selected atom to the set $S = S\{k$ to build the support of the sparse vector.

OMP then estimates the target variable by solving a least-square problem restricted to the columns in the set $S$, and it nulls all the other components of $x$ using the following formula:

$$(\hat{x}_i)_{|s} = (\phi|s)^{\dagger}.y; (x_i)_{|s^c} = 0 \tag{4.27}$$

Where $s$ denotes the atom set, $s^c$ denotes complement of the set $s$, $\phi^{\dagger}$ denotes the pseudo-inverse of the matrix $\phi$, and $\phi^{\dagger}_{|s}$ is the pseudo-inverse of the matrix $\phi$ restricted to the set $s$. After removing the contribution of the selected atom, OMP updates the residue $r = y - \phi x$ and iterates by selecting a new column. This process repeats until all the atoms are selected.

**Iterative Hard Thresholding [186]**

The iterative hard thresholding method is yet another greedy method that solves the sparse recovery problem iteratively. It iteratively updates the sparse vector $x$ using the following formula:

$$X^{[i+1]} = L_s(x^{[x]} + \phi^T(y - \phi x^{[i]})) \tag{4.28}$$

where $L_s$ denotes a hard thresholding operator that nulls all the largest elements of $x$ in terms of magnitude, $x^{[i]}$ denotes the value of $x$ at the $i^{th}$ iteration, and $\phi^T$ denotes the transpose of the measurement matrix $\phi$.

Iterative Hard thresholding algorithm starts by an initialization of the $x^{[0]} = 0$, then at the iteration $i+1$, it calculates the $x^{[i+1]}$ by calculating 4.28 and applying the operator $L_s$ on $z^i$ and iterates until reaching the stopping conditions [5].

### 4.5.3 Bayesian category

In order to recover the sparse signal $x$ subject to the equation $y = \phi x$, where $y \in R^M$ denotes the observations, $\phi$ denotes the measurement matrix, and $X \in R^N$ denotes the unknown sparse signal with $M << N$, Bayesian framework takes into account the prior knowledge of the signal distribution [187]. The unknown sparse signal $x$ is assigned a prior distribution $p(x)$. The observation $y$ is also a random process with conditional distribution $p(y|x)$. Then, the problem of compressive sensing can be stated as:

$$x = arg\max_{x} p(x|y) = arg\max_{x} p(y|x)p(x) \tag{4.29}$$

Bayesian Methods can be classified into two types: MAP Estimation and hierarchical Bayesian framework. MAP Estimation Framework underlines a distribution for $x$ and recovers the sparse signal based on the measurement vector $y$. The hierarchical Bayesian framework introduces one or more variables that control $x$. The estimation of this hyper-parameter $\gamma$ can be used to compute the posterior distribution for x. The problem can be formulated as follow:

$$y = arg\max_{y} p(\gamma|y) = arg\max_{\gamma} \int p(y|x)p(x|\gamma)p(\gamma)d\gamma \tag{4.30}$$

### Fast Laplace

Bayesian compressive sensing defines a joint distribution of the hierarchical model $p(x, \gamma, \beta, y)$ [188]. This joint distribution satisfies:

$$p(x, \gamma, \beta, y) = p(y|x, \beta)p(x|\gamma).p(\gamma)p(\beta) \tag{4.31}$$

Where $\beta = \sigma^2/2$ denotes the inverse of the noise variance, $\gamma$ and $\beta$ are the hyper-parameters, and the observation $y$ is Gaussian with zero mean and variance equal to $\beta^{-1}$, $p(y|x, \beta)/\ N(y|\phi x, \beta^{-1})$

With a gamma prior placed on $\beta$ as follows :

$$p(\beta|a^\beta, b^\beta) = \Gamma(a^\beta, b^\beta) \tag{4.32}$$

The signal model is equivalent to using a Laplace prior on the coefficients $x$

$$P(x|\gamma) = (\gamma/2)^N exp(\frac{-\gamma}{2}||x||_1) \tag{4.33}$$

The Bayesian inference is given by: $p(x, \gamma, \beta, \lambda/y) = p(x/y, \gamma, \beta, \lambda)p(\gamma, \beta, \lambda/y)$.

Since $p(x|y, \gamma, \beta, \lambda)p(x, y, \gamma, \beta, \lambda)$, then the distribution $p(x|y, \gamma, \beta, \lambda)$ is a multivariate Gaussian distribution $N(x|\mu, \Sigma)$ with parameters :

$$\mu = \Sigma\beta\phi^T y \tag{4.34}$$

$$\Sigma = [\beta\phi^T\phi + diag(\frac{1}{\gamma_i})]^{-1} \tag{4.35}$$

$$\mu = \Sigma\beta\phi^T y \tag{4.36}$$

$p(\gamma, \beta, \lambda|y) = [p(\gamma, \beta, \lambda, y)/p(y)]p(\gamma, \beta, \lambda, y)$ is used to estimate the hyperparameters.

The hyperparameters are estimated by maximizing the joint distribution $p(\gamma, \beta, \lambda, y)$ or equivalently its logarithm:

$$\begin{aligned} l = -1/2log|C| - 1/2y^T C^{(} - 1)y + Nlog(\lambda) - 1/2\Sigma\gamma_i + \nu/2log(\nu/2) - log(\nu/2) + \\ (\nu/2 - 1)log(\lambda) - \nu/2\lambda + (a^{\beta-1})log(\beta) - b^{\beta}\beta \end{aligned} \tag{4.37}$$

The parameters update can be estimated by solving $\frac{dl}{d\lambda} = 0$ and $\frac{dl}{d\beta} = 0$. Performing these operations yields,

$$\lambda = \frac{N - 1 + \nu/2}{\sum_i \gamma_i/2 + \nu/2} \tag{4.38}$$

$$\beta = \frac{N/2 + a^{\beta}}{||y - \phi x||^2 + b^{\beta}} \tag{4.39}$$

The parameter $\nu$ can be estimated by maximizing4.37 with respect to $\nu$, which results in solving the following:

$$log(\nu) + 1 - \psi(\frac{\nu}{2}) + log(\lambda) - \lambda \tag{4.40}$$

To decrease the computational complexity of laplace-based BCS, the authors of proposed to update a single element $\gamma_i$ instead of updating the whole vector $\gamma$; therefore, the function $l$

may be expressed as,

$$l = l(\gamma) = \frac{-1}{2}[log|C_{-i}| - 1/2y^T C_{-i}^{-1} + \lambda/2\Sigma_{i\neq j\gamma_i}] + 1/2[log(\frac{1}{1+\gamma_i s_i}) + \frac{q_i^2\gamma_i}{1+\gamma_i 2i} + \lambda\gamma_i] \quad (4.41)$$

which can be written as

$$l(\gamma) = l(\gamma_{-i} + h(\gamma_i)) \quad (4.42)$$

where $h(\gamma_i) = 1/2[log(\frac{1}{1+\gamma_i s_i}) + \frac{q_i^2\gamma_i}{1+\gamma_i 2i} + \lambda\gamma_i]$ and $q_i$ and $s_i$ are defined as

$s_i = \phi_i^T C_{-i}^{-1}\phi_i$; $q_i = \phi_i C_{-i}^{-1} y$

$\frac{dl}{d\gamma_i} = \frac{dh}{\lambda_i} = 0$ is satisfied at:

$$\gamma_i = \frac{-s_i(s_i+2) + s_i\sqrt{(s_i+\lambda)^2 - 4\lambda(s_i - q_i^2 + \lambda)}}{2s_i^2\lambda} \quad (4.43)$$

This algorithm starts by initializing all $\gamma$ and $\lambda$ to zero then testing if the $q_i^2 - s_i > \lambda$ and $\gamma_i$ if so then $\gamma_i$ is added to the model; otherwise , if $q_i^2 - s_i > \lambda$ and $\gamma_i > 0$ , then the algorithm re-estimates $\gamma_i$; otherwise, if $q_i^2 - s_i < \lambda$, then the algorithm prunes $I$ from the model by setting $\gamma_i$ to zero. Then the algorithm updates $\Sigma$,$q_i$, $s_i$ by using the formula 4.38 and $\nu$ using the formula (4.40)

### Relevance Vector Machine Based BCS

Another probabilistic approach used to estimate the components of $x$ is Relevance Vector Machines (RVM). This algorithm uses a hierarchical prior to estimating a full posterior on

$x$ and on the variance $\sigma^2$, which defines a zero-mean Gaussian prior on each element of $x$. Instead of using the inverse of noise variance, RVM models the prior on $x$ using the precision of a Gaussian density function $a\alpha_i$.

$$p(x|\alpha) = \prod_{i=0}^{N} N(x_i|0, \alpha_i^{-}1) \tag{4.44}$$

In addition, a Gamma prior is considered over *alpha* as,

$$p(\alpha|a, b) = \prod_{i=1}^{N} \gamma(\alpha_i|a, b) \tag{4.45}$$

Similarly, a Gamma prior is considered over $\alpha_0 = \frac{1}{\sigma^2}$:

$$p(\alpha_0|c, d) = \prod_{i=1}^{N} \Gamma(\alpha_0|c, d) \tag{4.46}$$

In order to finds hyper-parameters $\alpha$ and $\alpha_0$ that maximize the marginal likelihood or equivalently its logarithm $L(\alpha, \alpha_0)$ that may be expressed analytically as

$$L(\alpha, \alpha_0) = log(p(y|\alpha, \alpha_0)) = log(\int p(y|x, \alpha_0)p(x|\alpha, \alpha_0)dx) = \frac{-1}{2}[Klog(2\pi)+log(|C|)+y^T C^{-1}Y] \tag{4.47}$$

where $C = \sigma^2 I + \phi\Delta^{-1}\phi^T$ and $\Delta = s = diag(\frac{1}{\gamma_i})$

Thus, recovering a sparse signal from a few measurements in the context of RVM becomes the search for the hyper-parameters $\alpha$ and $\alpha_0$. The hyperparameters are estimated using the EM algorithm

$$\alpha_i^{new} = \frac{\gamma_i}{\mu_i}, \quad i \in \{1, 2, 3, ..., N\} \tag{4.48}$$

with $\mu_i = 1 - \alpha_i \Sigma_{ii}$ and $\Sigma_{ii}$ denotes the $i^{th}$ diagonal element of $\Sigma$

and

$$\alpha_0 = \frac{k - \Sigma_i \gamma_i}{||y - \phi\mu||_2^2} \tag{4.49}$$

## 4.6 Illustrative Results and Discussions

This work considers a sparse signal of length $N = 1024$ that contains $K$ spikes randomly chosen. This sparse signal is then sensed using a Toeplitz measurement matrix whose size $M \times N$, where $M$ is the number of measurements and $N$ is the length of the sparse signal. The reason behind using the Toeplitz matrix is that it reduces the randomness and memory usage, and it also allows fast acquisition and recovery, as we will show in the next section. The sparse signal is corrupted with an additive Gaussian white noise. The generation of the Gaussian noise is performed via a random number generator with a zero mean and a standard deviation $\sigma_m = 0.005$. For each sparsity level K or a number of measurements M, we consider Monte Carlo simulations with a number of trials equal to 100 trials.

To compare the performance of sparse recovery algorithms, we used four metrics: recovery error, recovery time, covariance, and phase transition diagram. Recovery error is a metric to evaluate the error between the original sparse signal and the recovered one. In order to calculate the recovery error, we used the mean squared error between the original signal and the reconstructed signal. We also used the recovery time, which is a metric that measures the time needed by each algorithm to find the solution to the sparse recovery problem. In

addition, we used covariance as a metric that reflects the correlation between the original signal and the sparse signal.

### 4.6.1 Examples of Results

Fig. 4.4 shows the recovery error of the six algorithms for the number of measurements with a fixed sparsity level of the signal. When the number of measurements is very low, Basis Pursuit shows better performance than all other techniques. However, when the number of measurements is higher than 130, Bayesian techniques and orthogonal matching Pursuit show better performance in recovery error, which decreases to 0% when the number of measurements is larger than 170.

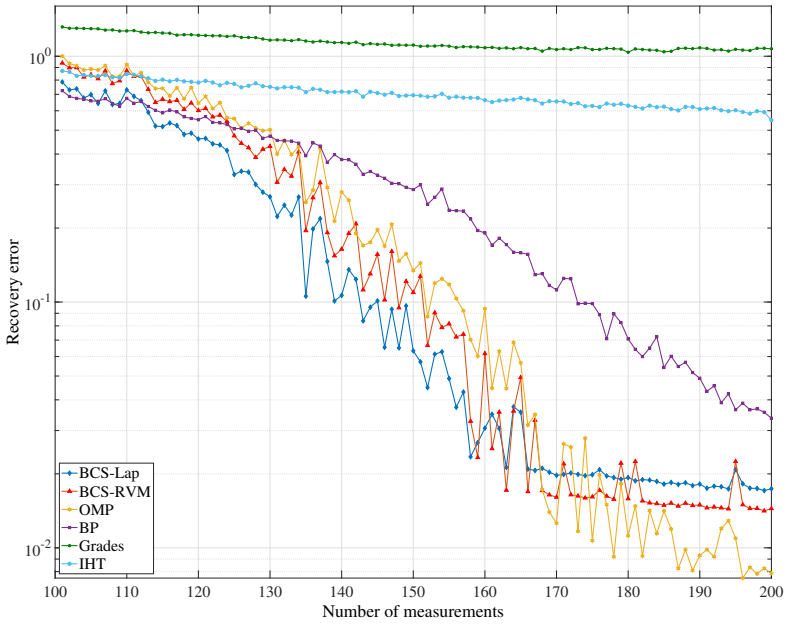


Figure 4.4: Recovery error vs. number of measurements

Fig. 4.5 shows the recovery error of the previous algorithms as a function of the sparsity level for a fixed number of measurements $M = 200$. According to this Figure, the recovery errors

of all algorithms increase as the sparsity of the signal increases. This figure also compares the recovery error of these six sparse recovery algorithms. This figure indicates that Bayesian techniques, Orthogonal Matching Pursuit, and Basis Pursuit at low sparsity level ($k < 50$) consistently recover the original signal with 0% recovery error. As the sparsity increases, the recovery error of these algorithms increases, but they still show better performance compared to Gradient Descent and Iterative Hard Thresholding techniques.

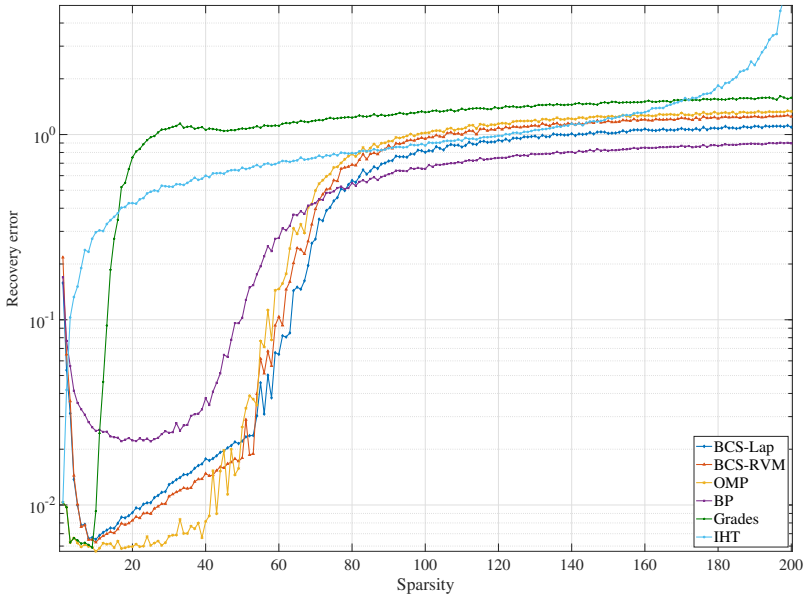


Figure 4.5: Recovery error vs. sparsity

Fig. 4.6 shows that the behavior of Bayesian techniques is opposite to those of Iterative Hard Thresholding and Basis Pursuit techniques. As the number of measurements increases, the recovery time of Bayesian techniques decreases. However, the recovery time for Basis Pursuit and Iterative Hard Thresholding increases as the number of measurements increases.

Fig.4.7 shows that Orthogonal Matching Pursuit and Gradient Descent techniques are faster than all other techniques for all the number of measurements and all sparsity levels. As shown

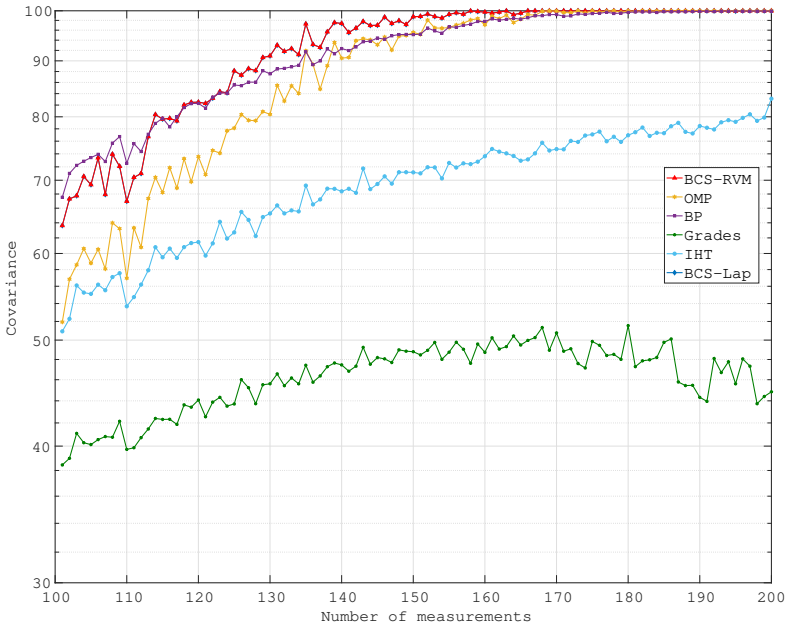


Figure 4.6: Covariance vs. number of measurements

in this figure, for a very low sparsity level (k<20), Bayesian techniques are faster than all techniques except Orthogonal Matching Pursuit. However, when sparsity is higher than 50, the behavior of the recovery time of the Bayesian technique via Laplace Prior increases significantly to become higher than those of the other techniques. However, Bayesian via Relevance Vector Machine recovers the signal with an average time below 0.1 sec.

Fig. 4.8 shows that as the number of measurements increases, the covariance increases reaching 100% when the number of measurements is very high for Bayesian, Basis Pursuit, and Orthogonal Matching Pursuit techniques. On the other hand, Iterative hard Thresholding and Gradient Pursuit show a lower correlation.

Fig. 4.9 shows that the covariance is around 100% when sparsity is between 0 and 40 for Bayesian techniques, Basis Pursuit, and Orthogonal Matching Pursuit techniques. The covariance decreases as the sparsity level of the signal increases. This figure also shows

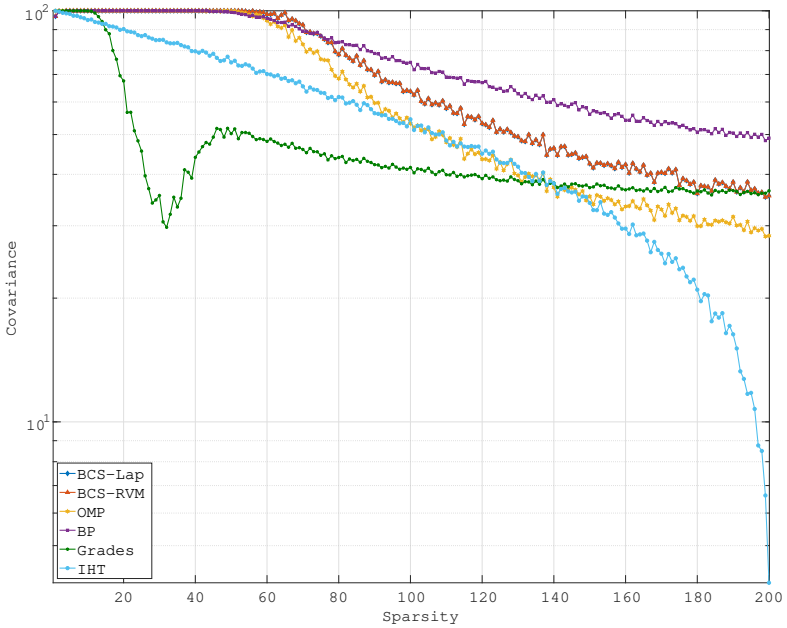


Figure 4.7: Covariance vs. sparsity

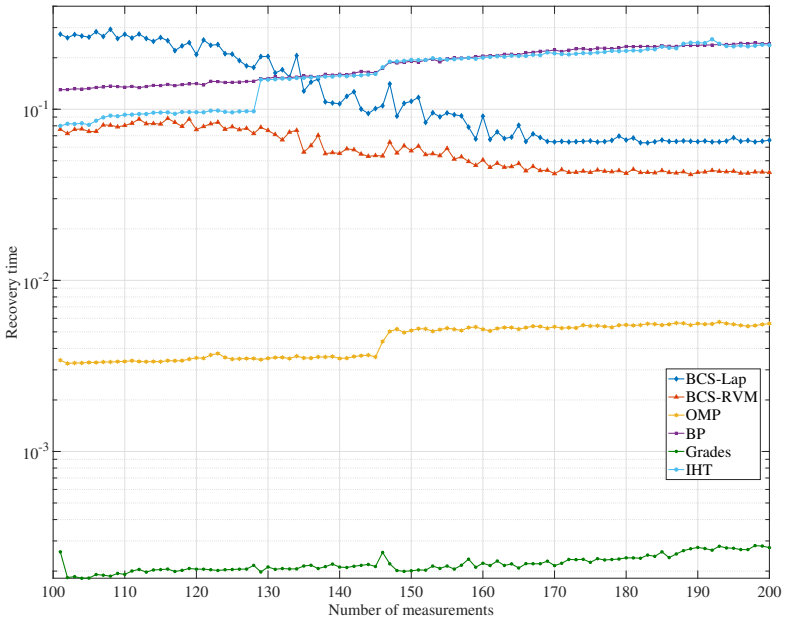


Figure 4.8: Recovery time vs. number of measurements

that the Basis Pursuit technique has better performance followed by Bayesian and greedy techniques.

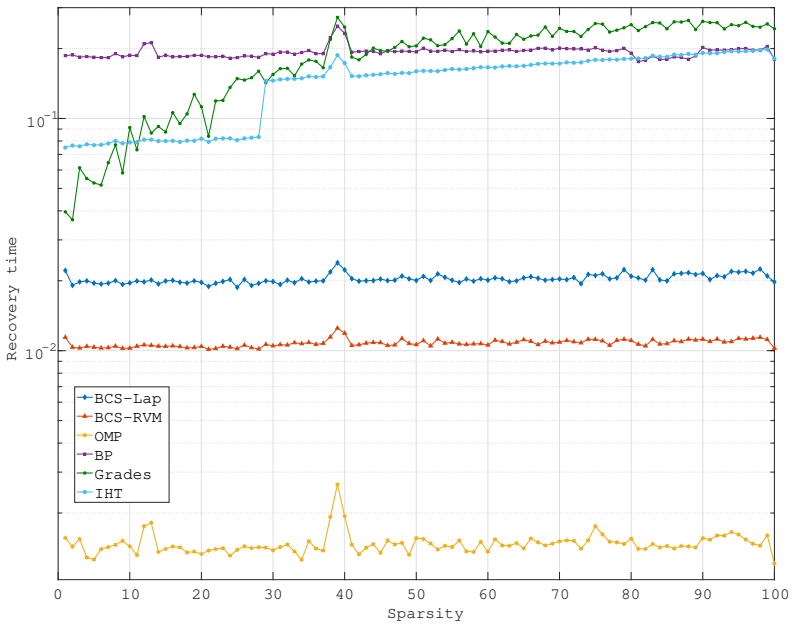


Figure 4.9: Recovery time vs. sparsity

To select the best measurement matrix, we have used Bayesian compressive sensing with eight different measurement matrices. Examples of results are shown in Figs. **??** through (f). Fig.4.10 shows the reconstruction error of the Bayesian recovery algorithm via Fast Laplace with eight different measurement matrices as a function of the sparsity level. This figure indicates that for a sparsity level small than 60, all the measurement matrices have approximately the same recovery error. They enable CS with a small reconstruction error except for the Fourier measurement matrix with a higher recovery error than the other measurement matrices. For a sparsity greater than 60, the recovery error of the matrices, Binary BCH, Chirp, Circulant, Bernoulli, Gaussian, and Toeplitz, increases exponentially, and it becomes higher than that of the Fourier measurement matrix when the sparsity order reaches the value K=120. The binary BCH matrix and Hadamard have better performance

than all the measurement matrices for a sparsity level of less than 100. For a sparsity between 100 and 120, the Toeplitz matrix performs better in terms of recovery error, and for a sparsity higher than 120, the Fourier matrix has the smallest error compared to all the other matrices.

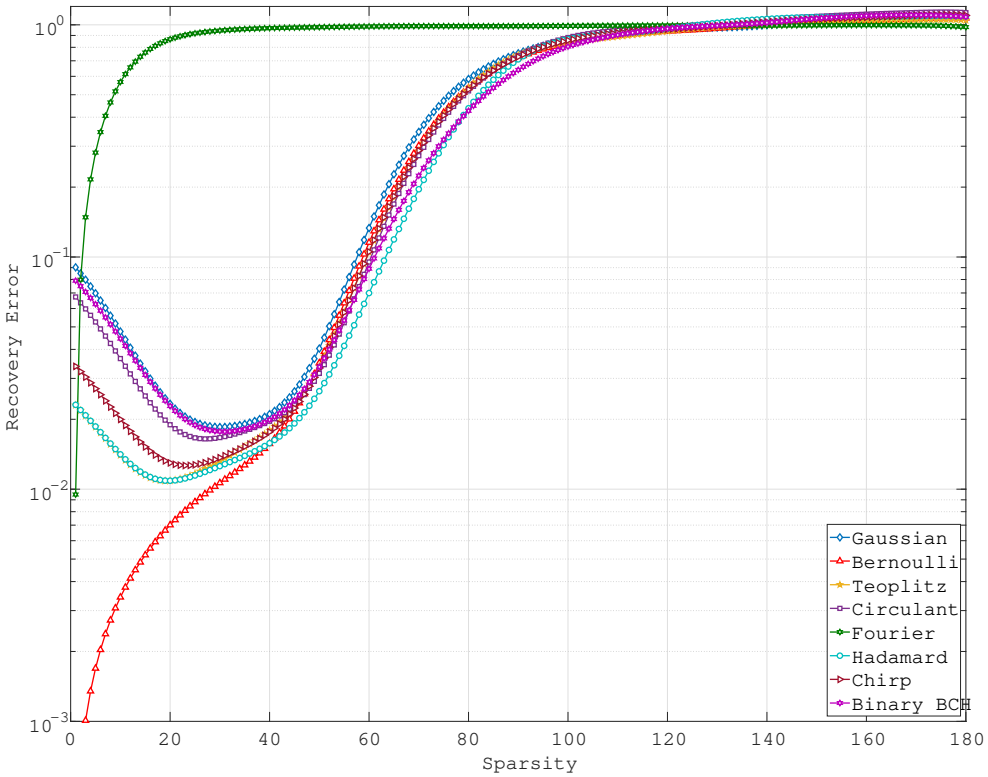


Figure 4.10: Recovery error vs. sparsity

4.11 shows the recovery error for the number of measurements. For a small number of measurements less or equal 100, The Fourier matrix has the smallest recovery error among all the other measurement matrices. For a number of measurements higher than 85, the recovery errors of the Gaussian, Bernoulli, Toeplitz, circulant, Hadamard chirp, and binary BCH, decrease to become smaller than the one of the Fourier matrix. All these measurement matrices, except the Fourier matrix, have the same performance, but the binary BCH and

Hadamard matrices show better performance than all the other matrices.

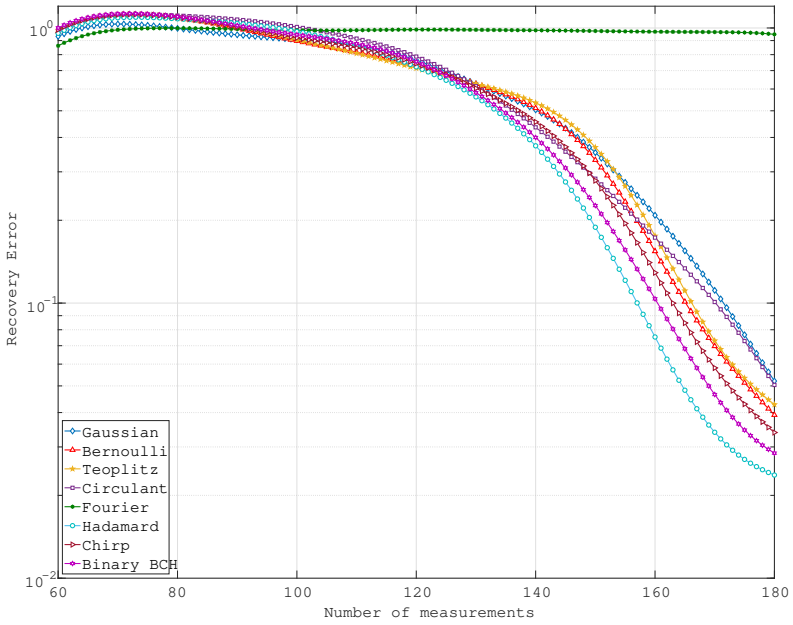


Figure 4.11: Recovery error vs. number of measurements

Figs. 4.12 and 4.13 show the recovery time of the Bayesian recovery technique for different measurement matrices as a function of the sparsity level and the number of measurements, respectively. From these two figures, one can see that, for all the measurement values and for all the sparsity values, the circulant matrix is the fastest to recover the signal and the Partial Fourier matrix is the slowest. Toeplitz, Gaussian, and Bernoulli have slightly the same recovery times.

Fig. 4.14 illustrates the covariance of the Bayesian recovery technique for eight measurement matrices as a function of the sparsity level. This figure indicates that most of the measurement matrices have the same covariance shape except the Fourier matrix. For sparsity orders less than 60, the covariance is almost 100%, but it decreases exponentially for sparsity orders higher than 60. Among these measurement matrices, Binary BCH and Partial Hadamard

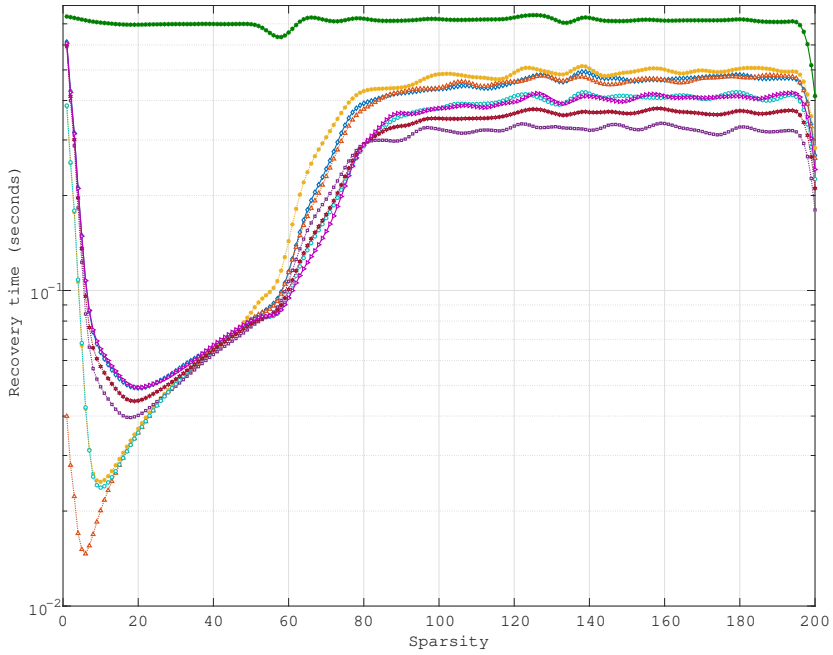


Figure 4.12: Recovery time vs. sparsity

matrices have the highest covariance values. On the other hand, Random Partial Fourier has the lowest covariance value for all the sparsity values, which means that this matrix performs less than all the other measurement matrices in terms of covariance.

Fig. 4.15 shows the covariance of the eight measurement matrices for the number of measurements. As can be observed, the covariance increases as the number of measurements increase to reach 100% except for the Partial Fourier matrix. It can also be observed that the covariance of all measurement matrices has slightly the same shape, except for the Random Partial Fourier matrix, which has the lowest value of the covariance for all values of measurements.

Fig. 4.16 gives the phase transition diagram curves of the eight measurements matrices. These curves are obtained for 90% of the success rate of the recovery process. As can be seen, the curve of the Binary BCH matrix is above all the other curves for an undersamling

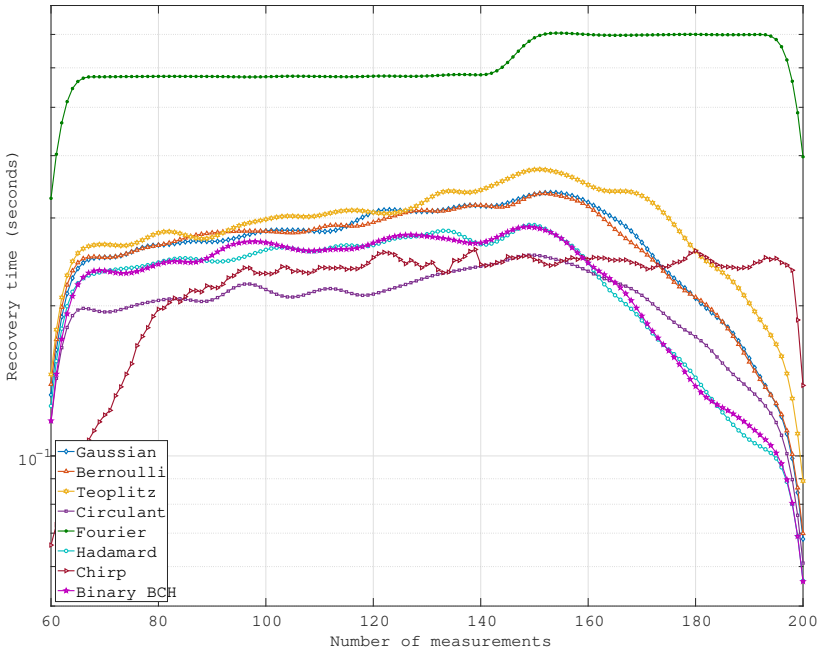


Figure 4.13: Recovery time vs. number of measurements

rate smaller than 0.4, which means that this measurement matrix has a higher success rate than the other measurement matrices. For higher undersampling rates, the curve of the Binary BCH matrix is below all the other curves, which means that this matrix has a lower success rate than the other measurement matrices. The Partial Fourier and Partial Hadamard curves are above the curves of random matrices for an undersampling rate smaller than 0.5. These two curves are below the random matrices curves, Gaussian and Bernoulli curves, for a higher rate. Toeplitz's curve is above all the other curves for an undersampling rate higher than 0.7. The curve of the circulant matrix is below all the other curves, which means that it performs less than all the other measurement matrices.

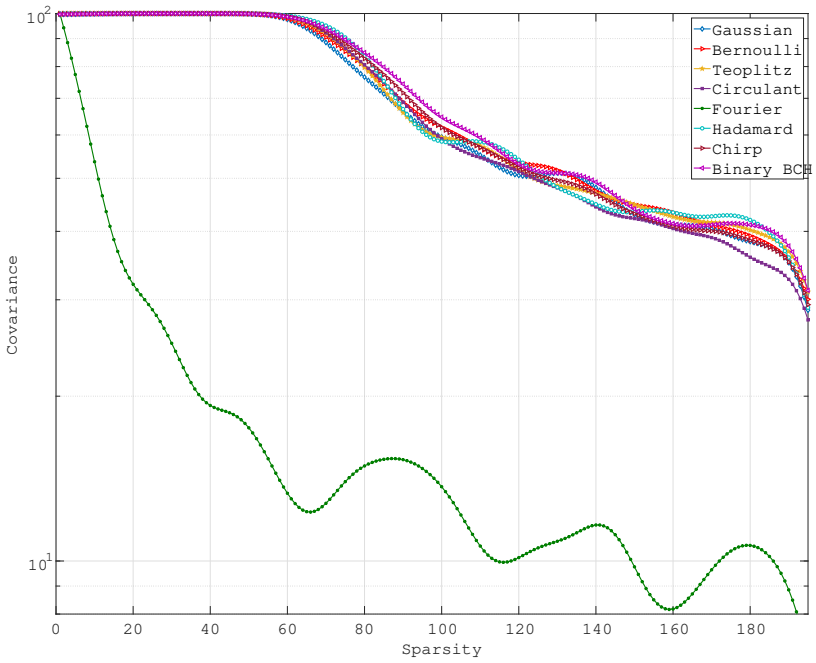


Figure 4.14: Covariance vs. sparsity

## 4.7 Summary

This chapter showed how the theory of compressive sensing techniques could be leveraged in estimating the mmWave channel. The state-of-the-art techniques focus on random channel measurement vectors and greedy methods. However, this remains an issue because it is not optimal to distribute the sensing power in all spatial directions. Therefore, we proposed to investigate the optimal compressive sensing techniques for mmWave communication. We compared the performance of several recovery algorithms from three different categories: convex and relaxation, greedy, and Bayesian, and several measurement matrices belonging to mainly two classes: random and deterministic classes. We showed that a deterministic measurement matrix could achieve comparable performance to a random measurement matrix. We demonstrated that it is interesting to design deterministic sensing vectors instead

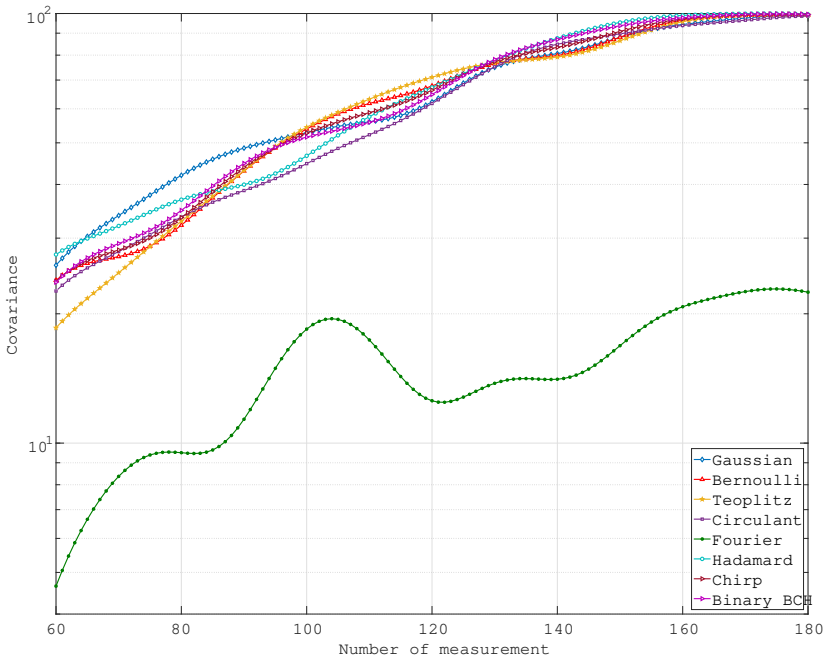


Figure 4.15: Covariance vs. number of measurements

of using random channel measurements. Using deterministic channel measurement vector construction can help focus the sensing power in a specific direction and achieve the same performance as random ones. We also showed that greedy techniques are fast but have higher recovery errors than convex and relaxation techniques. We also showed that the Bayesian compressive recovery techniques achieve better performance while being fast.

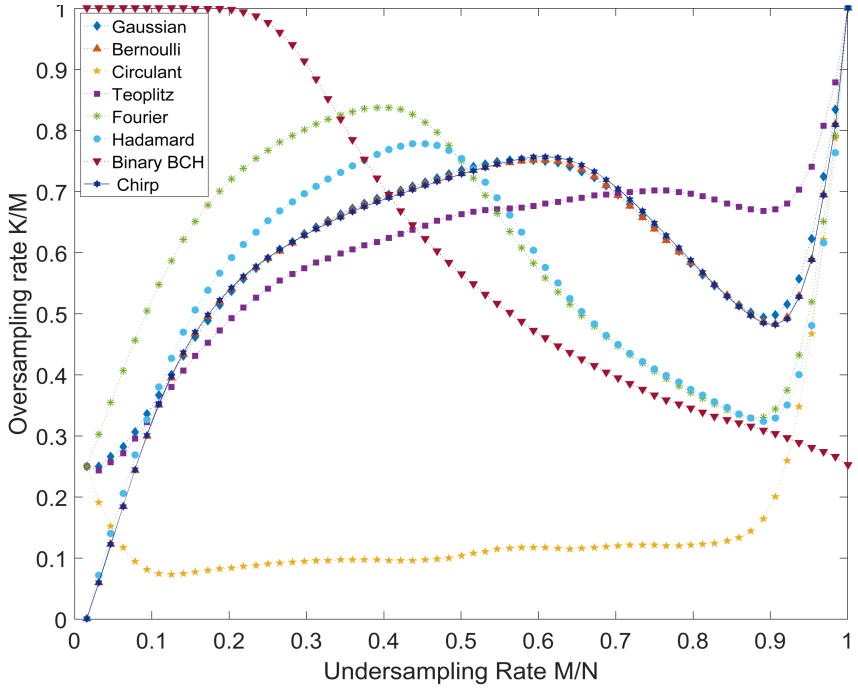


Figure 4.16: Phase Transition Diagram of measurement matrices (curve at 90%).

# Chapter 5

# Deep Learning Based Frequency Selective Channel Estimation in Hybrid Beamforming

The mmWave channel is considered to be frequency selective. Estimation of the channel in hybrid beamforming is challenging because the mmWave channel is high dimensional. The analog combiner at the receiver compresses the antennas' output and, therefore, the receiver from a limited number of pilot samples has to reconstruct the channel. This challenging and several traditional techniques based on compressive sensing theory has been proposed. This chapter aims to investigate deep learning theory in estimating the frequency selective channel considering a hybrid architecture. To train machine learning, we consider ray tracing to simulate the wireless environment as training deep learning requires collecting a big dataset, and using real-world measurement is expensive and impractical. Since ray-tracing provides a good wireless environment simulator, it remains a reliable technology to build a realistic dataset to train deep learning techniques.

In this model, we consider a base station (BS) with $N_{BS}$ communicating with a mobile station (MS) with $N_{MS}$. The number of RF chains are $N_{rf} << N_{BS}$. The base station transmits a signal after applying the baseband and analog precoding matrix on the vector

of transmitted symbols $s$.

$$\mathbf{x} = \mathbf{F_T}[k]\mathbf{s}[k] + \mathbf{n}[k] \tag{5.1}$$

where $F_T[k]$ is equal to $F_{RF}F_{BB}[k]$ and $s[k]$ is the data stream, and $n[k]$ is an additive white Gaussian noise.

At the mobile station side, the received signal can be expressed as

$$\mathbf{r}[k] = \mathbf{H}[k]\mathbf{F_T}[k]\mathbf{s}[k] + \mathbf{n}[k] \tag{5.2}$$

where $\mathbf{H}[k]$ is the mmWave channel matrix, $\mathbf{n}[k]$ is the Gaussian noise.

On the MS side always and after the digital and analog combiners are applied on the observed signal, the received signal can be expressed as

$$\mathbf{r}[k] = \mathbf{W_T^H}[\mathbf{k}]\mathbf{H}[\mathbf{k}]\mathbf{F_T}[\mathbf{k}]\mathbf{s}[\mathbf{k}] + \mathbf{W_T^H}[\mathbf{k}]\mathbf{n}[\mathbf{k}] \tag{5.3}$$

Given that the channel is of limited scattering, we consider a channel of L scatterers, each of which is contributing a single propagation path between the transmitter and the receiver. Under this model, the channel can be expressed as

$$\mathbf{H}[k] = \sqrt{\frac{N_t N_r}{L\rho_L}} \sum_{l=1}^{L} \alpha_l \rho_{rc}(dT_s - \tau_l)\mathbf{a}_{MS,k}(\theta_l)\mathbf{a}_{BS,k}^H(\phi_l) \tag{5.4}$$

where $a_{MS,k}(\theta)$ and $a_{BS,k}(\phi_l)$ are the antenna array response vectors at the BS and MS, respectively, $\alpha_l$ is the complex gain of the $l^{th}$ path, and $\rho$ denotes the average pathloss, $\rho_{rc}(\tau)$ denotes a function that models pulse shape and analog filtering evaluated at $\tau$. $\tau_l$ denotes the time delay,

$$\mathbf{a}_{BS,k}(\phi_l) = \frac{1}{\sqrt{N_{BS}}}[1, e^{j2\pi/\lambda_k dsin(\phi_l)}, ..., e^{j(N_{BS}-1)2\pi/\lambda_k dsin(\phi_l)}]^T \tag{5.5}$$

where $\lambda$ is the wavelength, and d is the distance between antenna elements. In the same fashion,

$$\mathbf{a}_{MS,k}(\theta) = \frac{1}{\sqrt{N_{MS}}}[1, e^{j2\pi/\lambda_k dsin(\theta_l)}, ..., e^{j(N_{MS}-1)2\pi/\lambda_k dsin(\theta_l)}]^T \tag{5.6}$$

In more compact format, the channel can be written as

$$\mathbf{H}[k] = \mathbf{A}_{MS}[k] diag(\alpha)[k] \mathbf{A}_{BS}^H[k] \tag{5.7}$$

where $\boldsymbol{\alpha} = \frac{\sqrt{N_{BS}N_{MS}}}{\rho}[\alpha_1, ...\alpha_L]^T$

$$\mathbf{A}_{MS}[k] = [\mathbf{a}_{MS,k}(\theta_1), \mathbf{a}_{MS,k}(\theta_2), ..., \mathbf{a}_{MS,k}(\theta_L)] \tag{5.8}$$

$$\mathbf{A}_{BS}[k] = [\mathbf{a}_{BS,k}(\phi_1), \mathbf{a}_{BS,k}(\phi_2), ..., \mathbf{a}_{BS,k}(\phi_L)] \tag{5.9}$$

When the receiver has no prior knowledge of the channel, it is a question of the receiver to estimate this channel, the authors of [43] have shown that the problem of estimating the channel in mmWave can be formulated as a sparse recovery problem, where the measurement matrix is formed by the precoding and combining matrix and a sparsifying matrix. In the next step, we describe this process step by step. During the training phase,

$$\mathbf{y_{p,q}}[k] = \mathbf{w_q^H H}[k]\mathbf{f_p s_p}[k] + \mathbf{W_q^H n_{q,p}}[k] \tag{5.10}$$

if each MS perform the measurements,

$$\mathbf{y_p}[k] = \mathbf{W^H H}[\mathbf{k}]\mathbf{f_p s_p}[\mathbf{k}] + diag(\mathbf{W^H}[\mathbf{n_{1,p}}, ..., \mathbf{n_{M_{MS},p}}]) \tag{5.11}$$

where $\mathbf{W} = [\mathbf{w_1}, \mathbf{w_2}, \mathbf{w_q}, ..., \mathbf{w_{M_{MS}}}]$ is the measurement matrix, which to be formed the MS has to perform successive measurements, $\mathbf{w}_q$ is the combining vector at the $q^{th}$ instant.

$$\mathbf{Y}[k] = \mathbf{W^H H[k] F S[k]} + \mathbf{Q}[k] \tag{5.12}$$

$\mathbf{F} = [\mathbf{f_1}, \mathbf{f_2}, \mathbf{f_3}, ..., \mathbf{f_{M_{BS}}}]$ is the beamforming matrix used at the base station, and $\mathbf{Q}$ is the noise matrix.

As all the data stream are equal and with the same power.

$$\mathbf{Y}[k] = \sqrt{P}\mathbf{W^H H}[k]\mathbf{F} + \mathbf{Q}[k] \tag{5.13}$$

To exploit the sparsity nature of the mmWave, we vectorize

$$\mathbf{y_v}[k] = vec(\mathbf{Y}[k]) = \sqrt{P}vec(\mathbf{W^H H}[k]\mathbf{F}) + vec(\mathbf{Q}[k]) \tag{5.14}$$

using the properties of the Kronecker's product, we can write

$$\mathbf{y_v}[k] = \sqrt{P}vec(\mathbf{W^H H}[k]\mathbf{F} + vec(\mathbf{Q}[k]) \tag{5.15}$$

$$\mathbf{y_v}[k] = \sqrt{P}(\mathbf{W^H} \otimes \mathbf{F})vec(\mathbf{H}[k]) + vec(\mathbf{Q}[k]) \tag{5.16}$$

$$\begin{aligned} \mathbf{y_v}[k] = \sqrt{P}(\mathbf{F^T} \otimes \mathbf{W^H})(\mathbf{A}^*_{BS}[k] \odot \mathbf{A}_{MS}[k]) \\ \alpha + vec(\mathbf{Q}[k]) \end{aligned} \tag{5.17}$$

where $\odot$ is Khatri-Rao product. The matrix $\mathbf{A}^*_{BS}[k]()\mathbf{A}_{MS}[k]()$ consists of the columns

made of $\mathbf{a}^*_{BS}[k](\phi_l) \otimes \mathbf{a}_{MS}[k](\theta_l)$, $l = 1, 2, ..., L$ where each column $l$ represents the Kronecker product of the BS and MS array response vectors associated with the AoA/AoD of the $l^{th}$ path of the channel. Further, we assume that the AoAs, and AoDs are taken from a uniform grid of $N$ points with $N >> L$.

By neglecting the quantization error, we can write

$$\mathbf{y_v}[k] = \sqrt{P}(\mathbf{F^T} \otimes \mathbf{W^H})\mathbf{A_D}[k]\mathbf{z}[k] + \mathbf{n_Q}[k] \tag{5.18}$$

where $\mathbf{A_D}[k]$ is an $N_{BS}N_{MS} \times N^2$ dictionary matrix that consists of $N^2$ columns $\mathbf{a}^*_{BS}(\phi_u) \otimes \mathbf{a}_{MS}(\theta_v)$ and $\phi_u = 2\pi u/N$ and $\theta_v = 2\pi v/N$. $\mathbf{z}[k]$ is an $N^2 \times 1$ vector which carries the path gains of the corresponding quantized directions.

$\mathbf{y_v}[k]$ shown in (5.18) represents a sparse formulation of the channel estimation problem. We define $= (\mathbf{F^T} \otimes \mathbf{W^H})\mathbf{A_D}[k]$ as the sensing matrix, which should guarantee the recovery of the non-zero elements of the vector $z$ with high probability, and with a small number of measurements.

The estimated vectors $\mathbf{z}[k], k = 0, 1, ..., K-1$ can be found by solving the sparse recovery problem:

$$\begin{aligned} \mathbf{z}[k]_{k=0,1,...,K} = & arg \min_{z[l], l=0,...,K-1} \sum_{l=0}^{K-1} \|\mathbf{z}[l]\|_1 \\ & subject \ \ to \ \ \sum g^*[l] C_w g[l] \leq \epsilon \end{aligned} \tag{5.19}$$

where $g[l] = y[l] - \phi\mathbf{A_D}[k]z[l]$, $\epsilon$ denotes the maximum error between the measurement and the recovered signal. Compressive sensing uses the RIP property to guarantee recovery. At the end of the channel estimation, the base station should have acquired the knowledge of

the steering matrix $A_{BS}$, the MS steering matrix $A_{MS}$ and the estimated gain path-vector $\alpha$

In the case of hybrid beamforming, the $\mathbf{F^T}$ is the product of the digital and analog precoder, while the $\mathbf{W^T}$ is the product of the analog and digital combiners. Reinforcement learning techniques can be used to determine the analog precoder and combiner.

Compressive sensing recovers the sparse vector from a measurement vector acquired by a random measurement matrix, so one can use random precoder and combiner to construct the measurement matrix.

Given a Raymobtime training set $[(RXtrainingPilot^{(1)}, h^{(1)}), (RXtrainingPilot^{(2)}, h^{(2)}), \ldots, (RXtrainingPilot^{(N)}, h^{(N)})]$, the neural network optimize the weights $W_{ij}^{l}$ in order to minize the mean squared error. This can be expressed as,

$$NMSE(W) = \frac{1}{N}\sum_{i=1}^{N}(h_i - \hat{h}_i)^2 \tag{5.20}$$

where $N$ is the training data size, $h_i$ denotes the actual channel of the $i^{th}$ training sample, $\hat{h}_i$ denotes the predicted channel for the $i^{th}$ training sample, and $W$ denotes the weight matrix. Therefore, the goal of the training is to minimize $NMSE(W)$. This optimization can be formulated as,

$$\min_{W} NMSE(W) \tag{5.21}$$

Several optimizers can be used to determine the optimal weights that minimize the mean squared error. Examples of these techniques are gradient descent, stochastic gradient descent, and Adam optimizer.

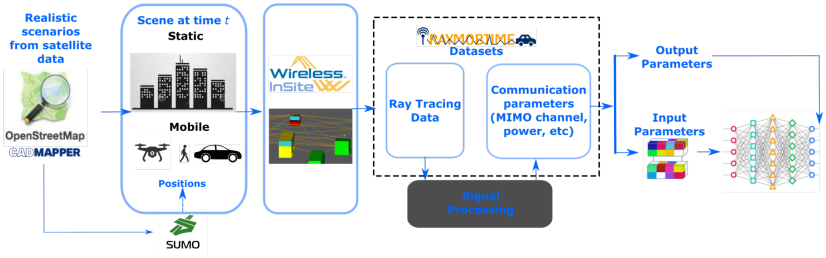


Figure 5.1: Flow for channel estimation [1]dataset [7]

## 5.1 Dataset Description

To train CNN-based channel estimation, we have considered a dataset generated from Raymobtime. The process of Raymobtime dataset generation is shown in Fig. 5.1. This dataset has been generated using ray tracing to build a realistic dataset, which can be used to train machine learning techniques. The use of this methodology overcomes the high cost of collecting real-world measurements. ITU has used this dataset for many challenges for AI5G challenges for wireless communications. A description of the process of Raymobtime dataset generation is given in [7, 189]

## 5.2 DNN-Based Channel Estimation

### Feature Learning

In feature learning, there are three main tasks on which the convolution layers are built: convolution, Rectified linear unit, and Pooling. Convolution: puts the input signals through a set of convolutional filters, each of which activates certain features from the signals. Rectified linear unit (ReLU) allows faster and more effective training by mapping negative values to

[1]https://research.ece.ncsu.edu/ai5gchallenge/

zero and maintaining positive values. Pooling: simplifies the output by performing nonlinear downsampling, reducing the number of parameters that the network needs to learn.

The neural network consists of at least three parts: one input layer, several hidden layers, one output layer. The input layer is composed of the input features selected in the feature layers previously described in Å. The hidden layers learn the inputs' abstract representations. The output layer implements the network classification. Each neuron is an activation function g. The neuron's output is the activation function of the sum of all the weighted input values $\sum_{i=0}^{i=n} w_i x_i$and the bias $Bias$, which can be expressed as:

$$g = g(\sum_{i=0}^{i=n} w_i x x_i + Bias) \tag{5.22}$$

where g denotes the activation function. Forward propagation is one but an essential concept used in neural networks. Next, we describe the mathematical model of forward propagation. The input features are given as:

$$x = [x_0, x_1, x_2, ..., x_m] = a^0 \tag{5.23}$$

where m denotes the number of the inputs features.

In general, the $i^{th}$ neuron of the $l^th$ layer is given by:

$$a_j^{(l)} = g_j^{(l)}(\sum_{i=0}^{i=n} a_i^{(l-1)} + b_{ji}^{(l)}) \tag{5.24}$$

The output layer gives the hypothesis function as follows:

$$h_\theta(x) = g(\sum_{i=0}^{i=n} a_i^{(l)}) \tag{5.25}$$

Next, we describe how neural networks acquire their weights and biases using the gradient descent algorithm; this process is known as backpropagation.

Given a training set $[(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), ..., (x^{(m)}, y^{(m)})]$, the cross-entropy cost function $J(\theta)$, or the cross-entropy error function describes how the cost changes as a function of the weights and the bias. It is equivalent to minus log-likelihood for the data $y^{(i)}$ under the hypothesis $h_\theta(x)$. The mathematical expression of this cost function is given by:

$$J(\theta) = \frac{-1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} y_k^{(i)} log(h_\theta(x^i)_k) + (1 - y_k^{(i)}) log(1 - (h_\theta(x^{(i)})_k)) \tag{5.26}$$

where $m$ is the training data size, $K$ is the number of the output classes, $h_\theta$ is the hypothesis function, and $\theta_(j, i)^(l))$ is the weights assigned to the connection between the $i^{th}$ and $j^{th}$ neurons of the $l^{th}$ layer. Once the cross-entropy function is determined, the next step is to minimize $J(\theta)$ in backpropagation, expressed as:

$$\min_\theta J(\theta) \tag{5.27}$$

The process computes the partial derivatives of the cross-entropy function J with respect to any weights w in the neural network. Several optimizers can be used to determine the optimal weights that minimize the cost function. Examples of these techniques are gradient descent, stochastic gradient descent, and Adam optimizer.

## 5.3 TD3 Based Channel Estimation

In this section, we describe TD-3 based channel estimation. It can be seen that the channel depends on the steering matrix $A_{BS}$, the MS steering matrix $A_{MS}$, and the estimated gain path-vector $\alpha$. Given that the channel contains only a few dominant paths, we consider

the action to be the concatenation of vectors AoAs estimated, AoDs estimated, and the corresponding real and imaginary part of the gain.

$$\begin{aligned} action^{(t)} =& [vec(AoAs^{(t)}), vec(AoDs^{(t)}), \\ & vec(real(\alpha^{(t)})), vect(Img(\alpha^{(t)}))] \end{aligned} \tag{5.28}$$

where $vec(AoAs)$ is the vectorized matrix of the $A_{BS}$, where $vec(AoDs)$ is the vectorized matrix of the $A_{BS}$ of the dominant paths, $vec(real(\alpha))$ and $vect(Img(\alpha))$ corresponds to the vector of the real part of the gain and the vector of the imaginary part of the gain, respectively, $^{(t)}$ refers to the current learning iteration. The state considered is the current signal-to-noise ratio value, and the next state is a higher signal-to-noise ratio value plus one. When the signal to noise ratio reaches the value of $0dB$, the algorithm reaches its terminal state.

TD-3 agent takes action (5.28) and receives a positive reward if the action help in constructing an estimate of the channel close to the actual channel matrix and receives a negative signal reward otherwise.

## 5.4 Results

To evaluate the performance of the proposed approach, we have considered a dataset that consists of 10000 training samples, and we split it to 70% and 30% for training and testing, respectively. We have considered SNR to be as low as -15, -10, and -5 dB. The evaluation metric consists of the mean square error. The learning rate is set to 0.0001, and the number of iterations is 60.

To generate these channels, we have considered a massive MIMO system with the number

Table 5.1: System Parameters for Data Generation

| **Parameter** | **Value** |
|---|---|
| $N_t$ | 16 |
| $N_r$ | 64 |
| Qbit | 4 |
| $\mathrm{N}_{rf}^{t}$ | 2 |
| $\mathrm{N}_{rf}^{t}$ | 4 |
| Number of fft | 256 |
| $P_t$ | $1mW$ |
| Number of samples | 10000 |
| Number of Training symbols | 100 |
| Training/testing percentage | 70%/30% |

of TX antennas 16, and the number of RX antennas is 64 antennas elements. The number of TX RFCs is two, and the number of RX RFCs is four. The number of data streams is two. Frequency sampling is 1760. The transmit power is 1mW. The number of filters is 20, and the number of training symbols to be received by each channel is 100. The number of the quantization bits is 4. The number of subcarriers in the MIMO-OFDM system is 256. These parameters are summarized in Table 5.1.

**Structure of the inputs and the outputs:** The inputs consist of two matrices of size (256, 400), one for the real and one for the imaginary part. The output consists of a matrix (100, 8). To work with these inputs, we have to perform some preprocessing to merge these two matrices to form the input which has to be (150,150,3) for Inception and (224,224,3) for Vgg16, and (160,160,3) for DensNet.

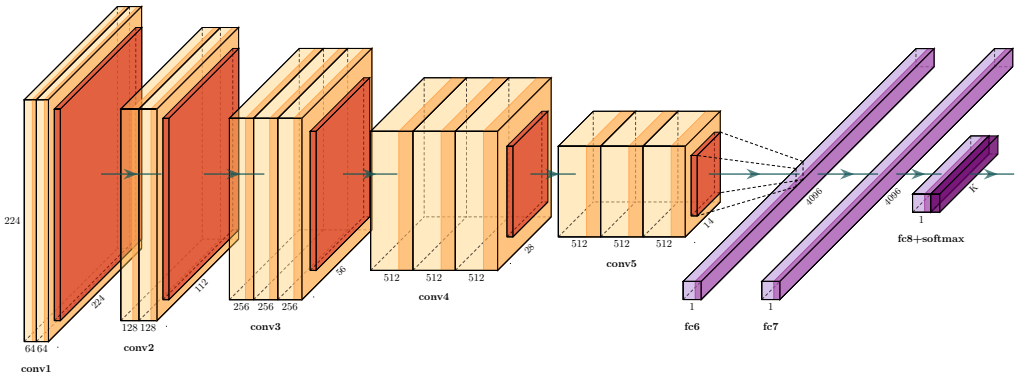


Figure 5.2: Architecture VGG16 [2]

Table 5.2: Performance comparison between different CNN architectures in estimating frequency selective channel (raymobtime 70%- Testing 30 %) (frozen parameters)

| **Architecture** | **SNR** | **MSE** | **Training time** |
|---|---|---|---|
| VGG16 | -10 dB | 0.100 | (4560s) 1.26 hours |
| DenseNet169 | -10 dB | 0.1073 | (5940s) 1.65 hours |

## 5.5 Illustrative Results and Discussions

### 5.5.1 DRL-Based Channel Estimation: Example of Results

We also investigated the theory of deep reinforcement learning for compressive channel estimation. Examples of results are shown in Fig. 5.3. Twin-delayed deep deterministic policy gradient has a relative spectral efficiency to the unconstrained precoding with perfect CSI. This figure also shows that the proposed algorithm is superior to the hybrid precoding based on adaptive channel estimation.

[2]https://research.ece.ncsu.edu/ai5gchallenge/

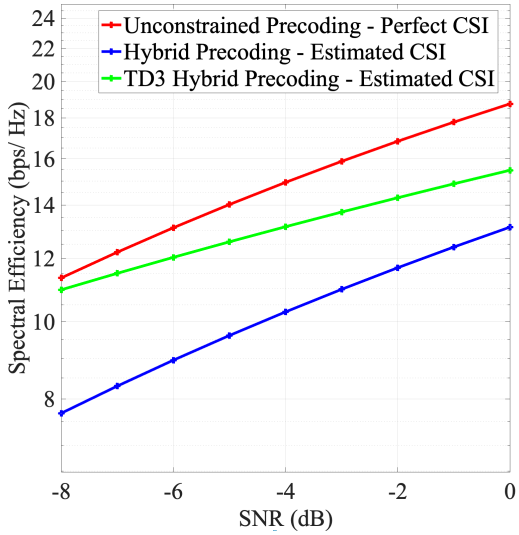


Figure 5.3: Spectral efficiency of TD-3, Adaptive Channel Estimation, and unconstrained precoding based on perfect CSI

### 5.5.2 Light GBM based Channel Parameters Estimation: Example of Results

Examples of the results are shown in Fig. 5.4 through Fig. 5.12.

Fig. 5.4 shows the learning curve l2 error as a function of the number of iterations for both training and validation of light GBM with GOSS as a boosting technique and with a maximum depth of 20 and the number of leaves is 127. It can be seen from this figure that both the curves are decreasing functions of the number of iterations. At iteration 100, the functions experience a spike before decreasing again to reach their low value again. This figure gives valuable insight in the sense that light GBM with 127 leaves and a max depth of 20 is not experiencing any overfitting problem.

Fig. 5.5 depicts the learning curve of light GBM but this time with GOSS as a boosting technique with 50 as maximum depth and 700 leaves. This figure shows that light GBM with this configuration can reach lower training/validation error compared with the previous

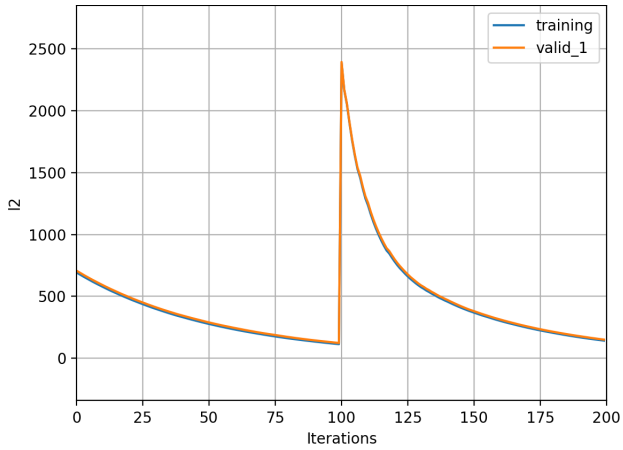


Figure 5.4: Learning curve of light GBM with GOSS boosting, 20 max depth, and 127 leaves

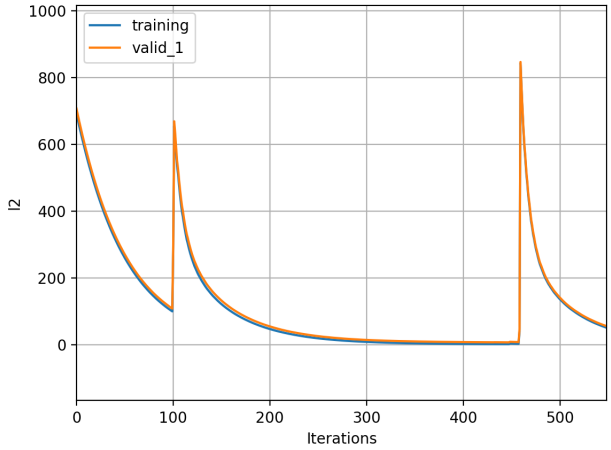


Figure 5.5: Learning curve of light GBM with GOSS boosting, 50 max depth and 700 leaves

configuration.

Fig. 5.6 depicts the learning curve of light GBM with DART as a boosting technique with 50 as maximum depth and 700 leaves. One can see that the learning curve is decreasing as the number of iteration increases to reach its lowest value at 1500 iterations, and after that the degree error does not change much. It can be seen that the training and validation degree errors are almost identical which means that the model has done a good job in generalizing the learning.

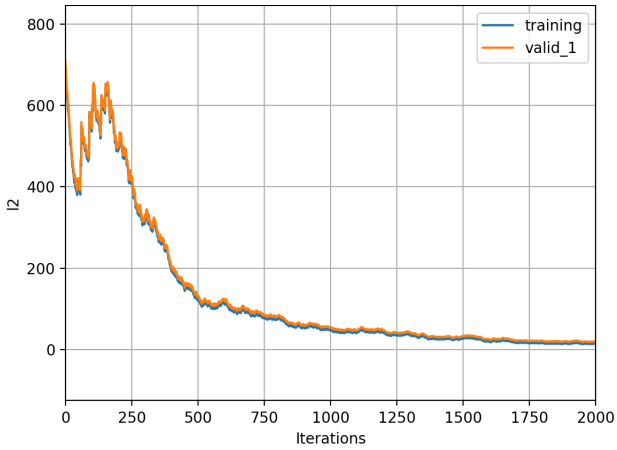


Figure 5.6: Learning curve of light GBM with dart boosting, 50 max depth and 700 leaves

Fig. 5.7 depicts the learning curve of light GBM with random forest as a boosting it can be seen that both the training and testing are decreasing as a function of the number of iterations. However, it can be seen that the training degree error is lower than the validation degree error which show that the model is experiencing overfitting. In other words, the model is not able to generalize the learning.

Fig. 5.8 depicts the learning curve of light GBM with gbdt as a boosting technique with 100 a maximum depth and 700 leaves. One can see from this figure that the training/validation degree error curves are a decreasing function of the number of iterations. The model converges at the iterations 300 as the degree error does not change as the number of iterations is increasing.

Some examples of results of light GBM model for elevation estimation with different boosting techniques are shown in Figs. 5.9 through 5.12.

Table. 5.3 depicts a performance analysis and comparison of the trained models with different boosting techniques. Specifically, this model shows the testing degree error of light GBM with DART, GOSS, Random Forest, and gbdt. It can seen from this table that light GBM with gbdt as boosting technique outperforms light GBM with the remaining boosting technique

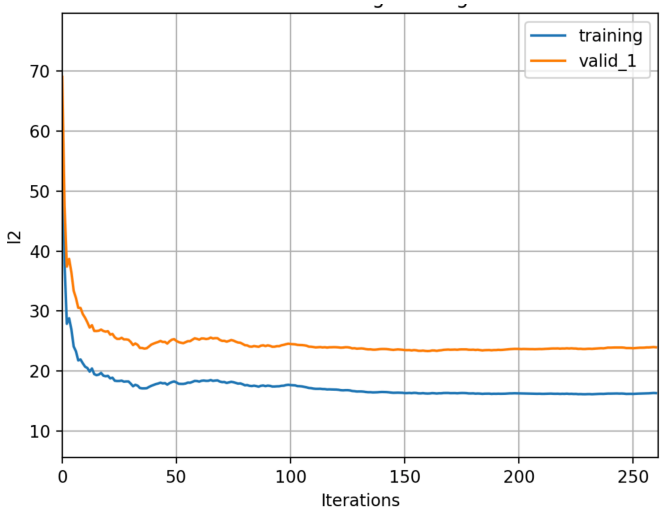


Figure 5.7: Learning curve of light GBM with random forest boosting, 20 max depth and 127 leaves

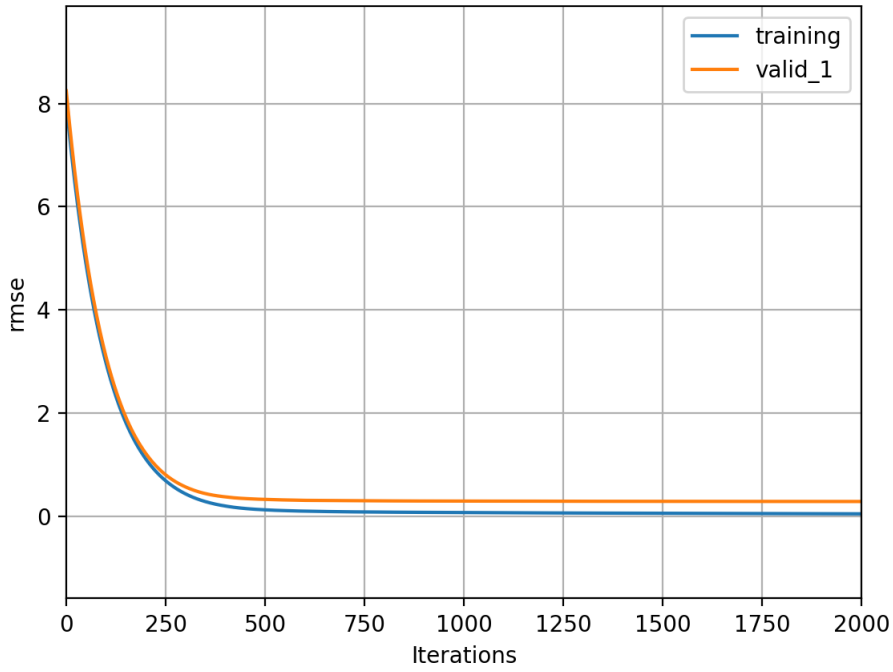


Figure 5.8: Learning curve of light GBM with gbdt, 100 max depth and 700 leaves for Azimuth

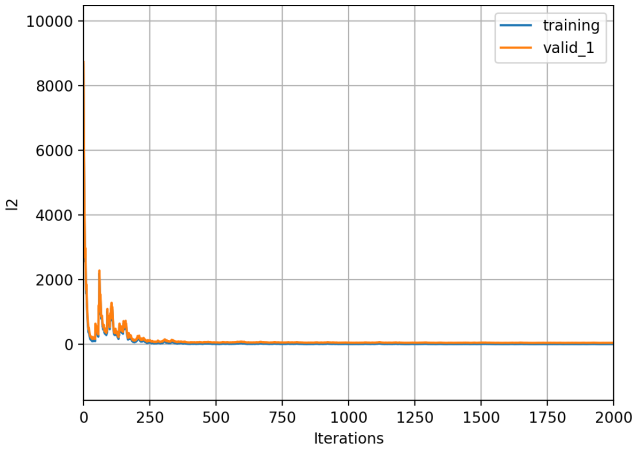


Figure 5.9: Learning curve of light GBM with DART, 100 max depth and 1000 leaves for Elevation estimation

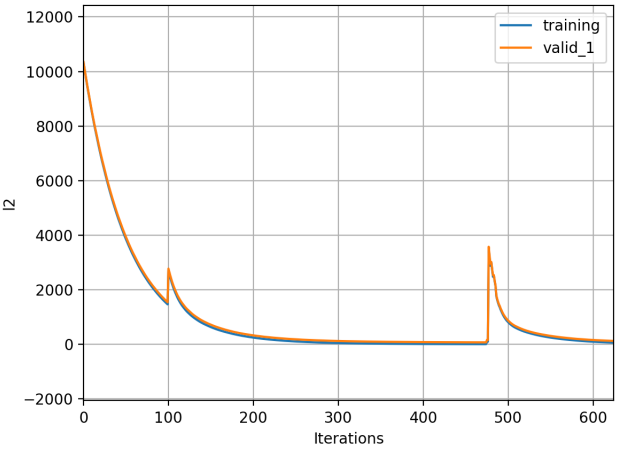


Figure 5.10: Learning curve of light GBM with RF, 20 max depth and 500 leaves, learning rate 0.01 for Elevation estimation

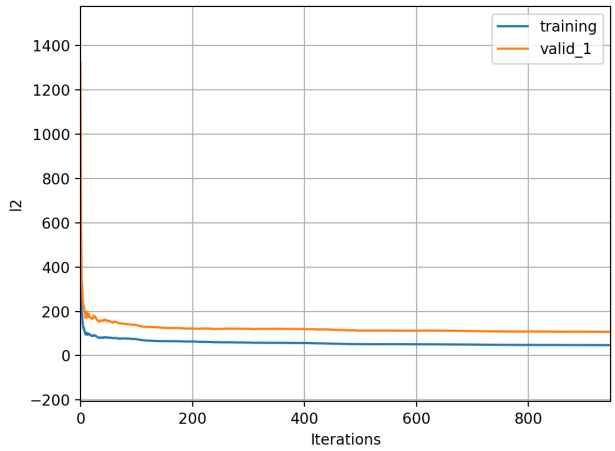


Figure 5.11: Learning curve of light GBM with RF, 100 max depth and 1024 leaves, learning rate 0.1 for Elevation estimation

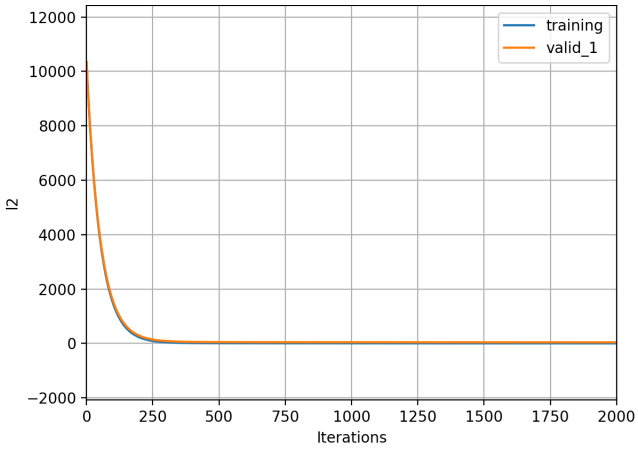


Figure 5.12: Learning curve of light GBM with gbdt, 100 max depth and 1000 leaves for Elevation estimation

Table 5.3: Performance analysis of light GBM with different configurations, Bagging frequency 10 and feature fraction 10

| | Max depth | # Leaves | # Iter | Bag Fraction | RMSE$\theta$ | MAE$\theta$ |
|---|---|---|---|---|---|---|
| DART | 50 | 700 | 2000 | 0.5 | 4. 33 | 2.00 |
| GOSS | 50 | 700 | 400 | - | 2.70 | 1.05 |
| GOSS | 20 | 127 | 400 | - | 2.84 | 1.00 |
| RF | 100 | 500 | 2000 | 0.8 | 3.18 | 1.20 |
| gbdt | 100 | 700 | 2000 | 0.8 | 2.18 | 0.74 |

Table 5.4: Performance analysis of light GBM with different configurations, Bagging Frequency is 10 and number of iteration is 2000

| Boosting Type | Max depth | No. Leaves | Feat Frac | Bag Frac | RMSE$\phi$ |
|---|---|---|---|---|---|
| DART | 50 | 700 | 0.5 | 0.5 | 4. 33 |
| GOSS | 20 | 500 | 0.8 | - | 8.68 |
| GOSS | 25 | 512 | 0.8 | - | 8.68 |
| Random Forest | 100 | 1024 | 0.8 | 0.8 | 10.40 |
| gbdt | 100 | 1000 | 0.8 | 0.8 | 6.44 |

followed with light GBM with GOSS, then Random Forest, and last DART.

Table 5.5 shows the execution time required for training and testing of the previous light GBM models. It can be seen from this Table that light GBM with GOSS and RF takes less time to be trained compared with light GBM with DART and GDBT. It can bee seen also that light GBM with random forest takes less time to estimate the DOAs of the testing set, followed by light GBM with GOSS, GBDT, and DART.

Table 5.5: The training time and testing time for light GBM based DOA estimation with different boosting algorithms

| **Model$\theta$** | **Training Time$\theta$** | **Testing Time$\theta$** |
|---|---|---|
| Light GBM (DART) | 615.987 s | 1.196 s |
| Light GBM (GOSS) | 67.207 s | 0.082 s |
| Light GBM (GOSS) | 23.339 s | 0.082 s |
| Light GBM (RF) | 37.658 s | 0.01 s |
| Light GBM (GDBT) | 397.608 s | 0.734 s |

To test the performance of light GBM in estimating AoAs in hybrid beamforming archi-

Table 5.6: The training time and testing time for light GBM based DOA estimation with different boosting algorithms

| **Model$\phi$** | **Training Time$\phi$** | **Testing Time$\phi$** |
|---|---|---|
| Light GBM (DART) | 615.987 s | 1.196 s |
| Light GBM (GOSS) | 91.909 s | 0.097 s |
| Light GBM (GOSS) | 83.947 s s | 0.095 s |
| Light GBM (RF) | 231.084 s | 0.521 s |
| Light GBM (GDBT) | 584.94 s | 1.614 s |

Table 5.7: Performance analysis of light GBM with different configurations when combined with hybrid architecture with $M = 10\%$, bagging frequency is 10

| | **Max depth** | **# Leaves** | **# Iter** | **Feat Frac** | **Bag Frac** | **RMSE$\theta$** |
|---|---|---|---|---|---|---|
| DART | 50 | 700 | 2000 | 0.5 | 0.5 | 5.79 |
| GOSS | 50 | 700 | 400 | 0.8 | - | 3.75 |
| GOSS | 20 | 127 | 400 | 0.8 | - | 5.25 |
| RF | 100 | 500 | 2000 | 0.8 | 0.8 | 6.22 |
| gbdt | 100 | 700 | 2000 | 0.8 | 0.8 | 3.17 |

tecture, we have considered random measurement matrices that model the analog combiner and precoder. Examples of the obtained results are reported in the following Figures and Tables.

Table. 5.7 depicts a performance analysis and comparison of the trained models with different boosting techniques with hybrid beamforming architecture. Specifically, this model shows the testing degree error of light GBM with DART, GOSS, Random Forest, and gbdt. It can be seen from this table that light GBM with gbdt as boosting technique outperforms light GBM with the remaining boosting technique, followed with light GBM with GOSS, then Random Forest, and last DART. In comparison to fully digital beamforming architecture where light GBM has access to the non-compressed measurements, it can be seen that most of these models experienced an increase in the degree testing error of approximately 1 to 3 degrees.

Table 5.8 shows the execution time required for training and testing of the previous light

GBM models. The execution time of all the previous models has decreased by a factor of 10. And the ranking of the models has not changed from the one shown in Table 5.5

Table 5.8: The training time and testing time for light GBM based DOA estimation with different boosting algorithms with hybrid architecture

| **Model** | **Training Time** | **Testing Time** |
|---|---|---|
| Light GBM (DART) | 62.837 s | 0.73 s |
| Light GBM (GOSS) | 6.246 s | 0.071 s |
| Light GBM (GOSS) | 4.149 s | 0.056 s |
| Light GBM (RF) | 2.896 s | 0.009 s |
| Light GBM (GDBT) | 48.244 s s | 1.307 s |

We compared the performance of the proposed algorithm to support vector regression algorithm-based DOA estimation. Support vector regression-based methods formulate the problem of DOA estimation as a regression problem and are considered as the state-of-the-art machine learning-based AoAs estimation. We have considered the same % of the split for training and testing. Examples of results are depicted below in Figs and Tables.

Figs. 5.13 shows the execution time of light GBM versus support vector regression as a function of the training size. It can bee seen that light GBM takes less time to be trained and tested.

## 5.6 Summary

The significant contribution of this chapter is deep learning and deep reinforcement learning methods for frequency-selective channels. We showed that deep learning could directly estimate the channel from the received training pilots. This method is useful when prior knowledge of the channel's sparsity is not available. We also showed that we could estimate the channel by estimating the AoAs/DoAs and the corresponding gains. We also demon-

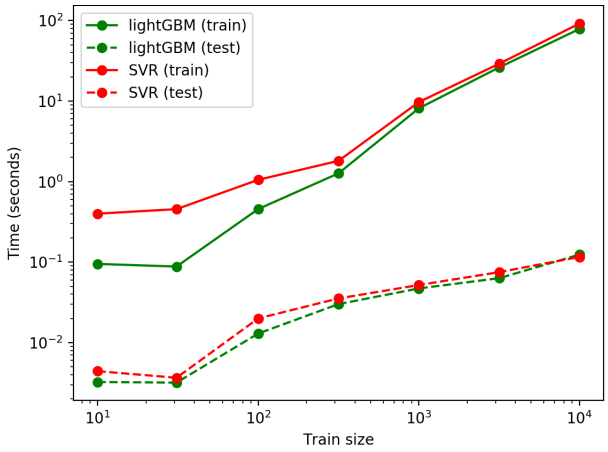


Figure 5.13: Execustion time of Light GBM based DOA estimation versus support vector regressor based DOA estimation

strated that deep reinforcement learning can achieve a close spectral efficiency compared to unconstrained precoding where a perfect channel state information is available.

# Chapter 6

# Machine Learning Based Jamming Detection

The fifth-generation New Radio (5G NR) is prone to several cyber-attacks, as indicated by many researchers and threat actors in assessing the 5G NR vulnerabilities to cyber-attacks. They showed that 5G NR is prone to intentional jamming, sniffing, and spoofing, to name a few [87, 190]. Therefore, to strengthen the physical layer security of 5G NR, it is urgent to develop detection and mitigation techniques as 5G systems are most likely to be used for public safety and emergency services[191, 192]. This chapter focuses on jamming attacks in massive MIMO 5G NR communications from assessment to detection and mitigation.

## 6.1 Jamming in 5G New Radio

Jammers are adversary wireless nodes that disrupt the existing wireless communication by causing intentional interference to decrease the signal-to-noise ratio at receiver sides [193, 194]. Jamming is an active adversary because the adversary nodes transmit energy to block data transmission. Jamming is old as wireless communication and dates back to World War II. Nevertheless, it continues to disrupt communication due to the openness of the wireless medium despite the effort to make 5G NR more resilient to denial-of-service attacks.

## 6.2 Jamming Classification

There are several types of jammers depending upon the attacking strategies. In the following, we discuss the most known jamming attacks.

### 6.2.1 Constant jammer:

In this type, the adversary node continuously transmit radio frequency signals. These signals can be either legitimate bit sequences or random bit sequences without any gap in between [193, 195, 196]. Constant jammers continuously occupy the bandwidth to starve transmissions initiated by legitimate users, therefore causing denial-of-service. This type of jamming is the most simple one, but it requires much power, draining the battery life quickly.

### 6.2.2 Deceptive jammer:

In this type, the adversary nodes continuously transmit legitimate radio frequency bit sequences on the communication channel. This jamming forces legitimate users to wait indiscriminately in the listening state by deceiving them into believing that this is a normal transmission [193, 196]. Compared with constant jammer, deceptive jammer does not transmit any random bit sequences. In addition, they are far more challenging to detect. Like a constant jammer, a deceptive jammer drains its power by continuously transmitting radio frequency signals.

### 6.2.3 Random jammer:

In this type, the adversary nodes conserve their power by randomly alternating between active state and idle state. During the active state, the adversary nodes jam for a predetermined duration before going into the idle state. During the active state, it can either follows consistent or deceptive behavior. This type of attack is more sophisticated compared with previous types. The success of this attack is related to how often the activity pattern of the jamming state follows the same activity pattern of the users [193, 195, 196].

### 6.2.4 Reactive jammer:

Constant, deceptive, and random jammers are active jammers because they attempt to block the communication channel without monitoring the activity pattern of the legitimate nodes. Alternatively, a reactive jammer, which can be considered a smarter and more power-efficient method, continually senses the communication channel to detect the activity pattern of the users before launching its attack. [195, 196, 197, 198]. Compared reactive jammers minimize power consumption because the power required for listening is far less than that required for jamming.

### 6.2.5 The follow-on jammer:

In this type, the adversary nodes hop over all available channels very frequently and jams each channel for a short period of time [193]. If a transmitter detects the presence of a jammer and switches its channel, the follow-on jammer scans the entire band and searches for a new frequency to jam again. Alternatively, it may follow a pseudo-random frequency hopping sequence. This type of jammer conserves power by limiting its attack to a single-channel

before hopping to another. Due to its high-frequency hopping rate, the follow-on jammer is particularly effective against some anti-jamming techniques, e.g., frequency hopping spread spectrum (FHSS), which uses a slow-hopping rate.

### 6.2.6 Pulsed-noise jammer:

In this type, the adversary nodes target one of multiple channels using different power levels and on different frame portions. Pulse-noise jammers produce noise pulses with different periods to avoid detection. When jamming multiple channels, it follows random patterns or a pre-scheduled hopping pattern. It can save power by continually switching between the active state and the idle state.

## 6.3 5G NR

This section briefly describes the 5G NR background, 5G NR waveform and numerology, frame structure, and channel propagation models

#### 5G NR Waveform and Numerology

To mitigate interference, 5G NR defines a new waveform. As depicted in Fig. 6.1 and similar to LTE frame structure, the 5G NR frame structure has a 10ms duration, and each frame consists of 10 subframes. Depending on the subcarrier spacing (15, 30, 60, and 120 kHz), each subframe contains either 1, 2, 4, 8, or 16 slots, and each slot consists of 14 OFDM symbols. Depending upon the subcarrier spacing, the duration of the slot varies. This new waveform and numerology enable 5G devices to operate on different bandwidths. In addition, the 5G New Radio defines two operation modes: normal cyclic prefix as shown in Table 6.1

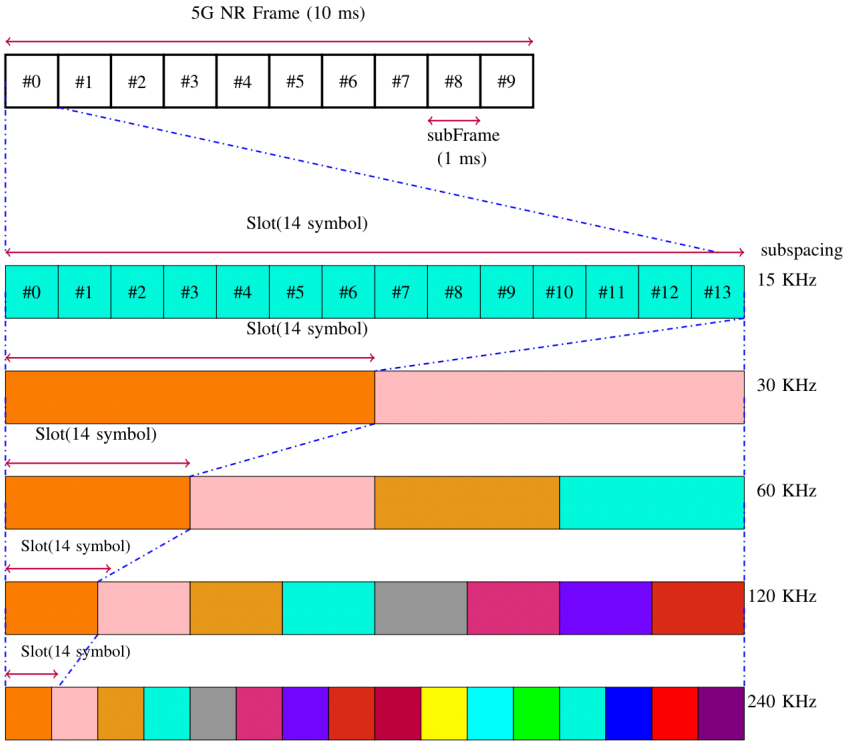


Figure 6.1: 5G NR Waveform [8]

and extended cyclic prefix as shown in Table 6.2.

In 5G NR, the physical layer defines reference signals, synchronization signals, and channels in downlink and uplink. The 5G NR contains PSS and SSS, which together are used for frame/slot/symbol timing and conveying the Physical Cell ID. There are 1008 unique Physical Cell IDs in 5G NR, the PSS has three possible combinations, and the SSS has 336 combinations. The PSS is an m-sequence of length 127, mapped to a contiguous set of 127 sub-carriers within the same OFDM symbol. The SSS also has 127, mapped to the same sub-carriers as the PSS but to a different OFDM symbol. The SSS uses a Gold sequence, which is formed by combining two m-sequences. Gold sequences within the same set have low cross-correlation, allowing a UE to distinguish between several nearby base stations on the same carrier at low SNIR, making them resilient to jamming.

Table 6.1: Normal cyclic prefix

| SCS | No. of symbols per slot | Number of slots per subframes | No. of subframes |
|---|---|---|---|
| 15 | 14 | 1 | 10 |
| 30 | 14 | 2 | 10 |
| 60 | 14 | 4 | 10 |
| 120 | 14 | 8 | 10 |

Table 6.2: Extended cyclic prefix

| SCS | No. of symbols per slot | Number of slots per subframes | No. of subframes |
|---|---|---|---|
| 60 | 12 | 4 | 10 |

## 6.4 Overall Vulnerability Assessment

### 6.4.1 Jamming Vulnerability of the PBCH:

PBCH are assigned symbols within two to four slots of each other (depending on whether the carrier is below or above 3 GHz, respectively), so a selective jammer targeting the PBCH will appear to have a very low duty cycle. As higher the sub-carrier spacing, the duration of one slot is lower, and the selective jamming duty cycle decreases.

#### Jamming Vulnerability of PDCCH:

Jamming the PDCCH channel is far more complex than jamming PBCH. In order to jam PDCCH, the jammer has to jam all the possible locations in which the PDCCH resides, assuming that the jammer does not have any knowledge of the CORESET freq-domain. However, the jammer can intercept and decode the CORESET freq-domain. This gives the jammer an advantage to jam every sub-carrier, using a duty cycle of less than 21% depending upon the value of CORESET-time-duration.

**Jamming Vulnerability of Physical Uplink Control Channel:**

UE uses the PUCCH channel to convey to the gNB a variety of control information, including HARQ ACKs, SRs, and CSI. There are many different PUCCH formats, and there are a variety of parameters provided by the higher layer to communicate with the UE, which subcarriers and symbols to transmit each PUCCH message. The PUCCH has an option for intra-slot hopping, which can provide some protection against selective jammers, but the greatness of this defense mechanism is dependent on the rate of the hopping, and this information is available to the jammer as the 5G standard is public. The PUCCH is modulated with either BPSK or QPSK. It uses either repetition codes or polar codes as error coding schemes depending on the number of bits to transmit, but these schemes do not protect against jamming.

### 6.4.2 Jamming Vulnerability Physical Random Access Channel

When a UE wants to connect to a cell, it first receives the PSS, SSS, and PBCH. After synchronizing to the cell in time and frequency, the next step in the 5G NR random access procedure is the uplink transmission of a random access preamble by the UE on a dedicated random access channel (RACH). After the reception of a preamble, a gNB estimates temporal synchronization parameters and allocates radio resources for further communication with the UE. The synchronization parameters and allocation of radio resources are then communicated to the UE that initiated the RA procedure within a specific time after the RA preamble transmission. This information is broadcasted on PRACH, which takes the form of a Zadoff-Chu sequence that embeds a value used to identify the UE temporarily. Despite a large number of possible locations and the complexity required to decode the positions in real-time, jamming PRACH is feasible, and in case the jammer could not succeed

in determining the positions, it can flood the with invalid preamble as the 5G NR does not specify what should be the behavior of the Gnb in this case.

### 6.4.3 Jamming Vulnerability Massive MIMO

5G NR, as we discussed, is going to use massive MIMO. Many research studies indicated that massive MIMO systems are vulnerable to jamming attacks. Jamming in MIMO systems targets the channel estimation of these systems. By targeting the channel estimation procedure, an adversary may launch effective jamming attacks against unsuspecting users. The authors of [18] presented several jamming methodologies for SVD-based MIMO systems, including a powerful and practical channel rank attack. Further, they presented strategies to undermine the popular Alamouti STBC-based MIMO scheme. Such attacks have been proven feasible by way of analysis, simulations, and real-world experimentation. Additionally, the attack strategies presented have been general and may be applied to other MIMO systems. Overall, protecting channel estimation is essential to achieve the gains associated with MIMO systems, and thus future directions for exploration should involve finding methods that reliably protect channel state information.

### 6.4.4 Polar coding and LDPC coding

Most of the control channels use polar coding as an error coding scheme, and the data channels will use LDPC coding. It has been shown that polar coding is vulnerable to jammers.

## 6.5 Detection and mitigation

### 6.5.1 Detection

Detection of smart-jamming attacks is feasible by monitoring for an excess amount of energy on any one specific physical channel (e.g., using masking). Common among these metrics are packet delivery ratio (PDR), packet sending ratio (PSR), bad packet ratio (BPR), bit error rate (BER), energy consumption amount (ECA), signal-to-noise ratio (SNR), clear channel assessment, received signal strength. PDR is the ratio of the number of packets that have been successfully delivered and acknowledged by the destination node to the number of packets sent by the transmitting node. BPR is the ratio of the number of damaged packets received by a node to the total number of packets received in a given period. SNR can be obtained by finding the ratio of the received signal power to the received noise power at the node. The bit error rate can be determined by computing the number of corrupt bits to the entire bits received during a transmission session. A common strategy in jamming detection is the use of a threshold with some performance metrics, such as the packet delivery ratio, clear channel assessment, the received signal strength indicator, idle time. These techniques monitor the level of these metrics during the absence and the presence of jamming attacks and set the threshold for detection manually. Another statistical approach detects different forms of jamming attacks. Machine learning techniques have been proposed to detect jamming. Several machine learning techniques such as random forest, decision tree, adaptive boosting, support vector machine, and expectation-maximization are investigated in detecting jamming attacks.

## 6.6 Jamming System Model and formulation

Detection of jamming in 5G NR is modeled as a classification problem where the machine learning model has to be trained to detect the presence of the jammer and classify the incoming signal into jammed or unjammed signal.

Consider a hybrid mmWave MIMO multiuser system discussed in the previous chapter 3. In addition, the system operates where there are J jammers transmitting interfering signals to degrade the target signal-to-noise ratio.

At the base station, the transmitted symbols are processed using a beamformer $F_T$ where the beamformer in the case of hybrid architecture can be divided into $F_{BB}$ and $F_{RF}$. The receiver is equipped with a combiner $W_T$ ($W_T = W_{BB}^H W_{RF}^H$ for hybrid architecture)

The $k^{th}$ user receiver a signal modeled as:

$$y_k = W_{T_k} H_k F_{T_k} s_k + \sum_{K}^{i \neq k} W_{Tk} H_k F_{T_i} s_i + \sum_{J}^{jk} W_{Tk} H_k Z_{k,j} x_j + W_{T_k}^H n_k \tag{6.1}$$

where $x_j$ represents the jamming signal of the $j^{th}$ jammer assuming $E[x_j x_j^*] = P_j$ where $P_j$ denotes the power of transmission of the $j^{th}$ jammer.

or in the case of the hybrid architecture,

$$\begin{aligned} y_k = W_{BB_k}^H W_{RF}^H H_k F_{RF} F_{BB_k} s_k + \underbrace{\sum_{i \neq k}^{K} W_{BB_k}^H W_{RF}^H H_k F_{RF} F_{BB_i} s_i}_{intra-cell interference} + \\ \underbrace{\sum_{jk}^{J} W_{BB_k}^H W_{RF}^H H_k Z_{k,j} x_j}_{Jamming} + \overbrace{W_{BB_k}^H W_{RF}^H n_k}^{noise} \end{aligned} \tag{6.2}$$

or in the case of the hybrid architecture, under no jamming attacks, the K-th

$$y_k = W_{BB_k}^H W_{RF}^H H_k F_{RF} F_{BB_k} s_k + \underbrace{\sum_{i \neq k}^{K} W_{BB_k}^H W_{RF}^H H_k F_{RF} F_{BB_i} s_i}_{intra-cell interference} + \overbrace{W_{BB_k}^H W_{RF}^H n_k}^{noise} \tag{6.3}$$

where $Z_{k,j} = \beta_{k,j} A_{k,j}$ is the precoder of the $j^{th}$ jammer, $\beta_{k,j}$ denotes the interfering channel gain, and the $A_{k,j}$ denotes the linear array response vector of the interfering channel.

Detection of jamming attacks consists of classifying the received signal into normal signal and jammed signal.

## 6.7 Jamming Detection Based on Real-time Machine Learning

Consider the training dataset $D = [(y_1, c_1), (y_2, c_2, ...., y_M, c_M)$ where $y_i$ denotes the received signal and $c_i$ denotes the class label of the $i^{th}$ element of the dataset $D$. $y_i$ is vector of features that can be the received samples (raw data) or features extracted from the received samples.

### 6.7.1 Decision-tree based Jamming Detection

Decision tree build the tree by selecting the root node, which corresponds to the most important attribute in the dataset. The selection of the root node is the most challenging. Several criteria are used including information gain (IG) and Gini-Index (GI). Information Gain measures the information contained in each attribute using the entropy.

Entropy measures the impurity in the given dataset, and it is given by,

$$Entropy(D) = -\sum_{i=1}^{C} p_i log_2(p_i) \tag{6.4}$$

where $C$ is the number of classes and $p_i$ denotes the probability associated with the $i^{th}$ class.

In this model, $C = 2$ as we have only two class labels: $c_1 = \{NormalSignal\}$ and $c_2 = \{JammedSignal\}$.

$$Entropy(D) = -p_1 log_2(p_1) - p_2 log_2(p_2) \tag{6.5}$$

And the entropy of $D$ after splitting on the attribute $A$ is given by

$$H(D, A) = -\sum_{c} \frac{H(D_a)|D_a|}{|D|} \tag{6.6}$$

where $D_a$ is the subset of $D$ where $A$ has value $a$ .

The information gain of A is then

$$IG(D, A) = H(D) - H(D, A) \tag{6.7}$$

Gini impurity reduction is the difference between the Gini index before and after splitting by the attribute [25]. This measure was made popular after the CART decision tree algorithm. The Gini index of a random variable C is another nonlinear measure of the dispersion of C, defined as.

$$G(C) = 1 - \sum_{i=1}^{C} (p_i)^2 \tag{6.8}$$

where $C$ is the number of classes and $p_i$ is the probability associated with the $i^{th}$ class.

**Algorithm 7:** Decision Tree [165, 199]

```
Training data set D,S ≠ null, Number_of_att > 0
Initialize: MaxGain ⟵ 0
Initialize: e ⟵ Entropy(attributes)
while All partition processed do
    Data: Testing set x
    for attributes a in D do
        Gain ⟵ InformationGain(a, e)
        if Gain > MaxGain then
            MaxGain ⟵ Gain
            SplitA ⟵ a
        partition (D,SplitA )
                    // This completes the building of the decision Tree model
Decision Tree
```

### 6.7.2 Bagging Decision Tree based Jamming Detection

Given the training set $D = [(y_1, c_1)..., (y_M, c_M)]$, bagging repeatedly selects a random subset with replacement of the training set and fits trees to these samples. If the bagging select repeats this process B times, then for each time, the bagging classifier samples, with replacement, $N$ training examples from $D$; denoted $D_b$. The classifier then fits a decision tree called $T_b$ on the training dataset $D_b$. After training, predictions are performed on unseen samples $D'$ can be made by taking the majority vote on the predictions from all the individual trees tested on $D'$:

$$\hat{t} = \frac{1}{B} \sum_{b=1}^{B} T_b(D') \tag{6.9}$$

This bootstrapping yields a better model performance due to the decrease in the variance of the model, without increasing its bias.

### 6.7.3 Random Forest based Jamming detection

Random forest is an ensemble method that modifies the previous bagging algorithms to select subsets that are uncorrelated. It uses random subsets. Each Tree takes all the features but each node takes only the random subset features. Since jamming detection is a classification problem, then the training dataset $D$, which has $p$ element has been split into down rounded $\sqrt{p}$. This split can be considered as a tuning parameter.

### 6.7.4 Adaptive Boosting Based Jamming Detection

Adaptive boosting (AdaBoost) is another decision tree based models

### 6.7.5 Hoeffding Decision Trees for Jamming Detection

Decision trees are required to reuse instances to compute the best splitting attributes, which is not possible when it comes to stream settings. To overcome this limitation, the authors of [201] proposed a new algorithm based on decision trees, called the Hoeffding tree. This algorithm is high-speed and suitable for streaming data. The authors showed that it is possible to construct decision trees without storing all the instances. The pseudocode of the Hoeffding tree algorithm is shown in Algorithm 4.

The Hoeffding tree introduces the Hoeffding bound, which helps in deciding how many instances are required to achieve a certain level of confidence [201]. This bound states that with probability $1 - \delta$, the true mean of the variable is at least $\bar{r} - \epsilon$. Moreover, $\epsilon$ is given by the following formula:

$$\epsilon = \sqrt{\frac{R^2 ln(1/\delta)}{2n}} \tag{6.10}$$

**Algorithm 8:** Random Forest Algorithm [164, 165]

Training data set D **Initialize:** A the feature space
**Initialize:** C the feature space
**Initialize** The number of Trees K
**Initialize** The size of the subspaces
**for** *index is in range (1, $n - Trees$)* **do**
    Draw a bootstrap sample $D_index$ from the training set

        `// Agent is playing several episodes to generate a training set`
    create decision tree $h_{index}(D_index) = CreateTree(D_index)$
    `/* Agent learning */`
    calculate target using 3.52
    Compute critic loss
    Optimize the critic
Combines the K tree classifiers $(h_1(D_1), h_2(D_2), ..., h_K(D_K))$ into a random Forest.
`// This completes the building of the random forest model`
**Function** createTree()
rootNode = createNode()
**return** rootNode
**Function** createNode()
create a new node $\eta$
**if** *stopping criteria is met* **then**
    return $\eta$ as a leaf node
**else if** *Condition 2* **then**
    **for** $j = 1$ *to* $M$ **do**
        compute the informativeness measure corr(Aj,Y) by Equation of the information gain ratio
    compute feature weights $w_1, w_2, ..., w_M$ by Equation (12)
    use the feature weighting method to randomly select $m$ features
    use these $m$ feature as candidates to generate the best split for the node to be partitioned
    call createNode() for each split
a Random Forest

where $R$ denotes the range of $r$, the real-valued random variable; $n$ denotes the number of independent observations; and $\bar{r}$ denotes the mean value computed from n independent observations

**Algorithm 9:** Adaboost [200]

**Initialize:** weights $w_i = 1/N$, $i = 1, 2, ..., N$
**Data:** Training data set D,confidence parameters
**for** *m=1 :1: M* **do**
    fit the classifier $f_m(x) \in \{-1, 1\}$ using weights $w_i$ on the training dataset
    Compute error using $err_m = E_m[1_{y \neq f_m(x)}]$
    compute $c_m = log(\frac{1 - err_m}{err_m})$
    set $w_i \longleftarrow w_i Exp(c_m 1_{y \neq f_m(x)})$, $i = 1, 2, ..., N$
    normalize the weights so $\sum w_i = 1$
    Output the classifier $sign[\sum c_m f_m(x)]$

`// This completes the building of the Ada boost model`

Adaboost

**Algorithm 10:** Hoeffding tree [201]

**Initialize:** counts for $n_{ijk}$ at root
**Initialize:** HT with a single leaf (root)
**Data:** A stream of labeled examples, confidence parameters
**for** *each example (x,y) in Stream* **do**
    HTGrow( HT, $\delta$)
**Function** HTGrow( HT, $\delta$):
sort(x,y) to leaf $l$ using HT
update counts $n_{ijk}$ at leaf $l$
**if** *examples seen so far at l are not all of the same class* **then**
    Compute G for each attribute
    **if** $G(best\ attribute) - G(second\ best) > \sqrt{R^2 ln(1/\delta)/2n}$ **then**
        split leaf on best attribute
        **for** *each branch* **do**
            start new leaf and initialize counts

`// This completes the building of the Ada boost model`

Adaboost

## 6.8 Illustrative Results and Discussions

We have built a training dataset from a simulation setup in which we have simulated communication between legitimate users considering a Massive MIMO system where the signal-

to-noise ratio varies between $-5dB$ and $5dB$. We have defined the physical downlink shared channel (PDSCH) of 5G NR as defined in 3GPP NR standard. The channel is encoded and modulated, and transmitted over a clustered delay propagation channel. The example implements PDSCH and downlink transport channel DL-SCH. The transmitter model includes PDSCH demodulation reference symbols (DM-RS) and synchronization signal (SS) bursts. We have considered a CDL channel model studied for channel model between 0.5GHz and 100GHz in [202]. At the receiver, we have assumed a perfect channel estimation model. The simulation parameters that were considered include the bandwidth in resource blocks, the Subcarrier spacing, the cyclic prefix length, the number of transmitting and receiving antennas (8, 16, 64, 128), the modulation scheme: "QPSK", "16QAM", "64QAM", "256QAM". To evaluate the performance of jamming detection based on decision trees, we considered the most common evaluation metrics such as the detection rate, the false alarm rate, the miss-detection rate, and the accuracy [203]. To compute these rates, we generated the confusion matrix (Table 6.3).

Table 6.3: Confusion matrix

| | **Positive(1)** | **Negative(0)** |
|---|---|---|
| **Positive(1)** | TP | TN |
| **Negative(0)** | FP | FN |

The detection rate defines the probability that the signal is actually jammed, and the model outputs a jammed signal [27]. This detection rate can be calculated from the confusion matrix using the following formula:

$$detection\ rate = \frac{TP}{TP + FP} \tag{6.11}$$

The false alarm defines the probability that the jammer is not present, and the model declares the presence of a jammer [28]. The false alarm can be derived from the confusion matrix

using the following formula:

$$false\ alarm = \frac{FP}{TP + FP} \tag{6.12}$$

The miss detection rate is the probability that the received signal is jammed, and the model did not detect that [29]. The miss-detection rate is derived from the confusion matrix using the following formula:

$$miss - detection\ rate = \frac{TN}{TP + TP} \tag{6.13}$$

$$Accuracy(\%) = \frac{True\ Positives + False\ negatives}{Total\ Number\ of\ Samples} \tag{6.14}$$

Area-under-Curve (AUC) is the probability that the classifier ranks a randomly chosen positive example higher than a randomly chosen negative example. Equivalently, it is defined as the area under the curve that plots the sensitivity (True Positive Rate) versus the specificity (False Positive Rate). The higher the value of AUC, the better the performance of the classifier.

The time required to train the machine learning to detect jamming attacks is another critical metric to evaluate the performance of the machine learning techniques.

### 6.8.1 Numerical Results

Examples of results are depicted in Figs. 6.2 through 6.7 and Tables 6.4 through 6.5. Fig. 6.2 illustrates the ROC curve of the decision tree, bagging decision tree, random forest, grid search, gradient boosting, Ada boosting, XGB boosting, and XG grid search. The number

of samples considered here is 512. This figure indicates that decision tree and gradient boosting techniques have poor performance, followed by grid search, random forest, Ada boosting, and bagging decision tree. This figure also indicates that XGB Boosting and XG grid search have the best performance.

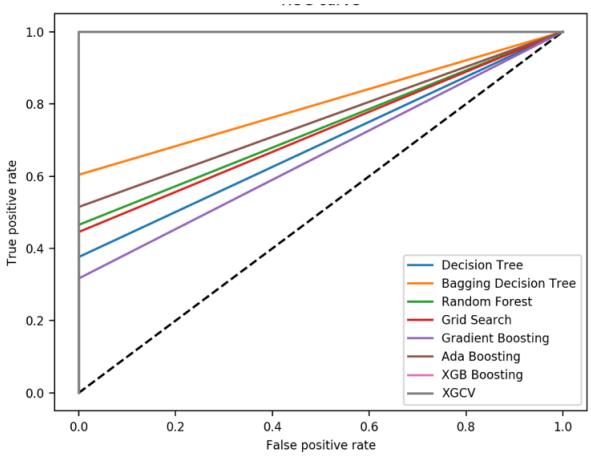


Figure 6.2: ROC curve of decision trees using 512 samples from noiseless observations

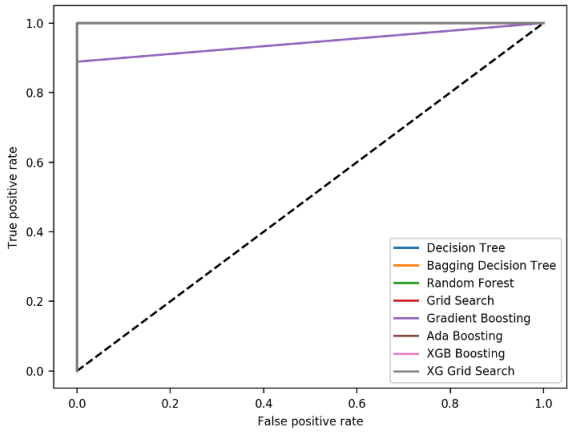


Figure 6.3: ROC curve for decision trees for 1024 from noiseless observations

Fig. 6.3 shows the ROC curve of all the previous techniques when a total number of 1024 is used. This figure indicates that the detection performance of all the previous algorithms have been improved as the number of samples increases to achieve 100% detection rate except gradient boosting, which performed poorly compared with all the other techniques.

To study the impact of noisy measurements on the detection performance of these algorithms, we have considered noisy signals with 512 and 1024 samples. Examples of results are shown in Fig. 6.4 and Fig. 6.5. These two figures display the ROC curve of the above-mentioned algorithms. One can see that the XGB and gradient boosting are more robust to noise than other techniques.

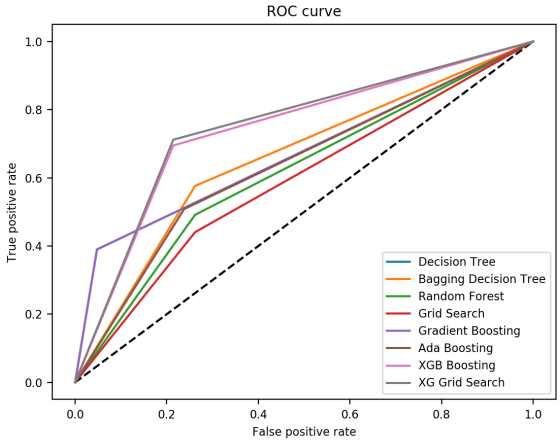


Figure 6.4: ROC curve of decision trees using 512 samples from noisy observations

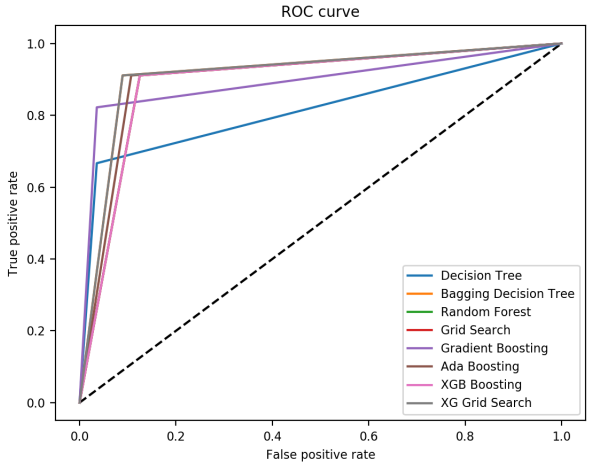


Figure 6.5: ROC curve for decision trees for 1024 from noisy observations

Table 6.4 depicts the jamming detection accuracy (%) as well as the training time (s) for 512 samples and 1024 samples. The decision tree achieves an accuracy of 61.88%, 100% with

512 and 1024, respectively. Bagging and boosting algorithms have an accuracy of around 75% and 99% for 512 and 1024, respectively. XGB boosting and XG grid search have 100% accuracy for 512 and 1024. The results shown in this table reflect what was reported in the previous Figures. This table also indicates the training time required for each one of these algorithms with 512 and 1024 samples. The decision tree takes less than 0.022 $s$ to be trained while boosting techniques take between 1$s$ and 9.169$s$. Grid search-based algorithms take up to 21$s$, and XG Grid search takes up to 585$s$. We can see that the algorithm that performed well takes more processing time.

Table 6.4: Detection performance of the models tested on noiseless signals

| | **DT** | **Bagging DT** | **RF** | **GBoost** | **AdaBoost** | **XGBoost** | **Grid Search** | **XG Grid Search** |
|---|---|---|---|---|---|---|---|---|
| **512 Samples** | 61.88% | 77.72% | 73.26% | 60.89% | 73.76 | 100% | 72.27% | 100% |
| **Training time** | 0.022s | 25.928s | 1.639 s | 1.894 s | 4.262 s | 0.343 s | 11.58 s | 339.49 |
| **1024 Samples** | 100% | 100% | 100% | 98.08% | 99% | 100% | 100% | 100% |
| **Training time** | 0.03 s | 25.693 s | 1.365 s | 1.791 s | 9.169 s | 1.457 s | 20.18 s | 584.07 s |

Table 6.4 shows the detection performance, but this time using noisy signals. It can be seen from this Table that the detection performance for most algorithms dropped when the number of samples was 512 samples. For 1024, even though detection performance dropped, most algorithms show an accuracy higher than 80%.

Fig. 6.6 shows the performance of the Hoeffding tree, and Fig. 6.7 shows the performance of the Hoeffding adaptive tree. These algorithms achieve an accuracy of 82.09% and 75.74% in real-time and process one instance at a time. These two figures also indicate that HT and HAT accuracy dropped when instances selected from the stream are jammed signals. This

Table 6.5: Detection performance of the models tested on noisy signals

| | DT | Bagging DT | RF | GBoost | AdaBoost | XGBoost | Grid Search | XG Grid Search |
|---|---|---|---|---|---|---|---|---|
| **512 Samples** | 62.4% | 64.36% | 59.40% | 59.40 % | 61.38 % | 73.26% | 56.43% | 74.26% |
| **Training time** | 0.017 s | 28.423 s | 1.58 s | 1.513 s | 4.262 s | 0.877 s | 14.992 s | 498.693 s |
| **1024 Samples** | 83.17% | 91.1% | 89.11% | 89.11% | 90% | 89.11% | 89.11% | 91.1% |
| **Training time** | 0.03 s | 25.433 s | 1.529 s | 3.132 s | 0.6 s | 0.563 s | 14.52 s | 681.749 s |

can be explained by the fact that the models have not yet learned the features of the jammed signal class. As the models are exposed to more instances from this class, they become more accurate.

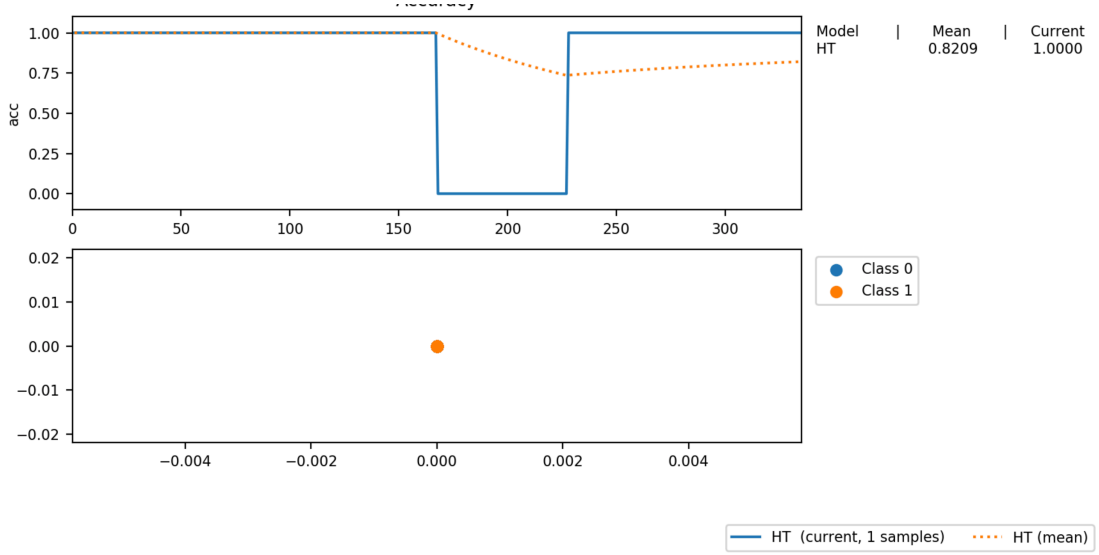


Figure 6.6: Accuracy of HT as function of the number of samples

## 6.9 Modified Code Division Multiple Access

Spread-spectrum is one technique used to design secure communication in an adverse environment [88]. Spread-spectrum rejects interference, which can be either intentional, such as jamming signals, or unintentionally caused by other legitimate users transmitting simulta-

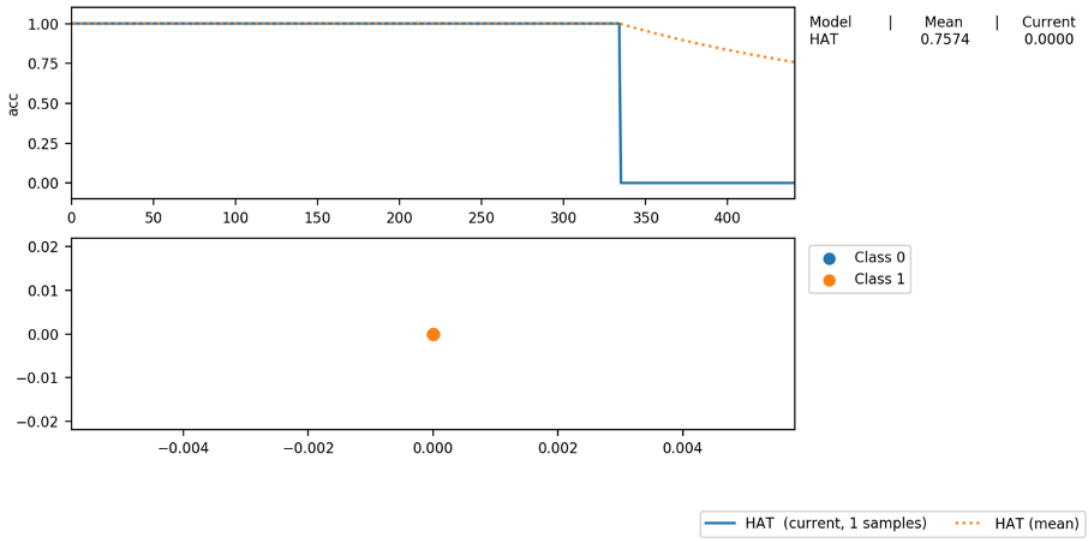


Figure 6.7: Accuracy of HAT as function of the number of samples

neously through the channel.

In spreading-spectrum, information sequence is expanded to occupy a bandwidth far more significant than the minimum required. Spread-spectrum can be accomplished by multiplying each bit of the data sequence by a pseudo-noise (PN) code independent of that sequence. The receiver uses the same code to despread the data [204].

Spread-spectrum uses orthogonal codes for spreading. Examples of orthogonal codes are Walsh codes, which J. L. Walsh initially developed in the early 1900. These codes are well-known for their orthogonal properties. They have been used in code multiple division access (CDMA) for spreading and user ID. Several authors have also investigated the use of orthogonal codes for forward error control coding. They have shown that orthogonal codes are bandwidth inefficient.

This section examines the error control properties of orthogonal codes to generate a robust and low-cost channel coding technique in CDMA systems. By exploiting the distance properties of orthogonal codes, one can show that CDMA can detect and correct errors with

bandwidth efficiency. We note that an n-bit orthogonal code has $n/2$ 1's and $n/2$ 0's; i.e., there are $n/2$ positions where 1's and 0's differ. Therefore the distance between two orthogonal codes is also n/2, where each orthogonal code generates a zero parity bit. On the other hand, the distance between an orthogonal code and its complements is d=n. These properties are used to show that orthogonal codes and their complements can be used to detect and correct errors in code multiple division access without the use of additional blocks for error coding.

We also show that the proposed technique has many advantages, especially robustness against opposing jammers who target the communication channel. Finally, constructions of CDMA schemes, along with BER performances, are presented to illustrate the concept.

### 6.9.1 CDMA: Related Work

CDMA or spread-spectrum has been investigated by many researchers over the past two decades. Subsequently, several authors proposed modifications to CDMA systems either to make it bandwidth efficient or to enhance its function [205, 206, 207, 208, 209]. For instance, the authors of [207] conducted a security analysis along with enhancement of CDMA systems. The authors investigated the weaknesses of CDMA systems that use scrambling methods. The authors showed that the CDMA PN codes can be cracked in real-time and that the eavesdroppers only need access to 42-bits for the polynomial characteristic is a public standard. To enhance the physical security capability of CDMA systems, the authors proposed the enhancement of the scrambling process AES encryption. A similar approach has been proposed by the authors of [206], in which they proposed the use of an advanced encryption standard (AES) based scrambling process instead of using the scrambling sequence generated from the 42- bit linear feedback shift register (LFSR). The authors of [208] investigated

CDMA capacity under disguised jamming, in which the fake signal is generated using the same characteristics of the authorized signal such as the spreading code, constellation, and pulse shaping filter. Based on the study conducted by the authors of [209], disguised jamming, using the same level of power as authorized signal and mimicking the pulse shape and space coding of the authorized signal, can be more harmful than strong Gaussian jamming attacks. The authors specified that disguised jamming happens at the symbol level and is immune to bit-level channel coding. Thus reducing the code rate does not improve the performance of the link under jamming attacks.

### 6.9.2 Construction of Orthogonal codes

Orthogonal codes can be generated by transformation of $N \times N$ Hadamard matrix. The first matrix in a Hadamard transform is given by:

$$H1 = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \tag{6.15}$$

The second matrix uses the first matrix

$$H_2 = \begin{bmatrix} H_1 & H_1 \\ H_1 & \overline{H_1} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \tag{6.16}$$

Each row of the Hadamard matrix is a unique orthogonal code. In order to construct 8-orthogonal codes, we need to transform the matrix $H_3$ defined as:

$$H_3 = \begin{bmatrix} H_2 & H_2 \\ H_2 & \overline{H_2} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \end{bmatrix} \tag{6.17}$$

$$H_4 = \begin{vmatrix} H_3 & H_3 \\ H_3 & \overline{H_3} \end{vmatrix} \tag{6.18}$$

The cross-correlation of Walsh cod with another is always making them possess excellent orthogonality property.

### 6.9.3 Error Control Properties of Orthogonal Codes

Orthogonal codes are defined as codes that have as many ones as many zeros in them. For instance, an m-bit orthogonal code consists of $m/2$ ones and $m/2$ zeros. An interesting property of orthogonal codes is that the distance between two orthogonal codes is $d_{m,m} = \frac{m}{2}$. While the distance between an orthogonal code and its complement is $d_{m,m^c} = m$. These properties are used for channel coding and error detection. As depicted in Fig.6.8, by setting a threshold midway between an orthogonal code and its complements, one can detect errors

in the received bit sequence where the received code is shown as a dotted line. This is given by:

$$T_m = \frac{m}{2} \tag{6.19}$$

Where $m$ denotes the code length, and $T_m$ is the threshold, midway between an orthogonal code and the antipodal code. This threshold is essentially another orthogonal code. Therefore, for instance, for a given 4-bit orthogonal code, we have

$$T_4 = 4/2 = 2 \tag{6.20}$$

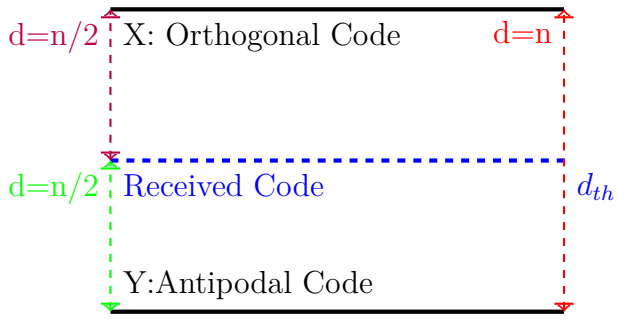


Figure 6.8: Distance between an orthogonal code and its complements. The complementary code is essentially the inverted code

The received code is examined for correlation with the neighboring codes for a possible match. The acceptance criterion for a valid code is that an n-bit comparison must yield a good auto-correlation value; otherwise, a false detection will occur. The following correlation process governs this, where an impaired received code is compared with the pair of complementary codes for a possible match to yield,

$$R(x, y) = \sum_{i=1}^{n} x_i y_i >= (n - d_{th}) + 1 \tag{6.21}$$

Where x is the desired orthogonal code, y is the impaired received code, R(x,y) is the

Table 6.6: Error Correction Capabilities of Orthogonal Codes

| Code Length | Number of Errors Corrected t |
|---|---|
| 4 | 1 |
| 8 | 3 |
| 16 | 7 |
| 32 | 15 |
| 64 | 31 |

correlation function, n is the code length, and dth is the threshold as defined earlier. Since the threshold is midway between two valid complementary codes, an additional 1-bit offset is added to equation-2 for reliable detection. The average number of errors that can be corrected by means of this process can be estimated by combining equations 1 and 2, yielding,

$$t = n - R(x, y) = \frac{n}{2} - 1 \tag{6.22}$$

In equation.3, $t$ is the number of errors that can be corrected by means of an n-bit orthogonal code. For example, a single error-correcting orthogonal code can be constructed by means of a 4-bit orthogonal code (n = 4). Similarly, a three-error-correcting, orthogonal code can be constructed by means of an 8-bit orthogonal code (n = 8). Likewise, a seven error-correcting orthogonal code can be constructed using a 16-bit orthogonal code and so on. Table-1 below shows a few orthogonal codes and the corresponding error-correcting capabilities.

From the above table, we see that approximately 50% errors can be corrected using orthogonal codes

## 6.10 CDMA Capacity

In CDMA, each user has (N-1) sources of interference where N is a total number of users. Thus, the interference is given by:

$$I = C(N-1) \tag{6.23}$$

which can be written as

$$\frac{C}{N} = \frac{1}{N-1} \tag{6.24}$$

Where $\frac{C}{N}$ denotes the carrie-to-interference ratio which can be described also using

$$\frac{C}{I} = \frac{R_b \times E_b}{N_o \times W} \tag{6.25}$$

where

- $R_b$ denotes the bite rate
- $E_b$ denotes the energy per bit
- $N_o$ denotes the thermal noise
- $W$ denotes the bandwidth after spreading

from equations (6.24) and (6.25), it follows that

$$\frac{1}{N-1} = \frac{R_b \times E_b}{N_o \times W} \tag{6.26}$$

solving this equation, we get

$$N = 1 + \frac{W/R_b}{E_b/N_o} \tag{6.27}$$

and since the Bandwidth $W = 2R_c$, then we substitute $W$ in (6.26), we get

$$N = 1 + \frac{2R_c/R_b}{E_b/N_o} \tag{6.28}$$

Thus, we get

$$N = 1 + \frac{2n}{E_b/N_o} \tag{6.29}$$

where n is the spreading factor length.

### 6.10.1 Illustrative Results and Discussions

Examples of results are shown in Fig. 6.9. This figure shows the theoretical and simulated bit error rate as a function of SNR for different M-CDMA for different code lengths: 8, 16, 32, and 64. From this figure, it can be seen that as the PN code's length increases, the BER decreases. As the SNR increases, the BER decreases. For instance, the BER is of the order of $10^{-20}$ at SNR of $0dB$ for PN code of 32. This scheme can be used for applications with low tolerance to BER, such as 4K and 8K video streaming. We can also see that the theoritical and simulated results are very close (2dB).

Fig. 6.10 shows the BER versus SNR for two scenarios under jamming and under no jamming using MCDMA. It can be seen that the jammer requires more power to cause denial-of-service. This figure shows that under no jamming attacks, the BER is lower than $10^{-20}$ for SNR higher than $7dB$. As we introduce a jamming signal, we can see that the BER starts increasing depending on the code length of the MCDMA and the power used by the jammer.

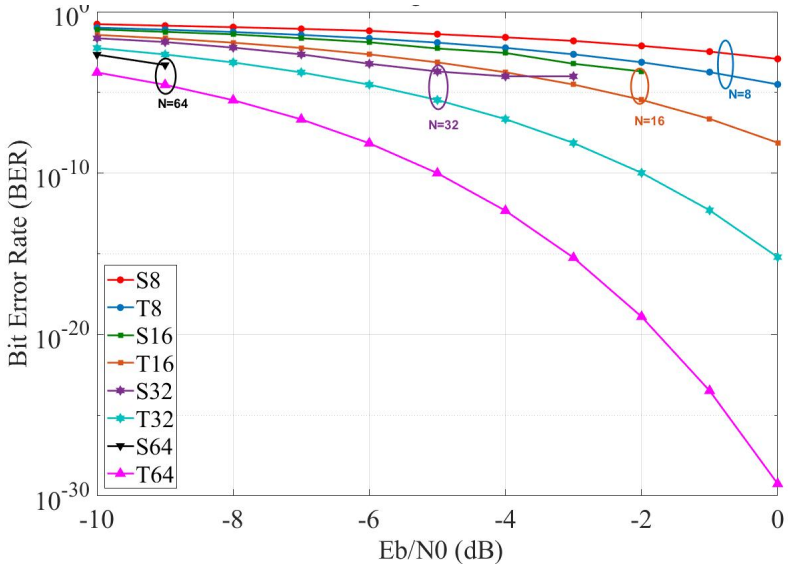


Figure 6.9: BER vs. Eb/N0 for modified CDMA with different code length

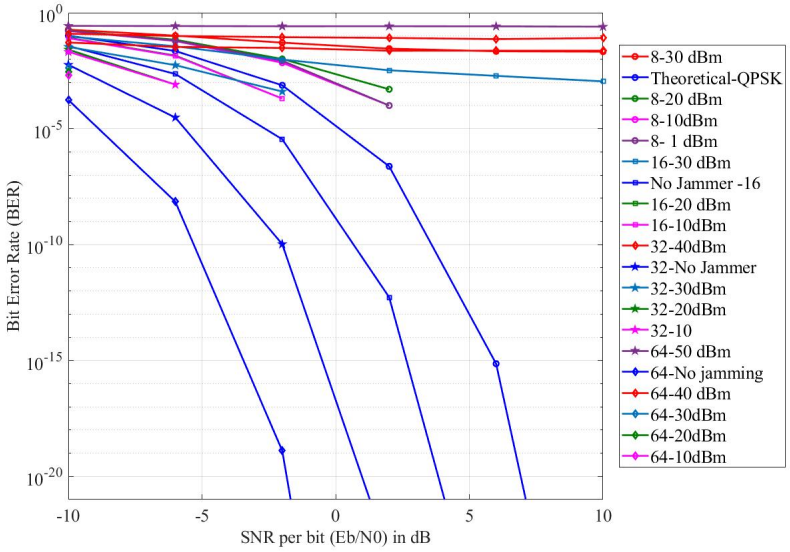


Figure 6.10: BER vs. Eb/N0 for modified CDMA with different code length under Jamming and No Jamming

For example, for a MCDMA with 32 code length and $10dBm$, the BER is $10^{-3}$, which can be considered high and can cause a denial of service for the application that requires BER of $10^{-7}$

## 6.11 Summary

This chapter revisited the direct-sequence spectrum sensing technique and provided an enhanced CDMA technique to provide extra protection against jamming attacks. The proposed technique can correct more errors compared with traditional CDMA. The results indicated that the proposed technique could drain the battery of the jammer as it requires more power to cause a denial of service.

# Chapter 7

# Sensing Based on Machine Learning and Compressive Sensing

Cognitive radio is one of the technologies that can assist spectrum sharing, which is essential to overcome the scarcity of the spectrum in sub-6GHz bands. However, the reliability of spectrum sensing is still an issue in cognitive radio. Early studies on spectrum sensing focused on energy detection, auto-correlation, and matched filtering, but these techniques present many challenges. In addition, wideband spectrum sensing is challenging because of the considerable sensing time and the hardware cost associated with this process. In this chapter, we proposed two algorithms to overcome the issue of spectrum sensing in CR systems. The first algorithm is based on machine learning for narrow-band spectrum sensing, and the second algorithm is based on compressive sensing theory.

## 7.1 System Model

### 7.1.1 Narrow-band Spectrum Sensing

Narrow-band spectrum sensing can be modeled using hypothesis testing. The first hypothesis corresponds to the absence of the primary user signal, denoted $\mathcal{H}_0$, while the second one corresponds to the presence of the primary user signal, denoted $\mathcal{H}_1$. Let $x(t)$ denote the

primary user signal and let $y(t)$ denote the received signal at the receiver of the secondary user. Thus, spectrum sensing can be expressed as:

$$y(t) = x(t)h(t) + w(t) \tag{7.1}$$

where $w(t)$ denotes additive white Gaussian noise. In the frequency domain, the received signal can be estimated by computing DFT, which is given by,

$$R = \frac{1}{\sqrt{N}} y(t) e^{-j2\pi t/N} \tag{7.2}$$

We compute the energy statistic as the sum of the received energy over S samples,

$$E = \sum_{s=0}^{S} |R(s)|^2 \tag{7.3}$$

and the decision rule is given by

$$decision = \begin{cases} \mathcal{H}_0 : & E \geq \gamma \\ \mathcal{H}_1 : & \text{otherwise.} \end{cases} \tag{7.4}$$

where $\gamma$ correspond to a threshold

**Machine learning-Based Spectrum Sensing**

It is challenging to set a threshold in a time-varying wireless environment, so we have investigated machine learning theory. To train machine learning techniques such as decision tree, random forest, SVM, and neural network, we have generated a dataset of 5000 training samples. To generate this dataset, we have considered primary users under two scenarios:

active and inactive, and each time we recorded the received signal at the SUs. In both scenarios, the SU are randomly distributed in a geographic area of $1km^2$. The SUs have different channels and different signal-to-noise-ratio. The energy statistic is then computed for each received signal and labeled as presence or absence of the primary user according to the primary user's activity. Then, the spectrum sensing is modeled as a classification problem where machine learning techniques have to classify the energy statistic into one of two classes: $\mathcal{H}_0$ or $\mathcal{H}_1$. Probability of detection, probability of miss-detection, probability of false alarm, and accuracy are considered to evaluate the performance of these machine learning techniques.

### 7.1.2 Wideband Spectrum Sensing

In this part, we consider multi-carrier communication over a wideband channel. This wideband channel is divided into k narrowband channels. The spectrum sensing decision on the $k^{th}$ narrowband channel as choosing between $\mathcal{H}_{o,k}$, which represent that the channel is not occupied by the PU, and $\mathcal{H}_{1,k}$, which represents that channel is occupied by the PU. Therefore, the task of a SU spectrum sensor is to sense these K channels and determine the activity of the PU.

The wideband signal

$$y(t) = \sum_{k=1}^{k=K-1} x_k(t) * h_(t) + w(t) \tag{7.5}$$

where $x_k(t)$ denotes the primary user at the $k^{th}$ narrowband at the time t, and $w(t)$ denotes an additive white Gaussian noise with zero mean and variance $\sigma_w^2$. In the frequency domain, the received signal at each subchannel can be estimated as

$$Y_k = \frac{1}{\sqrt{N}} \sum_{t=0} y(t) e^{-j2pitk/N} \tag{7.6}$$

$$Y_k = H_k S_k + W_k, k = 0, 1, ..., K-1 \tag{7.7}$$

where $s_k$ is the primary transmitted signal at subchannel k and

$$W_k = \frac{1}{\sqrt{N}} \sum_{t=0} w(t) e^{-j2pitk/N} \tag{7.8}$$

and

$$H_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{L-1} h(n) e^{-j2pitk/N} \tag{7.9}$$

Here the spectrum sensing function has to decide whether the $k^{th}$ subchannel is occupied or not.

$$SD = \begin{cases} \mathcal{H}_{0,k} : Y_k = W_k & \text{k } = 1, 2, ..., K-1 \\ \mathcal{H}_{1,k} : Y_k = H_k S_k + W_k & \text{sinon.} \end{cases} \tag{7.10}$$

where $\mathcal{H}_{0,k}$ and $\mathcal{H}_{1,k}$ indicate the absence and the presence of the primary user, respectively.

**Compressive Wideband Spectrum Sensing**

$Y_k$ is assumed to be sparse as only sub-carrier frequencies are occupied. This assumption has been validated by many measurements. Given that $Y_k$ is sparse, the wideband signal can be reconstructed from a few measurements. If the $Y_k$ is not sparse, then multiplication by a sparsifying matrix can transform the wideband signal into a sparse signal. The theory of compressive sensing, as we showed in Chapter 4 provides an excellent framework to recover the wideband signal from a few measurements.

The obtained measurements can be modeled by multiplying the wideband signal $Y_k$ by a measurement matrix $\phi_f$. Therefore, we can express the vector of measurement as

$$y_f = Y_k \phi_f \tag{7.11}$$

where $y_f$ denotes the measurement vector, $Y_k$ denotes the wideband signal, and $\phi$ denotes the measurement matrix.

Then, the recovery of the wideband signal consists of finding the sparse vector with the smallest $l-1$ norm that satisfies the measurement equation. We write

$$min_z ||z||_1 s.t. y_f = Y_k \phi_f \tag{7.12}$$

This recovery problem can be solved using Bayesian compressive sensing techniques or greedy methods. Based on the results we have shown in Chapter 4, we have solved the wideband signal reconstruction problem using Bayesian compressive sensing.

## 7.2 Illustrative Results and Discussions

### 7.2.1 Examples of Results of Narrowband Sensing

Examples of results are shown in Figs. 7.1 through 7.4. Fig. 7.1 shows the accuracy of Random Forest as a function of the number of folds. This figure indicates that the accuracy is an increasing function of the number of folds. For a number of folds less than 10, random forest achieves an accuracy of 99.2%. For a number of folds between 10 and 50, the accuracy is 99.55% and 99.6%. For a number of folds equal to 100, the accuracy is around 99.65%.

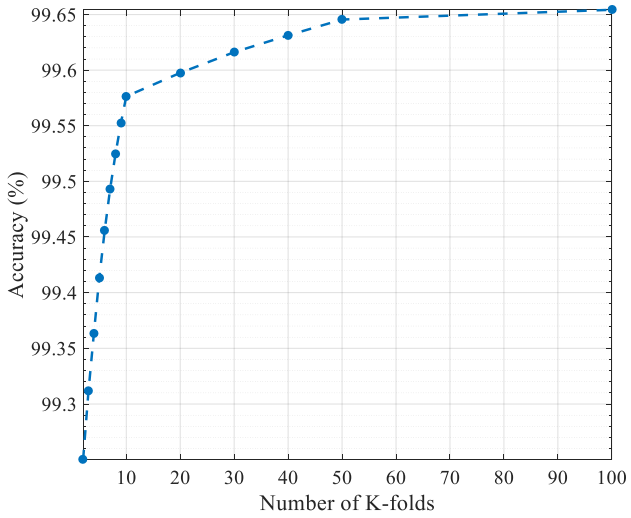


Figure 7.1: Accuracy versus the number of folds for random forest for a number of estimators equal 100.

Fig. 7.2 shows the accuracy of random forest for both training and testing as a function of the training size. The model achieves an accuracy of 100% on training while achieving an accuracy higher than 99%. This indicates that random forest does not overfit the training set.

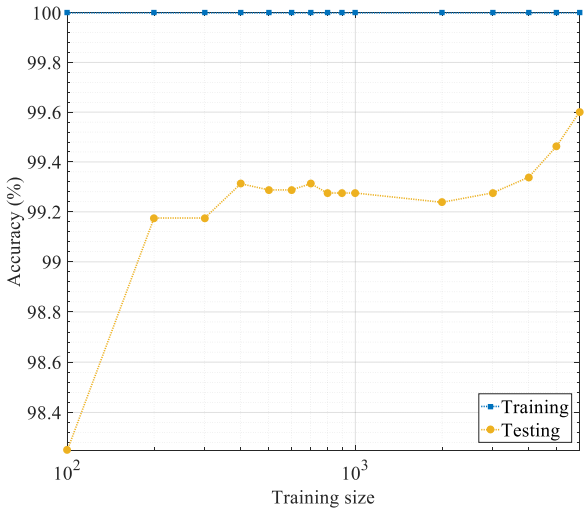


Figure 7.2: Accuracy versus number of estimators of random forest

Fig. 7.3 shows the performance of random forest in terms of the classification accuracy in

Table 7.1: Performance comparison of support vector machine-based sensing techniques with different kernels

| Models | $P_d$ | $P_m$ | $P_{fa}$ | Accuracy |
|---|---|---|---|---|
| Linear SVM | 99.09 | 0.909 | 0.91 | 98.9 |
| Quadratic SVM | 4.05 | 95.9 | 95.95 | 51.8 |
| Cubic SVM | 82.5 | 17.53 | 17.5 | 91.5 |
| RBF SVM | 99.1 | 0.91 | 0.90 | 98.8 |

terms of the number of estimators used. This figure indicates that as the number of estimators increases, the accuracy of random forest increases as well. However, we can see that the performance of random forest does not change a lot for a number of estimators between 10 and 100. For the rest of the results, we consider random forest with 100 estimators.

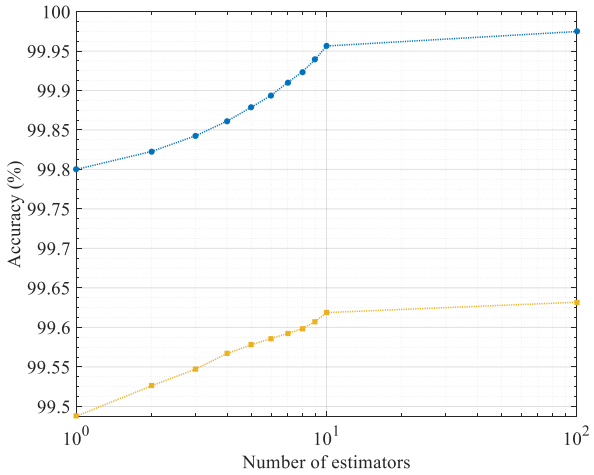


Figure 7.3: Accuracy of the random forest for both training and testing with 100 estimators.

Fig. 7.4 shows the probability detection versus the probability of false alarm of random forest and several other algorithms such as decision tree, support vector machine, logistic regression, naive Bayes, k-nearest neighbors, and neural networks. This figure shows that random forest outperforms all other techniques.

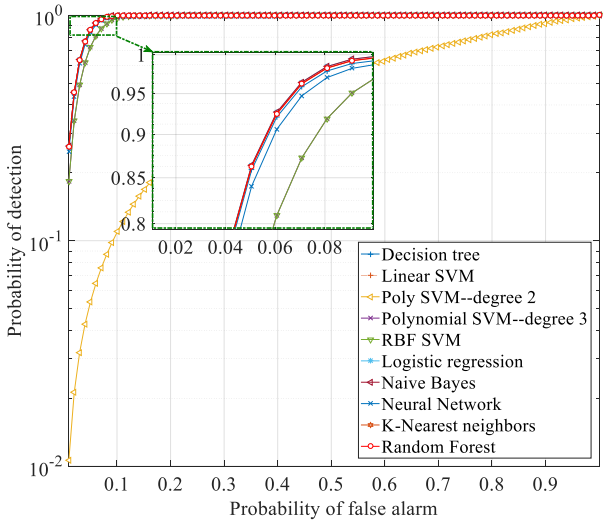


Figure 7.4: Probability of detection versus the probability of false alarm for machine learning-based spectrum sensing techniques

Table 7.2: Performance comparison between different machine learning-based sensing techniques

| Models | $P_d$ | $P_m$ | $P_{fa}$ | Accuracy |
|---|---|---|---|---|
| Linear SVM | 99.09 | 0.909 | 0.91 | 98.9 |
| Logistic Regression | 98.98 | 1.01 | 1.02 | 98.8 |
| Random Forest | 99.37 | 0.62 | 0.63 | 99.65 |
| KNN | 98.88 | 1.12 | 1.12 | 99.3 |
| Naïve Bayes | 98.6 | 1.43 | 1.4 | 99.27 |

### 7.2.2 Examples of Results of Wideband Sensing

Examples of results are shown in Figs. 7.9 through 7.12. Fig 7.9 presents a comparison between the probability of detection versus SNR for CS-based sensing using different numbers of measurements and the probability of detection versus SNR of the Nyquist-based sensing technique. The SNR values range from $-20dB$ to $+20dB$. This figure also shows that the probability of detection increases as SNR increases for both approaches, compressive and Nyquist-based sensing, to reach 100% for $SNR = 0dB$. It can also be observed that the probability of detection with CS-based sensing reaches 100% for lower SNR values as the

number of measurements increases. For a number of measurements $M = 128$, the probability of detection reaches 100% for $SNR = 15dB$ and for $M = 1024$, the probability of detection reaches 100% for $SNR = 0dB$. If we reduce the number of measurements by 50%, the probability of detection decreases by 10% for $SNR = -5dB$ and by 5% for $SNR = -2dB$, but remains constant for SNR higher than 0 dB. This means that if we are able to estimate the value of SNR, we can set the number of measurements to a specific value and still obtain the same value of the probability of detection. For instance, for $SNR = 3dB$, the number of measurements can be reduced to $M = 512$, and the detection performance remains the same as the one of the Nyquist-based sensing.

Fig. 7.10 shows the probability of detection versus the probability of false alarm for standard sensing and compressive sensing-based techniques using different numbers of measurements. One can see that the probability of detection increases as the probability of false alarm increases to reach 100% for $P_{fa} = 30\%$ for normal sensing and $P_{fa} = 50\%$ for compressive sensing based sensing for a compression ratio $M/N = 50$. The same figure shows that the probability of detection increases as the number of measurements increases. The lowest value of the probability of detection corresponds to a number of measurements M=128, and the highest one corresponds to the number of measurements $M = 1024$.

Fig. 7.11 shows the MSE of the proposed approach versus the number of measurements for different values of SNR. As expected, the mean square error decreases as the number of measurements increase for all values of SNR. The MSE corresponding to low SNR values is higher, and those corresponding to high SNR values are smaller. Also, the MSE increases as the SNR decreases. Hence, if we can estimate the value of SNR, we can choose the number of measurements accurately and then improve the detection performance of the compressive spectrum sensing technique.

Fig. 7.12 shows the sensing time needed to scan 20 channels of the band 2.4 GHz, versus

the number of measurements. As expected, the sensing time decreases as the number of measurements decreases. For instance, for a number of measurements M=2048, the sensing time is 54 ms, and for M=1024, the sensing time is 37 ms, which means that if we decrease the number of measurements by 50%, the sensing time is also decreased by almost 32%.

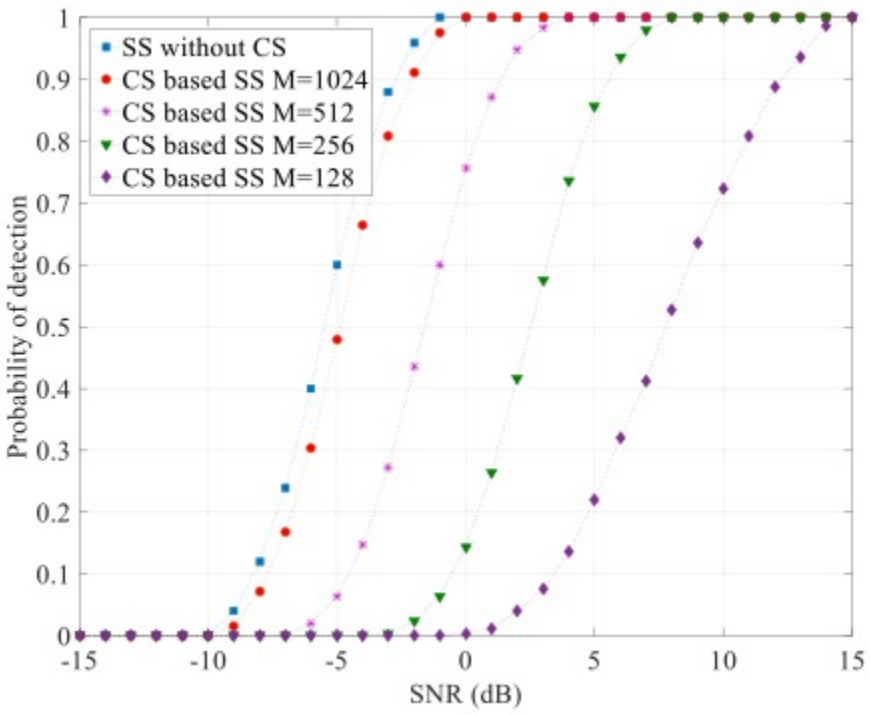


Figure 7.5: Probability of detection versus SNR for two approaches, normal sensing and CS based sensing, of the channel 2.437 GHz for a probability of false alarm $Pfa = 0.1$, number of samples $N = 2048$, and four values of the number of measurements, $M = 128, 256, 512, and 1024$

Table 7.3: Performance Comparison of sensing techniques for $SNR = 0dB$ $P_{fa} = 10\%$

| Metrics | NS | WCS 50% | WCS 25% | WCS 12.5% | WCS 6.25% |
|---|---|---|---|---|---|
| Pd | 100% | 100% | 75% | 15% | 0% |
| Processing time in (ms) | 54 | 37 | 33.6 | 30.6 | 24.33 |
| MSE for SNR = 0 dB | - | 0.09 | 0.88 | 1.15 | 1.46 |

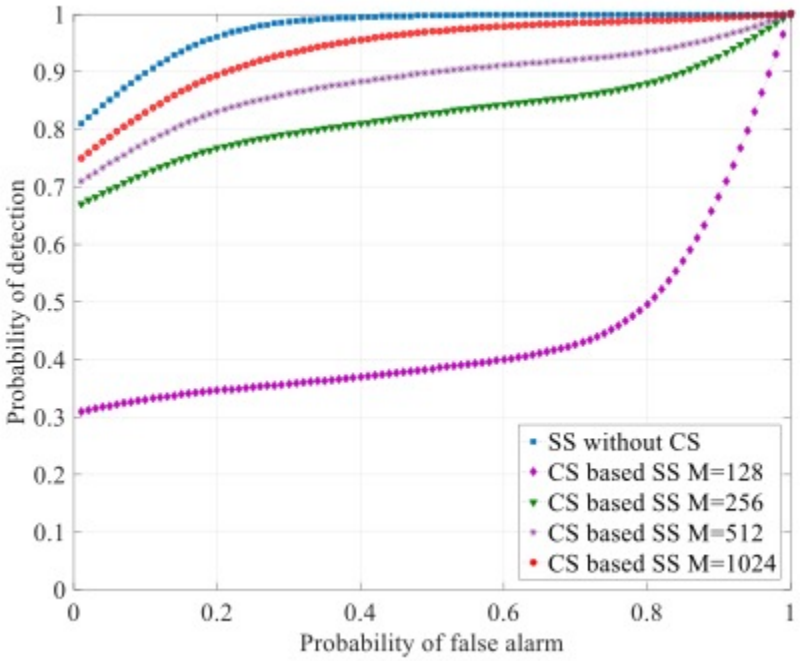


Figure 7.6: Probability of detection versus the probability of false alarm for the two approaches, normal sensing and CS based sensing, of the channel $2.437GHz$. In this experiment, $SNR = -5dB$, number of samples $N = 2048$, and number of measurements, $M = 128, 256, 512, and 1024$.

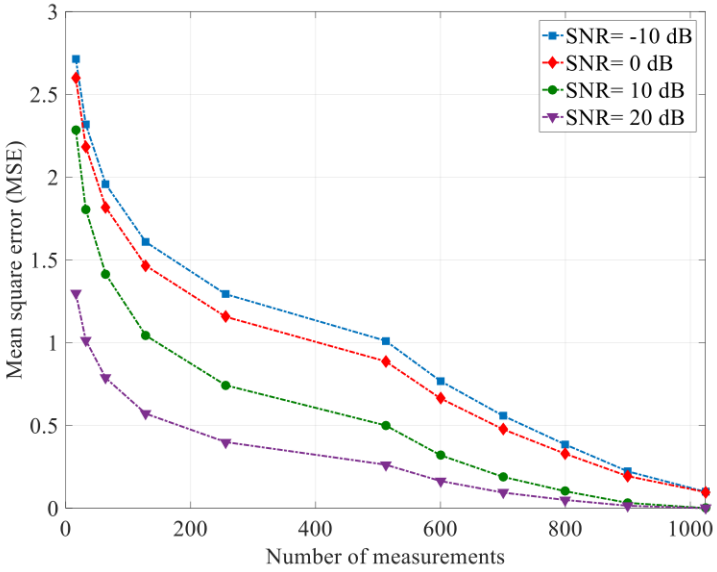


Figure 7.7: Mean square error of the proposed approach versus the number of measurements for the SNR values $-10, 0, 10, and 20dB$.

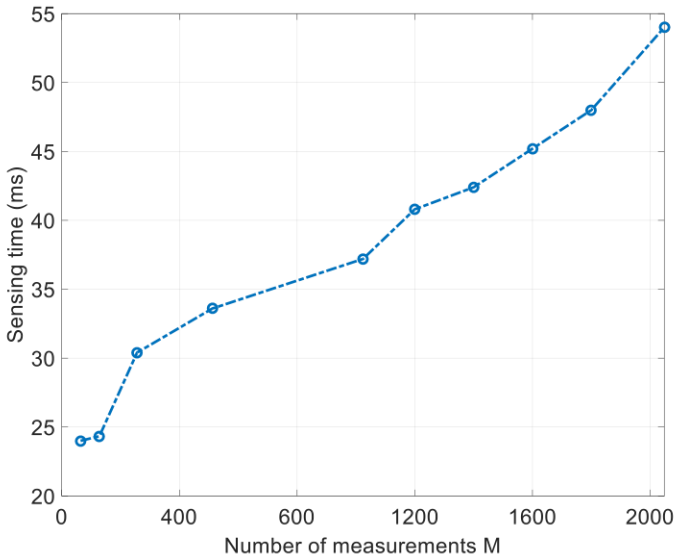


Figure 7.8: Sensing time for 20 channels of the band 2.437$GHz$ versus the number of measurements.

## 7.3 Deep Learnig-Based Spectrum Sensing Data Falsification

The sensing function is vulnerable to cyber-attacks, including spectrum sensing data falsification where the malicious users attempt to cause secondary users to access occupied channels to cause intentional interference to the primary user communication.

### 7.3.1 System Model

Consider a set of N users performing spectrum sensing collaboratively to detect the availability of a frequency channel. These users report their individual sensing decisions to a centralized unit such as the fusion center. The fusion center combines these individual decisions to make a final decision before broadcasting the final decision to the secondary users. Spectrum sensing data falsification aims at injecting false samples to mislead the secondary

users.

At the end of the sensing decision, each user has to decide,

$$SD = \begin{cases} \mathcal{H}_{0,k} : Y_k = W_k & \text{k } = 1, 2, ..., K-1 \\ \mathcal{H}_{1,k} : Y_k = H_k S_k + W_k & \text{otherwise.} \end{cases} \tag{7.13}$$

At the end of the individual sensing, the individual sensing data of each SU is determined. $d_k$ indicates the individual sensing data of $SU_k$, which is usually expressed as a binary variable:

$$d = \begin{cases} \mathcal{H}_{0,k} : d_i = 0 & \text{k } = 1, 2, ..., K-1 \\ \mathcal{H}_{1,k} : d_i = 1 & \text{otherwise.} \end{cases} \tag{7.14}$$

The final decision (d) of FC is also binary under the "logic AND", "logic OR" and "Majority" rule. For example, in the "logic AND" rule, if all individual sensing $d_k = 1$, then FC makes $d = 1$. The "logic OR" rule refers to $d = 1$ if one $d_i = 1$. The "Majority" rule refers to d = 1 from the $k$ out of $N$ rule under the case when $k \geq \frac{N}{2}$.

### 7.3.2 Deep Neural Network-Based SSDF Detection

Since the SU report their final decision to the fusion center, attackers can easily manipulate the final decision $d_i$ and alter the final decision. Based on this, we considered two types of SSDF attacks:

1. SSDF type 1: The user always reports a busy channel.

2. SSDF type 2: The user always reports a free channel.

To detect SSDF, features such as the RSSI and signal-to-noise ratio are considered.

### 7.3.3 Examples of Results and Discussions

To determine the number of layers, the number of hidden neurons in each layer, and the number of cross-validation folds, we consider a scenario in which the optimizer is fixed on a large number of iterations and a minimal learning rate that the algorithm converges. Then, for a different number of hidden neurons and hidden layers, we compute the accuracy of the neural network. An example of results is given in Fig. 7.9. This figure shows the accuracy of the neural network versus the number of hidden layers and the number of hidden neurons in each layer for different values of cross-validation folds. As it can be seen from this figure, the structure of the neural network that has the highest accuracy is for two hidden layers of 20 neurons, each for a number of cross-validation folds equal to 20. With this structure, the accuracy is 98.50%.

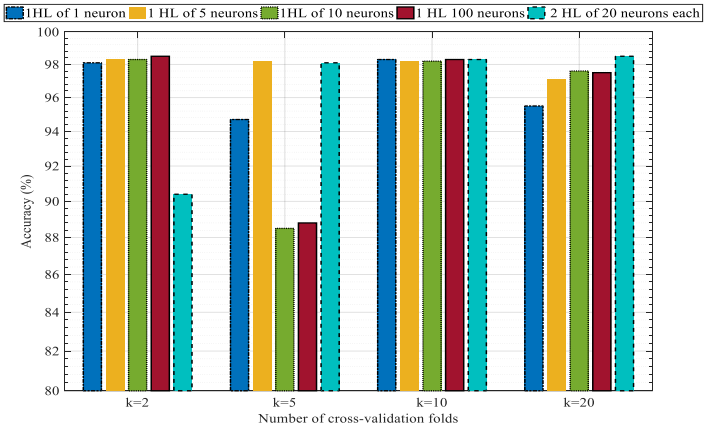


Figure 7.9: Probability of detection versus SNR for two approaches, normal sensing and CS based sensing, of the channel 2.437 GHz for a probability of false alarm $Pfa = 0.1$, number of samples $N = 2048$, and four values of the number of measurements, $M = 128, 256, 512, and 1024$

Fig. 7.10 shows the probability of detection as a function of the probability of false alarm for a neural network with a learning rate of 0.001 and a number of iterations equal to 400. Cross-Validation is considered in these experiments to compute the probability of detection.

A total number of 20 folds is considered to calculate the probability of detection as a function of the probability of false alarm. From this figure, it can be seen that the proposed model has an area under the curve on average equal to 99%. The ROC curve of the structure with two hidden layers of 20 neurons each is above the curves corresponding to one hidden layer with 29 neurons and 100 neurons. Fig. 7.11 shows the accuracy of a neural network with

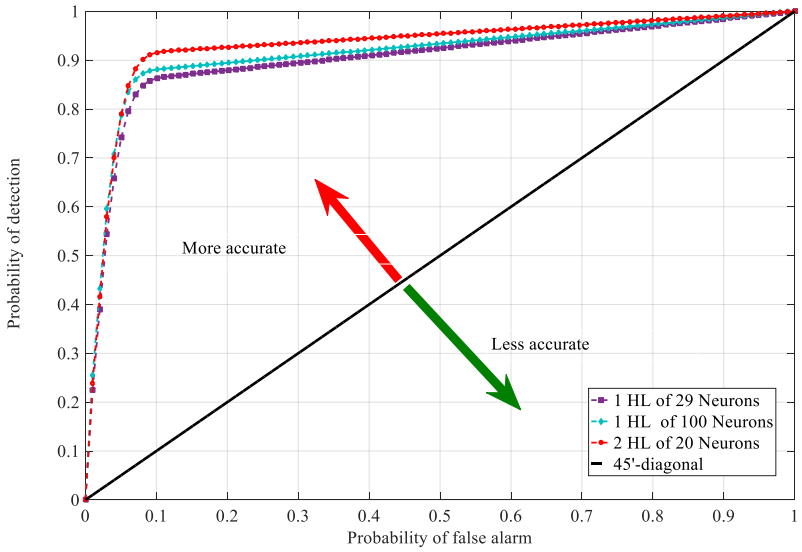


Figure 7.10: Probability of detection versus the probability of false alarm for the two approaches, normal sensing and CS based sensing, of the channel $2.437GHz$. In this experiment, $SNR = -5dB$, number of samples $N = 2048$, and number of measurements, $M = 128, 256, 512, and 1024$.

two hidden layers of 20 hidden neurons each as a function of the number of iterations. The activation function is a "relu" function for the input layer, a hyperbolic tangent function for the hidden layer, and a sigmoid function for the output layer. For a number of iterations less than 400, one can see that the accuracy increases as the number of iterations increases for both training and testing. However, for a number of iterations higher than 400, the accuracy for both the training and testing remains constant and is independent of the number of iterations. Fig. 7.12 shows the cross-entropy cost function versus the number of iterations for the same structure of the neural network. From this figure, it can be observed that,

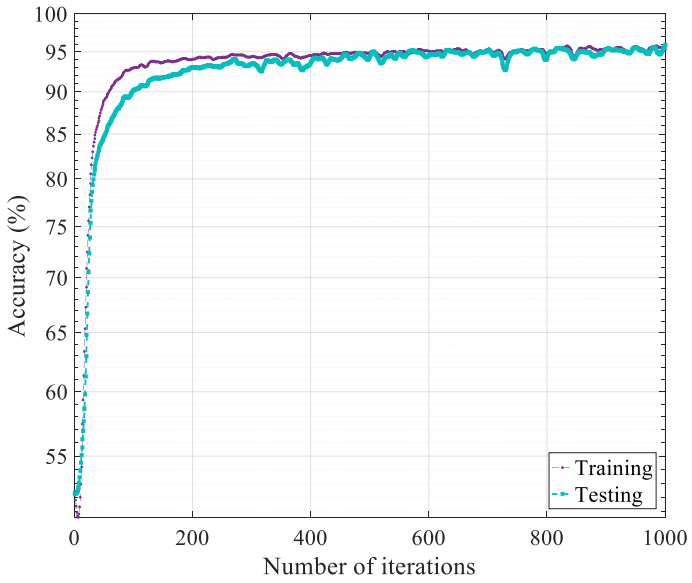


Figure 7.11: Mean square error of the proposed approach versus the number of measurements for the SNR values $-10, 0, 10, and 20dB$.

for both training and testing, the cross-entropy cost function decreases as the number of iterations increases to reach a value less than 2% of error after 200 iterations.

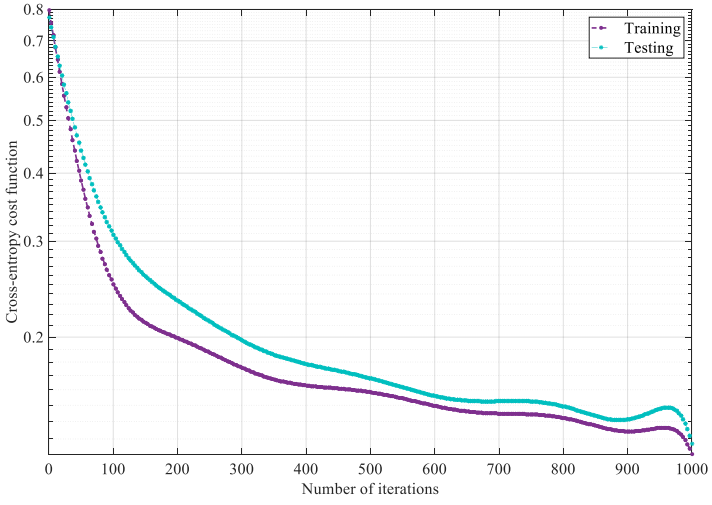


Figure 7.12: Sensing time for 20 channels of the band $2.437GHz$ versus the number of measurements.

Table 7.4 compares the proposed model and the support vector machine with different kernels

and logistic regression. The comparison is performed considering four different evaluation metrics, the probability of detection, the probability of miss-detection, the probability of false alarm, and the accuracy. From this table, one can see that SVM with a linear, RBF SVM, and logistic regression can achieve a high probability of detection of 100%. In contrast, the proposed model achieves a probability of detection of 90.2%, which is less than SVM with logistic regression but higher than polynomial SVM with degree 3, which has a probability of detection of 84.27%. In terms of probability of miss-detection, Linear SVM, RBF SVM, and logistic regression have a probability of miss-detection of 0%. In contrast, the proposed model has a probability of miss-detection of 9.8%, and polynomial SVM has a probability of miss-detection of 15.7%. However, in terms of the probability of false alarm, the proposed model has the lowest probability of detection, about 8%, followed by linear SVM with 39%, logistic regression with 39.24%, RBF SVM with 49.33%, and then polynomial SVM with 49.51%. The proposed model has the highest accuracy with 98.5%, followed by linear SVM with 81.2%, logistic regression with 80.68%, RBF SVM with 78.68%, and finally polynomial SVM with 69.12%.

Table 7.4: Performance comparison of DNN and machine learning

| **Classifier** | **Pd** | **Pm** | **Pfa** | **Accuracy** |
|---|---|---|---|---|
| DNN | 90.2% | 9.8% | 8% | 98.5% |
| Linear SVM | 100.0% | 0% | 39% | 81.2% |
| Polynomial SVM, degree 3 | 84.27% | 15.7% | 46.51% | 69.12% |
| RBF SVM | 100.0% | 0% | 43.33% | 78.68% |
| Logistic regression | 100% | 0% | 39.24% | 80.68% |

## 7.4 Summary

The main contributions of this chapter are threefold. First, we developed machine learning-based narrow-spectrum sensing techniques to overcome the limitations of traditional spec-

trum sensing, such as energy detection. We showed that machine learning with the proper tuning could achieve a high probability of detection. Second, we also developed techniques for wideband spectrum sensing based on Bayesian compressive sensing methods. We demonstrated that from a few measurements, the availability of the primary users could be determined. This solution reduces the sensing time and the complexity of wideband spectrum sensing techniques. Third, we developed a deep learning framework to detect spectrum sensing data falsification to enhance the robustness of cooperative spectrum sensing.

# Chapter 8

# Conclusions and Future Research Directions

In this dissertation, we investigated machine learning for designing mmWave communications to enable future wireless communication. mmWave communications exploit high frequencies to provide higher bandwidth, given that Shannon capacity has been touched and the only resort to increase the spectral efficiency is to increase the bandwidth. Te transmission over mmWave bands compared with sub-6 GHz is subject to extra path loss because of the small wavelength. This high path loss is exacerbated by material absorption and scattering. Massive MIMO techniques have been proposed as enabling technology for 5G and beyond that compensate for this extreme path loss. Massive MIMO also enables robust communications against intentional jamming, which is one of the vulnerabilities of wireless communication because of the open communication medium. The use of massive MIMO with fully digital beamforming is expensive and consumes high power. Therefore, massive MIMO with hybrid beamforming has attracted many researchers as a cost-effective and power-efficient while achieving close performance to fully digital beamforming architecture. However, the use of hybrid beamforming entails high signal processing complexity as the phase shifter networks make finding the precoding/combining matrices a non-convex problem. In addition, the receiver has only access to a few training pilots because of the analog combiner, making channel estimation in this architecture more challenging, especially since the signal-to-noise

ratio during beamforming training is very low (-20 dB to 0 dB). The goal of this dissertation was to design low-cost and model-free techniques for enabling mmWave communications.

## 8.1 Datasets

Table 8.1 provides a summary of the datasets used in this dissertation. These datasets are Raymobtime, Jamm5GNR, SenseML, DeepSSDF, and DeepMIMO. The first dataset Raymobetime is built using the Ray-Tracing technology to provide a large dataset to be used for channel estimation in hybrid beamforming. The parameters of this dataset are summarized in the Table. The second dataset Jam5GNR is a small dataset that was used to train traditional machine learning techniques. It consists of 627 instances, and it was generated using the 5G New Radio Matlab Toolbox. The new waveform and the numerology of 5G New Radio are considered. The jamming power changes between 30 dBm and 70 dBm. The third and fourth datasets consisted of 5000 and 1000 instances and were generated to deal with spectrum sensing in cognitive radio networks. The last dataset is DeepMIMO which has been made public to help advance research in hybrid beamforming.

Using These datasets, we could achieve adequate performance. For instance, we could achieve a 90% probability of detection for jamming and SSDF. We also achieved NMSE of $-20dB$ using Raymobetime for channel estimation in hybrid beamforming. We finally achieved high accuracy in spectrum sensing using SenseML.

## 8.2 Summary and Conclusions

In chapter 2, we reviewed the state-of-the-art and highlighted the gap of knowledge that this research is attempting to address. We showed an urgent need to incorporate machine

Table 8.1: Summary of dataset

| Dataset | Domain | size | Technology | Task | Class size | SNR | Number of Users/ base Station | System parameters | % split |
|---|---|---|---|---|---|---|---|---|---|
| **Raymobtime** | Channel Estimation in Hybrid Beamforming | 10,000 channels | Ray-Tracing/ WirelessInsite/ OpenStreepMap | Regression | NA | SNR=-15dB, -10 dB, and -5 dB | NA | Nt = 16; Number of TX antennas<br>Nr = 64; Number of RX antennas<br>Nbits= 4; Number of bits.<br>Lt = 2; Number of TX RF chains<br>Lr = 4; Number of RX RF chains<br>Ns = 2; Number of data streams<br>Nfft=256<br>Pt=1 Transmit power(mw) | 70 % (training)<br>20%(validation)<br>10% (Testing) |
| **Jam5GNR** | Jamming Detection | 627 | Matlab/ 5G New Radio Toolbox | Classification | 50%/50% | –10 dB to 10 dB | NA | 30 dBm to 70 dBm (jamming power) Pulse with 100% | 70 % (training) 30 % (Testing) |
| **SenseML** | Narrow-band Sensing | 5000 | Matlab | Classification | 50%/50% | -20dB to 20 dB | 1000 | Users randomly distributed in 1km^2 area | 70 % (training)<br>20%(validation)<br>10% (Testing) |
| **SSDF** | Spectrum Sensing Data Falsification in Cognitive Radio | 1000 | Matlab | Classification | 50%/50% | -20dB to 20 dB | 1000 | Users randomly distributed in 1km^2 area | 70 % (training)<br>20%(validation)<br>10% (Testing) |
| **DeepMIMO** | Hybrid Beamforming | UP to 10000 | Ray-Tracing/ WirelessInsite/ OpenStreepMap | Regression | NA | | Value Active BSs 3, 4, 5, 6<br>Active users From row R1000 to row R1300 | Number of BS Antennas Mx = 1, My = 32, Mz = 8 Antenna spacing d=0.5 System bandwidth B=0.5 GHz Number of OFDM subcarriers 1024 OFDM sampling factor 1 OFDM limit 64 Number of paths 5 | 70 % (training)<br>20%(validation)<br>10% (Testing) |

learning to transform mmWave massive MIMO communication from model-based to model-free communication. Machine learning tools such as reinforcement learning are suitable for achieving model-free wireless communication. Deep reinforcement learning could reduce the communication overheads and improve future wireless communication security and robustness. In order to advance this field, research on issues related to deep learning and reinforcement learning in wireless communication is essential. For instance, to train deep learning, there is a need to build a large dataset, so how to build a realistic dataset and how to select the hyperparameter and some other practical methodologies such as the choice of the performance metrics, the complexity, the memory, and the speed.

In chapter 3, we described the mathematical model of hybrid beamforming. We formulated the problem of finding the hybrid precoding and combining matrices in hybrid architecture as a Markov decision process and solved it using the theory of deep reinforcement learning. We investigated several techniques such as double deep Q-learning, soft actor-critic, and Twin-delayed deep deterministic policy gradient. The results indicated that deep reinforcement learning could learn the optimal policy to find the hybrid beamforming matrices and achieve near-optimal performance. We showed that machine learning could be leveraged in designing hybrid beamforming massive MIMO techniques to achieve adequate performance with low cost and minimum energy consumption. The advantages of using the previous deep reinforcement learning techniques-based hybrid beamforming techniques such as soft actor-critic methods and twin-delayed deep deterministic policy gradient method improved the performance of deep learning-based hybrid beamforming. They addressed the issues associated with Q-learning methods, such as the overestimation bias.

In chapter 4, we provided compressive sensing for mmWave communications. We showed that a compressive sensing framework could be incorporated to estimate the mmWave channel in the context of hybrid beamforming architecture where the receiver has only a few

radio frequency chains. The sparsity of the mmWAVE channel enables the recovery of the mmWave channel from a few measurements. We formulated the sparse channel estimation in the mmWave bands using the framework of compressive sensing. We showed that several measurement matrices could be used, including random measurement matrices and deterministic measurement matrices. Next, we showed that the channel could be recovered using several sparse recovery techniques from a few measurements. We compared the performance of six sparse recovery algorithms belonging to three categories: convex and relaxation, greedy, and Bayesian compressive sensing. We demonstrated that greedy techniques outperform the rest in terms of recovery time while providing adequate mean square error. Convex and relaxation techniques outperform the rest of the algorithms in terms of recovery error. Bayesian techniques provide a tradeoff between minor recovery errors and short recovery time. We also compared several sensing matrices' performances and showed that Toeplitz and Circulant matrices could perform close to random measurement matrices. This result is significant and can help fast design beamformers that can assist in estimating the mmWave channel at a low-cost, especially since compressive sensing requires fewer training samples to efficiently sense and recover the mmWave channel from a few measurements. The presented results suggest that the Toeplitz measurement matrix and Fast Laplace Bayesian Recovery can be good candidates for designing compressive mmWave channel estimation solutions.

In chapter 5, we have presented a classification of jamming attacks, and we have demonstrated the vulnerabilities of 5G NR channels and reference signals to jamming attacks. Then, we described the jamming detection system model in massive MIMO systems, and we showed that jamming detection could be formulated as a classification problem, which can be solved using machine learning classifiers such as decision trees, random forest, and gradient boosting techniques to name a few. We have developed Hoeffding trees for real-time jamming detection where the trees are built online instead of being trained offline. To

train these machine learning techniques, we have built a realistic dataset from a simulation setup as collecting real-world signals is expensive and used it to train these machine learning techniques. Examples of the detection performance of these techniques are reported and analyzed, and from which we draw the following conclusions:

1. Decision trees-based models achieve high accuracy given enough training samples. The performance of decision trees depends on the number of trees and the number of samples used for training.

2. XG Grid search and XGB boosting outperform the rest of the tested algorithms, but they require more time to be trained and tested;

3. The detection performance highly depends on the number of samples collected and the noise level under which these samples are collected;

4. Hoeffding decision trees achieve comparable performance to traditional decision trees and can be trained online, which makes them suitable for on-fly detection.

In chapter 6, we have revisited the direct-sequence spread spectrum to provide robust communication against friendly and intentional interference, which can be used for mmWave communications, which has enough bandwidth. We provided a modification of code division multiple access schemes, so it does not require any additional channel coding techniques using the capabilities of orthogonal codes and their complements. The proposed modification makes code division multiple access schemes more bandwidth-friendly while enhancing its correcting capabilities. We demonstrated that the built-in coding schemes correct more than 50% of the errors. The theoretical results have been validated using a simulation setup, in which we have shown that the modified CDMA enhances the correcting capability of CDMA systems. Our results indicated that the modified CDMA provides extra protection against adverse jammer.

## 8.3 Discussion and Future research Directions

There are several practical problems when it comes to the use of deep reinforcement learning. For instance, deep reinforcement learning tuning parameters is a tedious task as there are no guidelines that can help set the parameters that maximize the reward function. One has to tune the parameters and train and test the model to tell if these parameters yield the desired performance. If the obtained performance is not satisfying, one must change the parameters again and keep iterating until finding these parameters. This is very challenging in the realm of deep learning in general and not only in reinforcement learning. This is currently an active area of research in deep learning.

The second challenge in deep reinforcement learning is the choice of the reward function. A good choice of the reward function can save a big deal of time for parameter tuning. The binary reward choice, +1 for successful action and -1 for failed action, often leads the model to be stuck in the exploration phase and not get to the exploitation phase. The model starts with a total reward of zero, and if it takes failed action, which is very likely to happen at the start, push the agent to explore new action, and accumulate negative reward and keep exploring. Some alternative ways for reward functions are the choice of +1 reward for successful actions and small decay negative function for failed actions to encourage the model to exploit and accumulate positive results before the exploration phase.

The substantial difference in problem formulation, the choice of the actions, the states, and the reward of the system make it difficult to compare results from different papers as there is no common baseline. Open AI Gym platform provides a solution for these problems in other fields. RL research can progress faster if such a platform exists for wireless communication systems, including an environment for dynamic spectrum access and massive MIMO.

For the learned lessons, we can state them as follows. Training deep reinforcement learning

can benefit from a good reward function as it can save time spent tuning the model, which often leads to undesired performance because the agent is trained with a wrong reward function. The second learned lesson is that reply experience can enhance the performance of deep reinforcement learning. Deep reinforcement learning can also benefit from replay experience, which can help the model generate several episodes and get trained on sampled episodes only. The sampling of the episodes can save the model from being trained on a massive number of episodes that are not of any help to the agent. The third learned lesson is that soft target updates can help with enhancing the stability of reinforcement learning. Usually, fixing the target for the agent even for a while can help with fast convergence. If the model has a moving target can have problems in reaching the target and often yields divergence in the model. One way to bypass this issue is to fix the target model and update the model after a few iterations, for example, after ten iterations, which can help the model converge faster.

This section revisits the advantages of deep learning and machine learning in building intelligent 5G wireless communication and networking. It also discusses the challenges facing the integration of AI in wireless communication and some future research directions to speed up this integration.

### 8.3.1 Advantages

The use of deep learning to build intelligent 5G systems has many advantages. For instance, deep learning can perform automatic feature extraction in signal processing, which is a challenging task in wireless network engineering that often requires human expertise. Deep learning can perform this task with high accuracy. Another advantage is that deep learning models, in some cases, can achieve high-performance accuracy and outperform traditional

techniques. 5G wireless networks will generate big data at high data rates. Deep learning can enable 5G systems to take advantage of this to engineer optimized wireless networks.

### 8.3.2 Challenges

The integration of AI in 5G wireless communication systems faces many challenges. Some of these challenges can be listed as follows:

#### The reliability and speed trade-off

The reliability of these techniques is far less than traditional techniques in wireless communications in solving some problems. For instance, deep learning can compete with LS and MMSE in wireless channel estimation in massive MIMO, but slow feedback characterizes these techniques. Deep learning inference may elongate the system response time. This is because not most wireless devices have access to cloud computing, and even if it is the case, communication with cloud servers will introduce extra delays.

#### The complexity

Deep learning algorithms, in due course, need to be implemented in wireless devices. However, many wireless devices have limited memory and computing capabilities, which is not suitable for complex algorithms. Collecting large samples and training deep learning models takes considerable time, which is a significant impediment to deploying them on some wireless devices with limited power and storage. Also, some applications require real-time processing, and on-fly sampling and training often cannot be performed easily. In some cases, the higher the number of samples and the more significant the training time are, the higher the accuracy of recognition of the signal and network features is. Acquiring more

samples and training the models for longer times incur slow feedback. Therefore, the deep learning models should be designed to achieve the best accuracy with fewer samples within a short time.

### Data Collection and Cleansing

It is necessary to collect data and build large comprehensive datasets to train AI models. This task is not often easy to acquire because mobile service providers, for instance, cannot release these datasets, which contain confidential information about the users and can risk the violation of the privacy of their consumers. Also, even with transfer learning, which refers to using models trained on the previous dataset, it is necessary to adapt these models for specific networks and scenarios that require re-training the models. All these reasons restrict the development of wireless AI

### Privacy and Security

Preserving the privacy of the users is the primary concern of mobile and service providers. One of the main challenges in wireless AI is enabling the training on a dataset belonging to users without sharing the input data and putting the personal information of users at risk. It is necessary to have a security approach to boost the integration of deep learning in wireless communications.

## 8.4 Open Problems and Possible Future Research Directions

There are several practical problems when it comes to the use of the neural network or deep learning. For instance, tuning neural networks is a crucial practical question. This is currently an active area of research in deep learning.

Further research studies on anti-jamming techniques are highly needed. The cyber-security requirement of 5G NR has to be embedded in the initial design of these networks. In this way, one can ensure a low-cost deployment instead of developing solutions to deal with future failures. For instance, the base station must implement anti-jamming techniques. For example, if the exchange between the base station and user equipment, the base station should provide a spatial retreat, movement, time, and network reconfiguration.

Another future research direction is the use of a deep learning-based intelligent reflecting surface approach that can be used as anti-jamming. A large comprehensive dataset is needed to train deep learning. For that, a real-world setup is needed. Different jammers should be considered to collect data. Data collection should be done under both scenarios, under jamming, and under the normal scenario. The built dataset can be used to train and test deep learning techniques. Then, these techniques can be combined with sensing to detect the strategy of the jammer and actively select the communication channel that is not under jamming attacks. Deep learning models must be trained to recognize a jamming signal from a legitimate user on the fly. Only if that can be performed, then the legitimate user can identify the pattern of the jammer and dedicate its strategy and accordingly define a mitigation hopping.

The following future research direction is developing hybrid beamforming techniques, and

channel estimation methods for Tera Hertz communications. The developed models in this dissertation can be extended into Tera Hertz communications where the channel model changes, and therefore there is a need to study the impact of these changes on the communication in Tera Hertz communication. Deep Learning for Tera Hertz communication can be an exciting extension of this work as machine learning, and Tera Hertz communication defines the vision of the sixth-generation wireless communications.

There exist datasets for mmWave communication, such as DeepMIMO and Raymobetime datasets, that use ray tracing to build realistic datasets to advance the field of machine learning for mmWave communication (Table 8.1). It would be interesting to develop similar datasets for Tera hertz communications in order to train and test machine learning for hybrid beamforming and channel estimation in this context.

The use of machine learning to design model-free wireless communication may create a new landscape of cyber-attacks as adversarial machine learning techniques can disturb machine learning training by injecting false training samples, which can reduce the performance of deep learning in wireless communication. Studying the performance of deep learning and reinforcement learning techniques under adversarial nodes is important for securing future wireless communication systems. The security of deep learning models is another challenge, as neural networks are prune to adversarial attacks. Attackers can affect the training process by injecting fake training datasets; such injection can lower the accuracy of the models and yield wrong design, which may affect the network performance. Research in the security of deep learning or machine learning, in general, remains shallow.

Cameras are everywhere, and taking advantage of that is necessary to develop computer-vision-assisted beamforming and massive MIMO communication for mmWave AND Tera Hertz communications. Deep learning and thermal images can be incorporated to develop novel and low-cost beamforming techniques and tracking algorithms to address the high

mobility in future generation wireless communications. Out-of-band features can assist in reducing the communication overhead.

# Bibliography

[1] T. S. Rappaport, J. N. Murdock, and F. Gutierrez, "State of the art in 60-GHz integrated circuits and systems for wireless communications," *Proceedings of the IEEE*, vol. 99, no. 8, pp. 1390–1436, 2011.

[2] Y. Azar, G. N. Wong, K. Wang, R. Mayzus, J. K. Schulz, H. Zhao, F. Gutierrez, D. Hwang, and T. S. Rappaport, "28 GHz propagation measurements for outdoor cellular communications using steerable beam antennas in new york city," in *2013 IEEE international conference on communications (ICC)*, pp. 5143–5147, IEEE, 2013.

[3] M. Jacob, S. Priebe, T. Kürner, M. Peter, M. Wisotzki, R. Felbecker, and W. Keusgen, "Fundamental analyses of 60 GHz human blockage," in *2013 7th European Conference on Antennas and Propagation (EuCAP)*, pp. 117–121, 2013.

[4] M. K. Samimi and T. S. Rappaport, "Statistical channel model with multi-frequency and arbitrary antenna beamwidth for millimeter-wave outdoor communications," in *2015 IEEE Globecom Workshops (GC Wkshps)*, pp. 1–7, 2015.

[5] M. K. Samimi and T. S. Rappaport, "Ultra-wideband statistical channel model for non line of sight millimeter-wave urban channels," in *2014 IEEE Global Communications Conference*, pp. 3483–3489, 2014.

[6] N. Gonzalez-Prelcic, A. Ali, V. Va, and R. W. Heath, "Millimeter-wave communication with out-of-band information," *IEEE Communications Magazine*, vol. 55, no. 12, pp. 140–146, 2017.

[7] A. Klautau, P. Batista, N. González-Prelcic, Y. Wang, and R. W. Heath, "5g mimo

data for machine learning: Application to beam-selection using deep learning," in *2018 Information Theory and Applications Workshop (ITA)*, pp. 1–9, IEEE, 2018.

[8] N. Docomo, "Docomo 5G white paper," *White Paper,(Jul. 2014)*, 2014.

[9] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. K. Soong, and J. C. Zhang, "What will 5G be?," *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[10] I. Ahmed, H. Khammari, A. Shahid, A. Musa, K. S. Kim, E. De Poorter, and I. Moerman, "A survey on hybrid beamforming techniques in 5G: Architecture and system model perspectives," *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3060–3097, 2018.

[11] O. E. Ayach, S. Rajagopal, S. Abu-Surra, Z. Pi, and R. W. Heath, "Spatially sparse precoding in millimeter wave MIMO systems," *IEEE Transactions on Wireless Communications*, vol. 13, no. 3, pp. 1499–1513, 2014.

[12] X. Yu, J.-C. Shen, J. Zhang, and K. B. Letaief, "Alternating minimization algorithms for hybrid precoding in millimeter wave MIMO systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 3, pp. 485–500, 2016.

[13] T. Lin, J. Cong, Y. Zhu, J. Zhang, and K. B. Letaief, "Hybrid beamforming for millimeter wave systems using the MMSE criterion," *IEEE Transactions on Communications*, vol. 67, no. 5, pp. 3693–3708, 2019.

[14] M. S. Aljumaily and H. Li, "Machine learning aided hybrid beamforming in massive-MIMO millimeter wave systems," in *2019 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 1–6, 2019.

[15] J. Chen, W. Feng, P. Yang, G. Sobelman, D. Lin, and S. Li, "Hybrid beamforming/-combining for millimeter wave MIMO: A machine learning approach," *IEEE Transactions on Vehicular Technology*, 2020.

[16] H. Huang, Y. Song, J. Yang, G. Gui, and F. Adachi, "Deep-learning-based millimeter-wave massive MIMO for hybrid precoding," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 3, pp. 3027–3032, 2019.

[17] J. Tao, J. Xing, J. Chen, C. Zhang, and S. Fu, "Deep neural hybrid beamforming for multi-user mmwave massive MIMO system," in *2019 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pp. 1–5, 2019.

[18] A. M. Elbir and K. V. Mishra, "Robust hybrid beamforming with quantized deep neural networks," in *2019 IEEE 29th International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2019.

[19] A. Liao, Z. Gao, Y. Wu, H. Wang, and M.-S. Alouini, "2d unitary esprit based super-resolution channel estimation for millimeter-wave massive mimo with hybrid precoding," *IEEE Access*, vol. 5, pp. 24747–24757, 2017.

[20] T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez, "Millimeter wave mobile communications for 5G cellular: It will work!," *IEEE access*, vol. 1, pp. 335–349, 2013.

[21] Z. Pi and F. Khan, "An introduction to millimeter-wave mobile broadband systems," *IEEE communications magazine*, vol. 49, no. 6, pp. 101–107, 2011.

[22] M. K. Samimi and T. S. Rappaport, "3-D millimeter-wave statistical channel model for 5G wireless system design," *IEEE Transactions on Microwave Theory and Techniques*, vol. 64, no. 7, pp. 2207–2225, 2016.

[23] X. Gao, L. Dai, S. Han, C.-L. I, and R. W. Heath, "Energy-efficient hybrid analog and digital precoding for mmwave MIMO systems with large antenna arrays," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 998–1009, 2016.

[24] P. V. Amadori and C. Masouros, "Low RF-complexity millimeter-wave beamspace-MIMO systems by beam selection," *IEEE Transactions on Communications*, vol. 63, no. 6, pp. 2212–2223, 2015.

[25] X. Yu, J. Zhang, and K. B. Letaief, "A hardware-efficient analog network structure for hybrid precoding in millimeter wave systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 2, pp. 282–297, 2018.

[26] X. Zhang, A. Molisch, and S.-Y. Kung, "Variable-phase-shift-based RF-baseband codesign for MIMO antenna selection," *IEEE Transactions on Signal Processing*, vol. 53, no. 11, pp. 4091–4103, 2005.

[27] E. Zhang and C. Huang, "On achieving optimal rate of digital precoder by RF-baseband codesign for MIMO systems," in *2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall)*, pp. 1–5, 2014.

[28] F. Sohrabi, *Hybrid beamforming and one-bit precoding for large-scale antenna arrays*. PhD thesis, University of Toronto (Canada), 2018.

[29] A. M. Elbir and K. V. Mishra, "Joint antenna selection and hybrid beamformer design using unquantized and quantized deep learning networks," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 1677–1688, 2020.

[30] E. M. Lizarraga, G. N. Maggio, and A. A. Dowhuszko, "Hybrid beamforming algorithm using reinforcement learning for millimeter wave wireless systems," in *2019 XVIII Workshop on Information Processing and Control (RPIC)*, pp. 253–258, IEEE, 2019.

[31] Q. Wang, K. Feng, X. Li, and S. Jin, "Precodernet: Hybrid beamforming for millimeter wave systems with deep reinforcement learning," *IEEE Wireless Communications Letters*, vol. 9, no. 10, pp. 1677–1681, 2020.

[32] E. M. Lizarraga, G. N. Maggio, and A. A. Dowhuszko, "Hybrid beamforming algorithm using reinforcement learning for millimeter wave wireless systems," in *2019 XVIII Workshop on Information Processing and Control (RPIC)*, pp. 253–258, 2019.

[33] J. Chen, W. Feng, J. Xing, P. Yang, G. E. Sobelman, D. Lin, and S. Li, "Hybrid beamforming/combining for millimeter wave MIMO: A machine learning approach," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 11353–11368, 2020.

[34] A. M. Elbir, "A deep learning framework for hybrid beamforming without instantaneous CSI feedback," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 11743–11755, 2020.

[35] J. Tao, Q. Wang, S. Luo, and J. Chen, "Constrained deep neural network based hybrid beamforming for millimeter wave massive MIMO systems," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pp. 1–6, 2019.

[36] T. Peken, S. Adiga, R. Tandon, and T. Bose, "Deep learning for svd and hybrid beamforming," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6621–6642, 2020.

[37] S. Huang, Y. Ye, and M. Xiao, "Learning based hybrid beamforming design for full-duplex millimeter wave systems," *IEEE Transactions on Cognitive Communications and Networking*, pp. 1–1, 2020.

[38] T. Peken, *Machine Learning for Channel Estimation and Hybrid Beamforming in Millimeter-Wave Wireless Networks*. PhD thesis, The University of Arizona, 2021.

[39] C. Rusu, R. Méndez-Rial, N. González-Prelcicy, and R. W. Heath, "Low complexity hybrid sparse precoding and combining in millimeter wave mimo systems," in *2015 IEEE International Conference on Communications (ICC)*, pp. 1340–1345, 2015.

[40] I. Ahmed, H. Khammari, A. Shahid, A. Musa, K. S. Kim, E. De Poorter, and I. Moerman, "A survey on hybrid beamforming techniques in 5G: Architecture and system model perspectives," *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 3060–3097, 2018.

[41] J. Tao, J. Chen, J. Xing, S. Fu, and J. Xie, "Autoencoder neural network based intelligent hybrid beamforming design for mmwave massive MIMO systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 6, no. 3, pp. 1019–1030, 2020.

[42] G. Matheron, N. Perrin, and O. Sigaud, "The problem with DDPG: understanding failures in deterministic environments with sparse rewards," *arXiv preprint arXiv:1911.11679*, 2019.

[43] O. El Ayach, S. Rajagopal, S. Abu-Surra, Z. Pi, and R. W. Heath, "Spatially sparse precoding in millimeter wave MIMO systems," *IEEE transactions on wireless communications*, vol. 13, no. 3, pp. 1499–1513, 2014.

[44] A. Alkhateeb, O. El Ayach, G. Leus, and R. W. Heath, "Channel estimation and hybrid precoding for millimeter wave cellular systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 8, no. 5, pp. 831–846, 2014.

[45] W. Ma, C. Qi, Z. Zhang, and J. Cheng, "Sparse channel estimation and hybrid precoding using deep learning for millimeter wave massive MIMO," *IEEE Transactions on Communications*, vol. 68, no. 5, pp. 2838–2849, 2020.

[46] Y. Xiao, Y. Wang, and W. Xiang, "Dimension-deficient channel estimation of hybrid beamforming based on compressive sensing," *IEEE Access*, vol. 7, pp. 13791–13798, 2019.

[47] S. Sun and T. S. Rappaport, "Millimeter wave MIMO channel estimation based on adaptive compressed sensing," in *2017 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 47–53, 2017.

[48] A. Alkhateeb, O. El Ayach, G. Leus, and R. W. Heath, "Channel estimation and hybrid precoding for millimeter wave cellular systems," *IEEE journal of selected topics in signal processing*, vol. 8, no. 5, pp. 831–846, 2014.

[49] J. Lee, G.-T. Gil, and Y. H. Lee, "Channel estimation via orthogonal matching pursuit for hybrid MIMO systems in millimeter wave communications," *IEEE Transactions on Communications*, vol. 64, no. 6, pp. 2370–2386, 2016.

[50] K. Venugopal, A. Alkhateeb, R. W. Heath, and N. G. Prelcic, "Time-domain channel estimation for wideband millimeter wave systems with hybrid architecture," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 6493–6497, 2017.

[51] S. H. Lim, S. Kim, B. Shim, and J. W. Choi, "Efficient beam training and sparse channel estimation for millimeter wave communications under mobility," *IEEE Transactions on Communications*, vol. 68, no. 10, pp. 6583–6596, 2020.

[52] S.-H. Wu and G.-Y. Lu, "Compressive-sensing based beam and channel tracking with reconfigurable hybrid beamforming in mmwave MIMO ofdm systems," in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*, pp. 1–7, 2019.

[53] A. Abdallah, A. Celik, M. M. Mansour, and A. M. Eltawil, "Deep learning based

frequency-selective channel estimation for hybrid mmwave mimo systems," *arXiv preprint arXiv:2102.10847*, 2021.

[54] X. Li and A. Alkhateeb, "Deep learning for direct hybrid precoding in millimeter wave massive mimo systems," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 800–805, IEEE, 2019.

[55] M. B. Mashhadi and D. Gündüz, "Pruning the pilots: Deep learning-based pilot design and channel estimation for mimo-ofdm systems," *IEEE Transactions on Wireless Communications*, 2021.

[56] M. S. Oh, S. Hosseinalipour, T. Kim, C. G. Brinton, and D. J. Love, "Channel estimation via successive denoising in mimo ofdm systems: A reinforcement learning approach," *arXiv preprint arXiv:2101.10300*, 2021.

[57] Y.-S. Jeon, J. Li, N. Tavangaran, and H. V. Poor, "Data-aided channel estimator for mimo systems via reinforcement learning," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2020.

[58] K. Dovelos, M. Matthaiou, H. Q. Ngo, and B. Bellalta, "Channel estimation and hybrid combining for wideband terahertz massive mimo systems," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 6, pp. 1604–1620, 2021.

[59] W. Ma, C. Qi, and G. Y. Li, "High-resolution channel estimation for frequency-selective mmwave massive mimo systems," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3517–3529, 2020.

[60] S. Gao, X. Cheng, and L. Yang, "Estimating doubly-selective channels for hybrid mmwave massive mimo systems: A doubly-sparse approach," *IEEE Transactions on Wireless Communications*, vol. 19, no. 9, pp. 5703–5715, 2020.

[61] H. Xie and N. González-Prelcic, "Dictionary learning for channel estimation in hybrid frequency-selective mmwave mimo systems," *IEEE Transactions on Wireless Communications*, vol. 19, no. 11, pp. 7407–7422, 2020.

[62] O. Osanaiye, A. S. Alfa, and G. P. Hancke, "A statistical approach to detect jamming attacks in wireless sensor networks," *Sensors*, vol. 18, no. 6, p. 1691, 2018.

[63] H. I. Reyes and N. Kaabouch, "Jamming and lost link detection in wireless networks with fuzzy logic," *International Journal of Scientific & Engineering Research*, vol. 4, no. 2, pp. 1–7, 2013.

[64] B. Upadhyaya, S. Sun, and B. Sikdar, "Machine learning-based jamming detection in wireless IoT networks," in *2019 IEEE VTS Asia Pacific Wireless Communications Symposium (APWCS)*, pp. 1–5, 2019.

[65] M. Bensalem, S. K. Singh, and A. Jukan, "On detecting and preventing jamming attacks with machine learning in optical networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2019.

[66] Y. Arjoune, F. Salahdine, M. S. Islam, E. Ghribi, and N. Kaabouch, "A novel jamming attacks detection approach based on machine learning for wireless communication," in *2020 International Conference on Information Networking (ICOIN)*, pp. 459–464, IEEE, 2020.

[67] S. Gecgel, C. Goztepe, and G. K. Kurt, "Jammer detection based on artificial neural networks: A measurement study," in *Proceedings of the ACM Workshop on Wireless Security and Machine Learning*, pp. 43–48, 2019.

[68] G.-H. Lee, J. Jo, and C. H. Park, "Jamming prediction for radar signals using machine learning methods," *Security and Communication Networks*, vol. 2020, 2020.

[69] Y. Junfei, L. Jingwen, S. Bing, and J. Yuming, "Barrage jamming detection and classification based on convolutional neural network for synthetic aperture radar," in *IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium*, pp. 4583–4586, 2018.

[70] T. Erpek, Y. E. Sagduyu, and Y. Shi, "Deep learning for launching and mitigating wireless jamming attacks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 1, pp. 2–14, 2019.

[71] N. I. Mowla, N. H. Tran, I. Doh, and K. Chae, "AFRL: Adaptive federated reinforcement learning for intelligent jamming defense in FANET," *Journal of Communications and Networks*, vol. 22, no. 3, pp. 244–258, 2020.

[72] O. Puñal, I. Aktaş, C.-J. Schnelke, G. Abidin, K. Wehrle, and J. Gross, "Machine learning-based jamming detection for ieee 802.11: Design and experimental evaluation," in *Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*, pp. 1–10, IEEE, 2014.

[73] C. Natalino, M. Schiano, A. Di Giglio, L. Wosinska, and M. Furdek, "Field demonstration of machine-learning-aided detection and identification of jamming attacks in optical networks," in *2018 European Conference on Optical Communication (ECOC)*, pp. 1–3, 2018.

[74] M. Bensalem, S. K. Singh, and A. Jukan, "On detecting and preventing jamming attacks with machine learning in optical networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, IEEE, 2019.

[75] Z. Feng and C. Hua, "Machine learning-based RF jamming detection in wireless networks," in *2018 Third International Conference on Security of Smart Cities, Industrial Control System and Communications (SSIC)*, pp. 1–6, IEEE, 2018.

[76] D. Karagiannis and A. Argyriou, "Jamming attack detection in a pair of RF communicating vehicles using unsupervised machine learning," *Vehicular Communications*, vol. 13, pp. 56–63, 2018.

[77] M. A. Munir and A. R. M. Maud, "Direct-sequence spread spectrum with variable spreading sequence for jamming immunity," in *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*, pp. 933–937, 2019.

[78] S. Djuraev, J.-G. Choi, K.-S. Sohn, and S. Y. Nam, "Channel hopping scheme to mitigate jamming attacks in wireless LANs," *EURASIP Journal on Wireless Communications and Networking*, vol. 2017, no. 1, p. 11, 2017.

[79] Q. Wang, T. Nguyen, K. Pham, and H. Kwon, "Mitigating jamming attack: A game-theoretic perspective," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 7, pp. 6063–6074, 2018.

[80] Z. Zhang, Q. Wu, B. Zhang, and J. Peng, "Intelligent anti-jamming relay communication system based on reinforcement learning," in *2019 2nd International Conference on Communication Engineering and Technology (ICCET)*, pp. 52–56, 2019.

[81] Z. Fu, A. Hornbostel, J. Hammesfahr, and A. Konovaltsev, "Suppression of multipath and jamming signals by digital beamforming for gps/galileo applications," *GPS solutions*, vol. 6, no. 4, pp. 257–264, 2003.

[82] L.-M. Li and L. Milstein, "Rejection of narrow-band interference in pn spread-spectrum systems using transversal filters," *IEEE Transactions on Communications*, vol. 30, no. 5, pp. 925–928, 1982.

[83] G. J. Saulnier, "Suppression of narrowband jammers in a spread-spectrum receiver

using transform-domain adaptive filtering," *IEEE Journal on selected Areas in communications*, vol. 10, no. 4, pp. 742–749, 1992.

[84] A. Yousaf and A. Loan, "Effect of jamming technique on the performance of direct sequence spread spectrum modem," in *13th International Conference on Advanced Communication Technology (ICACT2011)*, pp. 874–877, 2011.

[85] G. Akcan and M. E. Çek, "Direct sequence spread-spectrum based covert communication using random pulse width modulation," in *2019 27th Signal Processing and Communications Applications Conference (SIU)*, pp. 1–4, 2019.

[86] Y.-h. Zhu, P.-h. Zhang, and M.-y. Deng, "Simulation research on direct sequence spread spectrum communication system over rayleigh fading channels," in *Proceedings of 2011 International Conference on Electronic Mechanical Engineering and Information Technology*, vol. 8, pp. 4317–4321, 2011.

[87] Y. Arjoune and S. Faruque, "Smart jamming attacks in 5G new radio: A review," in *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 1010–1015, IEEE, 2020.

[88] H. Simon, *Communication systems*. Springer Nature, 1999.

[89] K. Karkatzounis, "Performance evaluation of different jamming strategies over uncoded noncoherent fast frequency hopping mfsk communication systems," tech. rep., NAVAL POSTGRADUATE SCHOOL MONTEREY CA DEPT OF INFORMATIONAL SCIENCES, 2004.

[90] F. Salahdine and N. Kaabouch, "Security threats, detection, and countermeasures for physical layer in cognitive radio networks: A survey," *Physical Communication*, vol. 39, p. 101001, 2020.

[91] A. Mpitziopoulos and D. Gavalas, "An effective defensive node against jamming attacks in sensor networks," *Security and Communication Networks*, vol. 2, no. 2, pp. 145–163, 2009.

[92] A. E. Mansour, W. M. Saad, and S. H. El Ramly, "Adaptive chaotic frequency hopping," in *2015 Tenth International Conference on Computer Engineering Systems (ICCES)*, pp. 328–331, 2015.

[93] Dhivyadharshini and B. Gopalakrishnan, "Comparative analysis of FH and CFH spread spectrum under different jammers," in *2020 International Conference on Communication and Signal Processing (ICCSP)*, pp. 1361–1365, 2020.

[94] W. Xu, T. Wood, W. Trappe, and Y. Zhang, "Channel surfing and spatial retreats: defenses against wireless denial of service," in *Proceedings of the 3rd ACM workshop on Wireless security*, pp. 80–89, ACM, 2004.

[95] S. Djuraev and S. Y. Nam, "Channel-hopping-based jamming mitigation in wireless lan considering throughput and fairness," *Electronics*, vol. 9, no. 11, p. 1749, 2020.

[96] W. Xu, W. Trappe, and Y. Zhang, "Anti-jamming timing channels for wireless networks," in *Proceedings of the first ACM conference on Wireless network security*, pp. 203–213, ACM, 2008.

[97] Q. Xiaolin, J. Tao, Q. Xiaohui, Z. Min, Y. Shuqing, and Z. Qunxing, "Anti-millimeter wave polarization agile active jamming," in *2007 International Conference on Microwave and Millimeter Wave Technology*, pp. 1–4, IEEE, 2007.

[98] J. Tao, Q. Xiaohui, Q. Xiaolin, Z. Min, Y. Shuqing, and Z. Qunxing, "Anti-full polarization active jamming," in *2007 2nd IEEE Conference on Industrial Electronics and Applications*, pp. 2718–2722, IEEE, 2007.

[99] J.-W. Tao, "Performance analysis for interference and noise canceller based on hypercomplex and spatio-temporal-polarisation processes," *IET Radar, Sonar & Navigation*, vol. 7, no. 3, pp. 277–286, 2013.

[100] A. Poelman, "Virtual polarisation adaptation a method of increasing the detection capability of a radar system through polarisation-vector processing," in *IEE Proceedings F (Communications, Radar and Signal Processing)*, vol. 128, pp. 261–270, IET, 1981.

[101] L. Song, H. Qi, X. Qiao, and X. Meng, "Scheme of adaptive polarization filtering based on kalman model," *Journal of Systems Engineering and Electronics*, vol. 17, no. 1, pp. 13–18, 2006.

[102] W. Xuesong, C. Yuliang, D. Dahai, X. Shunping, and Z. Zhaowen, "Band characteristics of sinr polarization filter," *IEEE transactions on antennas and propagation*, vol. 55, no. 4, pp. 1148–1154, 2007.

[103] A. Aldarraji, L. Hong, and S. Shetty, "Polarized beamforming for near-field wireless jamming attacks mitigation," in *SoutheastCon 2017*, pp. 1–5, 2017.

[104] X. Tang, D. Wang, R. Zhang, Z. Chu, and Z. Han, "Jamming mitigation via aerial reconfigurable intelligent surface: Passive beamforming and deployment optimization," *IEEE Transactions on Vehicular Technology*, pp. 1–1, 2021.

[105] T. Erpek, Y. E. Sagduyu, and Y. Shi, "Deep learning for launching and mitigating wireless jamming attacks," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 1, pp. 2–14, 2018.

[106] J. Mitola and G. Q. Maguire, "Cognitive radio: making software radios more personal," *IEEE personal communications*, vol. 6, no. 4, pp. 13–18, 1999.

[107] L. Huang, G. Zhu, and X. Du, "Cognitive femtocell networks: an opportunistic spectrum access for future indoor wireless coverage," *IEEE wireless communications*, vol. 20, no. 2, pp. 44–51, 2013.

[108] W. Wang, G. Yu, and A. Huang, "Cognitive radio enhanced interference coordination for femtocell networks," *IEEE Communications Magazine*, vol. 51, no. 6, pp. 37–43, 2013.

[109] N. Kaabouch, *Handbook of research on software-defined and cognitive radio technologies for dynamic spectrum management*. IGI global, 2014.

[110] P. Rawat, K. D. Singh, and J. M. Bonnin, "Cognitive radio for m2m and internet of things: A survey," *Computer Communications*, vol. 94, pp. 1–29, 2016.

[111] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols, and applications," *IEEE communications surveys & tutorials*, vol. 17, no. 4, pp. 2347–2376, 2015.

[112] C.-H. Lee and C.-Y. Shih, "Coverage analysis of cognitive femtocell networks," *IEEE Wireless Communications Letters*, vol. 3, no. 2, pp. 177–180, 2014.

[113] A. Ranjan, B. Singh, *et al.*, "Design and analysis of spectrum sensing in cognitive radio based on energy detection," in *2016 International Conference on Signal and Information Processing (IConSIP)*, pp. 1–5, IEEE, 2016.

[114] D. M. M. Plata and Á. G. A. Reátiga, "Evaluation of energy detection for spectrum sensing based on the dynamic selection of detection-threshold," *Procedia Engineering*, vol. 35, pp. 135–143, 2012.

[115] L. Ruan, Y. Li, W. Cheng, and Z. Wu, "A robust threshold optimization approach for energy detection based spectrum sensing with noise uncertainty," in *2015 IEEE 10th*

*Conference on Industrial Electronics and Applications (ICIEA)*, pp. 161–165, IEEE, 2015.

[116] S. Atapattu, C. Tellambura, and H. Jiang, "Energy detection based cooperative spectrum sensing in cognitive radio networks," *IEEE Transactions on wireless communications*, vol. 10, no. 4, pp. 1232–1241, 2011.

[117] M. Z. Alom, T. K. Godder, M. N. Morshed, and A. Maali, "Enhanced spectrum sensing based on energy detection in cognitive radio network using adaptive threshold," in *2017 International Conference on Networking, Systems and Security (NSysS)*, pp. 138–143, IEEE, 2017.

[118] Y. Arjoune, Z. El Mrabet, H. El Ghazi, and A. Tamtaoui, "Spectrum sensing: Enhanced energy detection technique based on noise measurement," in *2018 IEEE 8th annual computing and communication workshop and conference (CCWC)*, pp. 828–834, IEEE, 2018.

[119] A. Eslami and S. Karamzadeh, "Performance analysis of double threshold energy detection-based spectrum sensing in low snrs over nakagami-m fading channels with noise uncertainty," in *2016 24th Signal Processing and Communication Application Conference (SIU)*, pp. 309–312, IEEE, 2016.

[120] X. Ling, B. Wu, H. Wen, P.-H. Ho, Z. Bao, and L. Pan, "Adaptive threshold control for energy detection based spectrum sensing in cognitive radios," *IEEE Wireless Communications Letters*, vol. 1, no. 5, pp. 448–451, 2012.

[121] Z. Bao, B. Wu, P.-H. Ho, and X. Ling, "Adaptive threshold control for energy detection based spectrum sensing in cognitive radio networks," in *2011 IEEE Global Telecommunications Conference-GLOBECOM 2011*, pp. 1–5, IEEE, 2011.

[122] Y. Zeng, Y.-C. Liang, and R. Zhang, "Blindly combined energy detection for spectrum sensing in cognitive radio," *IEEE signal processing letters*, vol. 15, pp. 649–652, 2008.

[123] D. R. Joshi, D. C. Popescu, and O. A. Dobre, "Adaptive spectrum sensing with noise variance estimation for dynamic cognitive radio systems," in *2010 44th Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–5, IEEE, 2010.

[124] M. Sarker, "Energy detector based spectrum sensing by adaptive threshold for low snr in cr networks," in *2015 24th wireless and optical communication conference (WOCC)*, pp. 118–122, IEEE, 2015.

[125] J. Wu, T. Luo, and G. Yue, "An energy detection algorithm based on double-threshold in cognitive radio systems," in *2009 First International Conference on Information Science and Engineering*, pp. 493–496, IEEE, 2009.

[126] P. S. Yawada and A. J. Wei, "Cyclostationary detection based on non-cooperative spectrum sensing in cognitive radio network," in *2016 IEEE International Conference on Cyber Technology in Automation, Control, and Intelligent Systems (CYBER)*, pp. 184–187, IEEE, 2016.

[127] I. Ilyas, S. Paul, A. Rahman, and R. K. Kundu, "Comparative evaluation of cyclostationary detection based cognitive spectrum sensing," in *2016 IEEE 7th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pp. 1–7, IEEE, 2016.

[128] F. Salahdine, H. El Ghazi, N. Kaabouch, and W. F. Fihri, "Matched filter detection with dynamic threshold for cognitive radio networks," in *2015 international conference on wireless networks and mobile communications (WINCOM)*, pp. 1–6, IEEE, 2015.

[129] X. Zhang, R. Chai, and F. Gao, "Matched filter based spectrum sensing and power

level detection for cognitive radio network," in *2014 IEEE global conference on signal and information processing (GlobalSIP)*, pp. 1267–1270, IEEE, 2014.

[130] C. Jiang, Y. Li, W. Bai, Y. Yang, and J. Hu, "Statistical matched filter based robust spectrum sensing in noise uncertainty environment," in *2012 IEEE 14th International Conference on Communication Technology*, pp. 1209–1213, IEEE, 2012.

[131] Q. Lv and F. Gao, "Matched filter based spectrum sensing and power level recognition with multiple antennas," in *2015 IEEE China summit and international conference on signal and information processing (ChinaSIP)*, pp. 305–309, IEEE, 2015.

[132] Z. Tian and G. B. Giannakis, "A wavelet approach to wideband spectrum sensing for cognitive radios," in *2006 1st international conference on cognitive radio oriented wireless networks and communications*, pp. 1–5, IEEE, 2006.

[133] Y. Zhao, Y. Wu, J. Wang, X. Zhong, and L. Mei, "Wavelet transform for spectrum sensing in cognitive radio networks," in *2014 International Conference on Audio, Language and Image Processing*, pp. 565–569, IEEE, 2014.

[134] X. Han, W. Xu, K. Niu, and Z. He, "A novel wavelet-based energy detection for compressive spectrum sensing," in *2013 IEEE 77th Vehicular Technology Conference (VTC Spring)*, pp. 1–5, IEEE, 2013.

[135] A. Kumar, S. Saha, and R. Bhattacharya, "Improved wavelet transform based edge detection for wide band spectrum sensing in cognitive radio," in *2016 USNC-URSI Radio Science Meeting*, pp. 21–22, IEEE, 2016.

[136] S. E. El-Khamy, M. S. El-Mahallawy, and E.-N. S. Youssef, "Improved wideband spectrum sensing techniques using wavelet-based edge detection for cognitive radio," in

*2013 international conference on computing, networking and communications (ICNC)*, pp. 418–423, IEEE, 2013.

[137] Z. Quan, S. Cui, A. H. Sayed, and H. V. Poor, "Optimal multiband joint detection for spectrum sensing in cognitive radio networks," *IEEE transactions on signal processing*, vol. 57, no. 3, pp. 1128–1140, 2008.

[138] K. Sharma and A. Sharma, "Design of cosine modulated filter banks exploiting spline function for spectrum sensing in cognitive radio applications," in *2016 IEEE 1st international conference on power electronics, intelligent control and energy systems (ICPEICES)*, pp. 1–5, IEEE, 2016.

[139] M. Lin, A. P. Vinod, and C. M. S. See, "A new flexible filter bank for low complexity spectrum sensing in cognitive radios," *Journal of Signal Processing Systems*, vol. 62, no. 2, pp. 205–215, 2011.

[140] B. Farhang-Boroujeny, "Filter bank spectrum sensing for cognitive radios," *IEEE Transactions on signal processing*, vol. 56, no. 5, pp. 1801–1811, 2008.

[141] H. Sun, W.-Y. Chiu, and A. Nallanathan, "Adaptive compressive spectrum sensing for wideband cognitive radios," *IEEE Communications Letters*, vol. 16, no. 11, pp. 1812–1815, 2012.

[142] Z. Qin, Y. Gao, and C. G. Parini, "Data-assisted low complexity compressive spectrum sensing on real-time signals under sub-nyquist rate," *IEEE Transactions on Wireless Communications*, vol. 15, no. 2, pp. 1174–1185, 2015.

[143] Y. Ma, X. Zhang, and Y. Gao, "Joint sub-nyquist spectrum sensing scheme with geolocation database over tv white space," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 5, pp. 3998–4007, 2017.

[144] Y. Ma, X. Zhang, and Y. G. Q. Mary, "An efficient joint sub-nyquist spectrum sensing scheme with geolocation database over tv white space," in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2017.

[145] A. M. Mikaeil, B. Guo, and Z. Wang, "Machine learning to data fusion approach for cooperative spectrum sensing," in *2014 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery*, pp. 429–434, IEEE, 2014.

[146] Y. Lu, P. Zhu, D. Wang, and M. Fattouche, "Machine learning techniques with probability vector for cooperative spectrum sensing in cognitive radio networks," in *2016 IEEE wireless communications and networking conference*, pp. 1–6, IEEE, 2016.

[147] D. Wang and Z. Yang, "An novel spectrum sensing scheme combined with machine learning," in *2016 9th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP-BMEI)*, pp. 1293–1297, IEEE, 2016.

[148] A. Attar, H. Tang, A. V. Vasilakos, F. R. Yu, and V. C. Leung, "A survey of security challenges in cognitive radio networks: Solutions and future research directions," *Proceedings of the IEEE*, vol. 100, no. 12, pp. 3172–3186, 2012.

[149] S. Mapunya and M. Velempini, "Investigating spectrum sensing security threats in cognitive radio networks," in *Ad Hoc Networks*, pp. 60–68, Springer, 2018.

[150] J. Rodriguez, *Fundamentals of 5G mobile networks*. John Wiley & Sons, 2015.

[151] L. Zhang, G. Ding, Q. Wu, Y. Zou, Z. Han, and J. Wang, "Byzantine attack and defense in cognitive radio networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1342–1363, 2015.

[152] Y. E. Sagduyu, "Securing cognitive radio networks with dynamic trust against spec-

trum sensing data falsification," in *2014 IEEE Military Communications Conference*, pp. 235–241, IEEE, 2014.

[153] A. S. Rawat, P. Anand, H. Chen, and P. K. Varshney, "Collaborative spectrum sensing in the presence of byzantine attacks in cognitive radio networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 2, pp. 774–786, 2010.

[154] M. Khasawneh and A. Agarwal, "A collaborative approach towards securing spectrum sensing in cognitive radio networks," *Procedia Computer Science*, vol. 94, pp. 302–309, 2016.

[155] M. Khasawneh and A. Agarwal, "A collaborative approach for monitoring nodes behavior during spectrum sensing to mitigate multiple attacks in cognitive radio networks," *Security and Communication Networks*, vol. 2017, 2017.

[156] A. Vempaty, K. Agrawal, P. Varshney, and H. Chen, "Adaptive learning of byzantines' behavior in cooperative spectrum sensing," in *2011 IEEE wireless communications and networking conference*, pp. 1310–1315, IEEE, 2011.

[157] M. Wang, B. Liu, and C. Zhang, "Detection of collaborative ssdf attacks using abnormality detection algorithm in cognitive radio networks," in *2013 IEEE International Conference on Communications Workshops (ICC)*, pp. 342–346, IEEE, 2013.

[158] Y. Li and Q. Peng, "Achieving secure spectrum sensing in presence of malicious attacks utilizing unsupervised machine learning," in *MILCOM 2016-2016 IEEE Military Communications Conference*, pp. 174–179, IEEE, 2016.

[159] G. Nie, G. Ding, L. Zhang, and Q. Wu, "Byzantine defense in collaborative spectrum sensing via bayesian learning," *IEEE Access*, vol. 5, pp. 20089–20098, 2017.

[160] F. Farmani, M. Abbasi-Jannatabad, and R. Berangi, "Detection of ssdf attack using svdd algorithm in cognitive radio networks," in *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*, pp. 201–204, IEEE, 2011.

[161] Z. Cheng, T. Song, J. Zhang, J. Hu, Y. Hu, L. Shen, X. Li, and J. Wu, "Self-organizing map-based scheme against probabilistic ssdf attack in cognitive radio networks," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–6, IEEE, 2017.

[162] S. L. Mohammed, M. H. Alsharif, S. K. Gharghan, I. Khan, and M. Albreem, "Robust hybrid beamforming scheme for millimeter-wave massive-mimo 5g wireless networks," *Symmetry*, vol. 11, no. 11, p. 1424, 2019.

[163] F. Sohrabi and W. Yu, "Hybrid digital and analog beamforming design for large-scale antenna arrays," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 3, pp. 501–513, 2016.

[164] B. Xu, J. Z. Huang, G. Williams, Q. Wang, and Y. Ye, "Classifying very high-dimensional data with random forests built from small subspaces," *International Journal of Data Warehousing and Mining (IJDWM)*, vol. 8, no. 2, pp. 44–63, 2012.

[165] N. Sirikulviriya and S. Sinthupinyo, "Integration of rules from a random forest," in *International Conference on Information and Electronics Engineering*, vol. 6, pp. 194–198, 2011.

[166] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," *arXiv preprint arXiv:1509.06461*, 2015.

[167] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *International Conference on Machine Learning*, pp. 1861–1870, PMLR, 2018.

[168] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*, pp. 1587–1596, PMLR, 2018.

[169] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[170] M. Rihan, T. Abed Soliman, C. Xu, L. Huang, and M. I. Dessouky, "Taxonomy and performance evaluation of hybrid beamforming for 5G and beyond systems," *IEEE Access*, vol. 8, pp. 74605–74626, 2020.

[171] T. S. Rappaport, S. Sun, R. Mayzus, H. Zhao, Y. Azar, K. Wang, G. N. Wong, J. K. Schulz, M. Samimi, and F. Gutierrez, "Millimeter wave mobile communications for 5G cellular: It will work!," *IEEE Access*, vol. 1, pp. 335–349, 2013.

[172] Y. Wang, N. J. Myers, N. González-Prelcic, and R. W. Heath Jr, "Deep learning-based compressive beam alignment in mmwave vehicular systems," *arXiv preprint arXiv:2103.00125*, 2021.

[173] E. J. Candes and M. B. Wakin, "An introduction to compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[174] W. Yin, S. Morgan, J. Yang, and Y. Zhang, "Practical compressive sensing with toeplitz and circulant matrices," in *Visual Communications and Image Processing 2010*, vol. 7744, p. 77440K, International Society for Optics and Photonics, 2010.

[175] A. Amini and F. Marvasti, "Deterministic construction of binary, bipolar, and ternary compressed sensing matrices," *IEEE Transactions on Information Theory*, vol. 57, no. 4, pp. 2360–2370, 2011.

[176] L. Applebaum, S. D. Howard, S. Searle, and R. Calderbank, "Chirp sensing codes: Deterministic compressed sensing measurements for fast recovery," *Applied and Computational Harmonic Analysis*, vol. 26, no. 2, pp. 283–290, 2009.

[177] N. Yang and J. Li-li, "Measurement matrix construction algorithm for compressed sensing based on qc-ldpc matrix," *International Journal of Grid and Distributed Computing*, vol. 9, no. 2, pp. 121–130, 2016.

[178]

[179] C. Zhuoran, Z. Honglin, J. Min, W. Gang, and S. Jingshi, "An improved hadamard measurement matrix based on walsh code for compressive sensing," in *2013 9th International Conference on Information, Communications Signal Processing*, pp. 1–4, 2013.

[180] T. L. Nguyen and Y. Shin, "Deterministic sensing matrices in compressive sensing: a survey," *The Scientific World Journal*, vol. 2013, 2013.

[181] H. Rauhut, J. Romberg, and J. A. Tropp, "Restricted isometries for partial random circulant matrices," *Applied and Computational Harmonic Analysis*, vol. 32, no. 2, pp. 242–254, 2012.

[182] J. Haupt, W. U. Bajwa, G. Raz, and R. Nowak, "Toeplitz compressed sensing matrices with applications to sparse channel estimation," *IEEE Transactions on Information Theory*, vol. 56, no. 11, pp. 5862–5875, 2010.

[183] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[184] R. Garg and R. Khandekar, "Gradient descent with sparsification: an iterative algorithm for sparse recovery with restricted isometry property," in *Proceedings of the 26th annual international conference on machine learning*, pp. 337–344, 2009.

[185] J. A. Tropp and A. C. Gilbert, "Signal recovery from random measurements via orthogonal matching pursuit," *IEEE Transactions on Information Theory*, vol. 53, no. 12, pp. 4655–4666, 2007.

[186] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and computational harmonic analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[187] S. Ji, Y. Xue, and L. Carin, "Bayesian compressive sensing," *IEEE Transactions on Signal Processing*, vol. 56, no. 6, pp. 2346–2356, 2008.

[188] D. Baron, S. Sarvotham, and R. G. Baraniuk, "Bayesian compressive sensing via belief propagation," *IEEE Transactions on Signal Processing*, vol. 58, no. 1, pp. 269–280, 2010.

[189] A. Nascimento, W. Frazão, A. Oliveira, D. Gomes, and A. Klautau, "Multimodal dataset for machine learning applied to telecommunications," *SBrT, Santa Catarina, Brazil*, pp. 13–16, 2020.

[190] M. Lichtman, R. Rao, V. Marojevic, J. Reed, and R. P. Jover, "5G NR jamming, spoofing, and sniffing: threat assessment and mitigation," in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, IEEE, 2018.

[191] A. Jarwan, A. Sabbah, M. Ibnkahla, and O. Issa, "LTE-based public safety networks:

a survey," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1165–1187, 2019.

[192] A. Kumbhar, F. Koohifar, I. Güvenç, and B. Mueller, "A survey on legacy and emerging technologies for public safety communications," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 97–124, 2016.

[193] K. Grover, A. Lim, and Q. Yang, "Jamming and anti–jamming techniques in wireless networks: a survey," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 17, no. 4, pp. 197–215, 2014.

[194] M. Lichtman, J. D. Poston, S. Amuru, C. Shahriar, T. C. Clancy, R. M. Buehrer, and J. H. Reed, "A communications jamming taxonomy," *IEEE Security & Privacy*, vol. 14, no. 1, pp. 47–54, 2016.

[195] K. Pelechrinis, M. Iliofotou, and S. V. Krishnamurthy, "Denial of service attacks in wireless networks: The case of jammers," *IEEE Communications surveys & tutorials*, vol. 13, no. 2, pp. 245–257, 2010.

[196] W. Xu, W. Trappe, Y. Zhang, and T. Wood, "The feasibility of launching and detecting jamming attacks in wireless networks," in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pp. 46–57, ACM, 2005.

[197] D. Nguyen, C. Sahin, B. Shishkin, N. Kandasamy, and K. R. Dandekar, "A real-time and protocol-aware reactive jamming framework built on software-defined radios," in *Proceedings of the 2014 ACM workshop on Software radio implementation forum*, pp. 15–22, ACM, 2014.

[198] M. Strasser, B. Danev, and S. Čapkun, "Detection of reactive jamming in sensor networks," *ACM Transactions on Sensor Networks (TOSN)*, vol. 7, no. 2, p. 16, 2010.

[199] X. Wu and V. Kumar, *The top ten algorithms in data mining*. CRC press, 2009.

[200] J. Friedman, T. Hastie, R. Tibshirani, *et al.*, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The annals of statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[201] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 71–80, 2000.

[202] G. T. . version 14.0.0 Release 14, *5G; Study on channel model for frequencies from 0.5 to 100 GHz*. 2017.

[203] N. Japkowicz and M. Shah, *Evaluating learning algorithms: a classification perspective*. Cambridge University Press, 2011.

[204] S. Faruque, *Radio Frequency Multiple Access Techniques Made Easy*. Springer, 2019.

[205] K. S. Gilhousen, I. M. Jacobs, R. Padovani, A. J. Viterbi, L. A. Weaver, and C. E. Wheatley, "On the capacity of a cellular CDMA system," *IEEE transactions on vehicular technology*, vol. 40, no. 2, pp. 303–312, 1991.

[206] J. Ren and T. Li, "CDMA physical layer built-in security enhancement," in *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall (IEEE Cat. No. 03CH37484)*, vol. 3, pp. 2157–2161, IEEE, 2003.

[207] T. Li, J. Ren, Q. Ling, and A. Jain, "Physical layer built-in security analysis and enhancement of CDMA systems," in *MILCOM 2005-2005 IEEE Military Communications Conference*, pp. 956–962, IEEE, 2005.

[208] T. Song, K. Zhou, and T. Li, "CDMA system design and capacity analysis under disguised jamming," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 11, pp. 2487–2498, 2016.

[209] T. Li, T. Song, and Y. Liang, "Enhanced CDMA system with secure scrambling," in *Wireless Communications under Hostile Jamming: Security and Efficiency*, pp. 15–44, Springer, 2018.